CMPE 297, INSTRUCTOR: JORJETA JETCHEVA

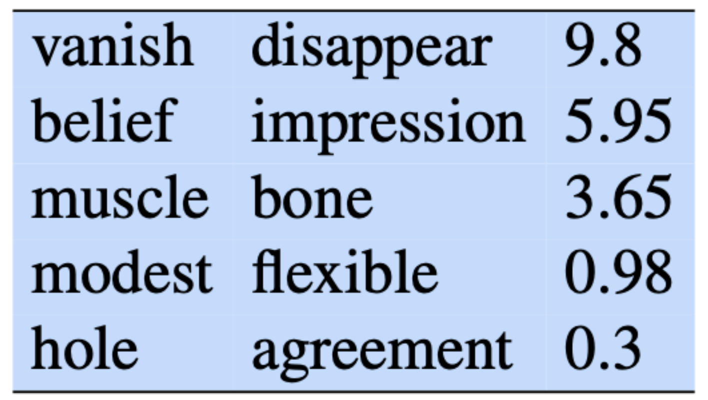
WORD AND DOCUMENT EMBEDDINGS

SEMANTICS OF WORDS

- Lexical semantics: the study of the meaning of words (aka word senses)
- Synonymy is defined between words (not word senses)
    - Two words are synonymous if they can be substituted in any sentence while preserving the truth of the sentence
- Principle of contrast (Girard 1718, Breal 1897, Clark 1987)
    - A difference in linguistic form is always associated with a difference in meaning, i.e., no two words are exactly the same
    - E.g., H_2O and water are used in different genres

WORD SIMILARITY VS. RELATEDNESS

- Quantifying how similar two words are plays a role in many NLP tasks, including question-answering, paraphrasing & summarization
  - Human annotations serve as a baseline
- Word relatedness or association
  - Coffee & cup share no features but co-participate in the same event
- Related Concepts
  - Semantic field: a set of words that cover a particular domain and have some structured relations with each other, e.g., hospital field: surgeon, scalpel, nurse, medicines
  - Semantic frame: a set of words that denote perspectives or participants in a particular type of event, e.g., a commercial transaction with a Buyer and Seller roles who have different perspectives on the transaction

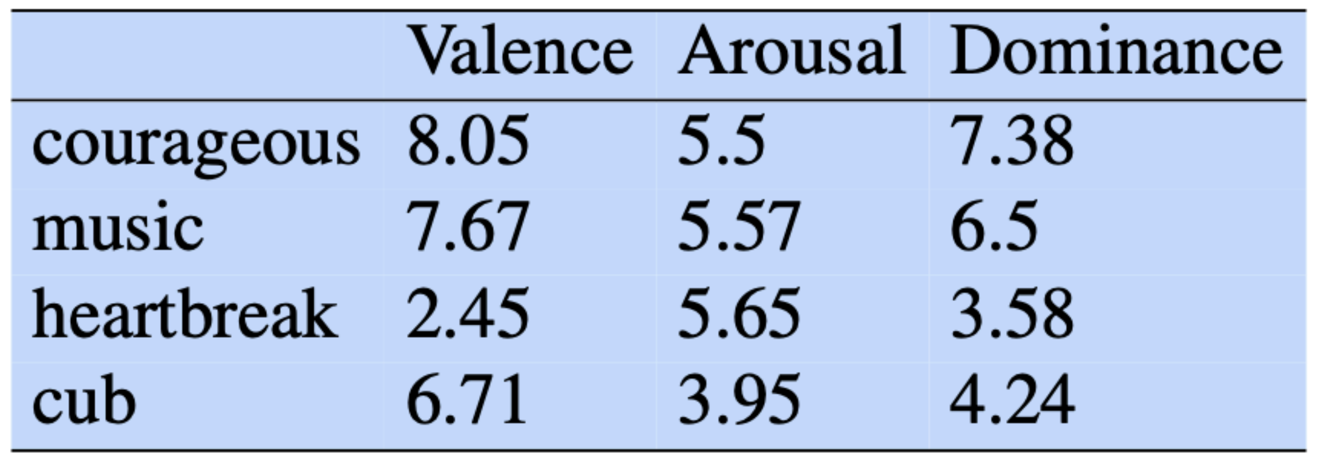


vanish	disappear	9.8
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

SimLex-999 dataset, Scale of 0-10

CONNOTATIONS

- Connotation: aspects of word meaning related to the writer's or reader's emotions, opinions, judgements, etc.
    - Can be positive, negative and nuanced, e.g., fake, knockoff, forgery vs. replica, reproduction
- Dimensions of affective meaning
    - Valence: pleasantness (happy vs. annoyed)
    - Arousal: intensity of emotion (enthusiastic vs. calm)
    - Dominance: degree of control (control vs. influence)
- Osgood et al. (1957) decided to use the 3 dimensions above as a vector to capture word meaning



	Valence	Arousal	Dominance
courageous	8.05	5.5	7.38
music	7.67	5.57	6.5
heartbreak	2.45	5.65	3.58
cub	6.71	3.95	4.24

EXAMPLE BASIC VECTOR EMBEDDING

BASIC WORD EMBEDDING: 1-HOT VECTOR

- Words can be encoded as 1-hot vectors
  - 1-hot vector: a vector with size = size of the vocabulary with a 1 at the index for the word it represents
- No semantic information

One-hot encoding

	it	is	puppy	cat	pen	a	this
it	1	0	0	0	0	0	0
is	0	1	0	0	0	0	0
puppy	0	0	1	0	0	0	0
..	..	..	..	..	..	..	..

SEMANTIC VECTOR EMBEDDINGS

DISTRIBUTIONAL HYPOTHESIS

- Distributional hypothesis: there is a link between how words are distributed and what they mean
- Observations by Joos (1950), Harris (1954), Firth (1957)
    - Similar words (e.g., “occulist” & “eye-doctor”) occur in similar environment/context (e.g., near “eye” and “exam”)
    - Difference in the meanings of two words corresponds to the difference in their “environments”

Vector embeddings are learned representations of words that are based on word distributions (in line with the distributional hypothesis and Osgood’s vector representation approach)

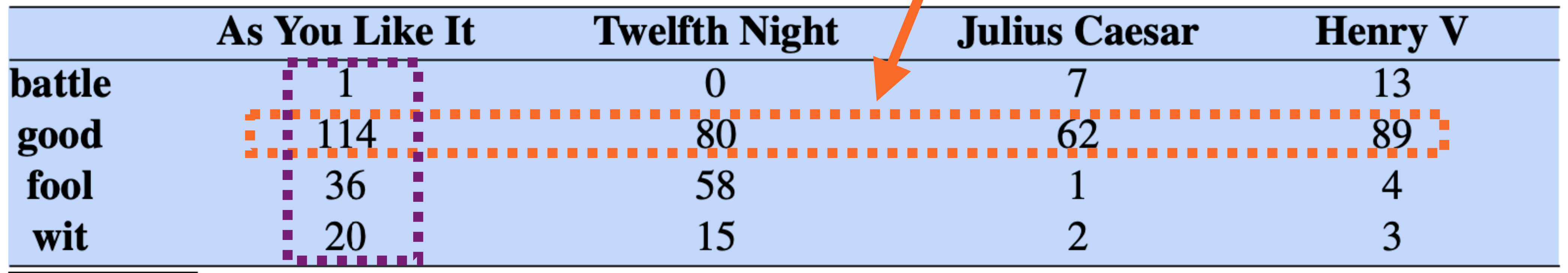
VECTOR EMBEDDINGS

- In a vector embedding, each word or document is a point in multi-dimensional space
- Embedding is a mapping from one space to another
  - In this case, embedding is a from words to numerical vectors
- Sparse embeddings - based on word counts and co-occurrence (e.g., tf-idf)
  - Sparse because they contain a lot of zeros (most words do not appear in the context of other words)
- Dense embeddings - e.g., automatically learned by a Deep Learning network such as word2vec embeddings
  - The word "embedding" is often reserved only for dense embeddings

TERM-DOCUMENT MATRIX

- All words are extracted from the document corpus & added to a "vocabulary"
- A matrix is created with all the words (terms) in the vocabulary as rows, and all the documents as columns
- Each cell contains the # of times a particular word appears in the respective document (also referred to as term frequency)

Vector representation for the word "good"



	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Figure 6.2 The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

Vector representation for document "As You Like It"

Note that word order and relationships/ context are not taken into account

Intuition when comparing document vectors: documents that are similar tend to have similar words & word counts, & thus similar column vectors

Intuition when comparing word vectors: similar words have similar vectors because they occur in similar documents

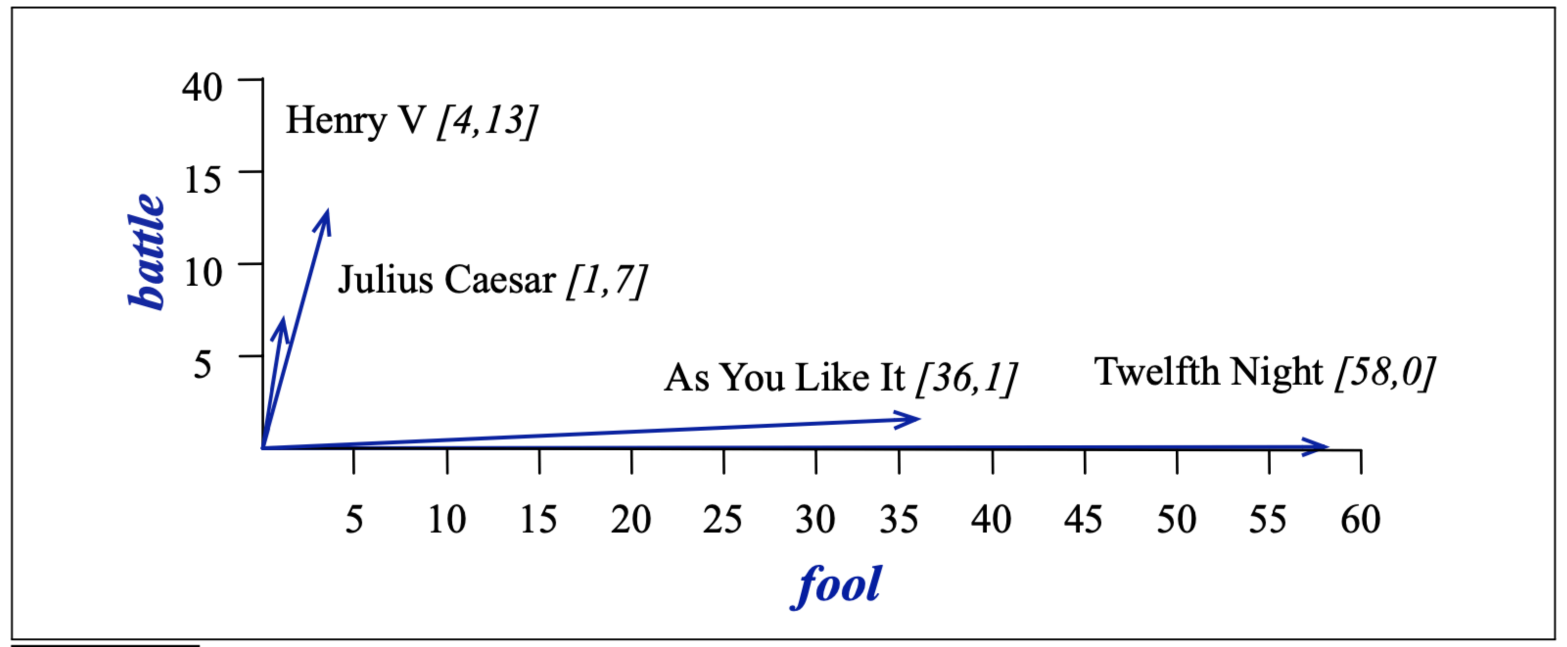


Figure 6.4 A spatial visualization of the document vectors for the four Shakespeare play documents, showing just two of the dimensions, corresponding to the words *battle* and *fool*. The comedies have high values for the *fool* dimension and low values for the *battle* dimension.

TERM-TERM MATRIX FOR GENERATING WORD EMBEDDINGS

- Skip-gram models: count words that are near each other but may have other words in between, e.g., how many times a pair of words is within distance k of each other in the document corpus

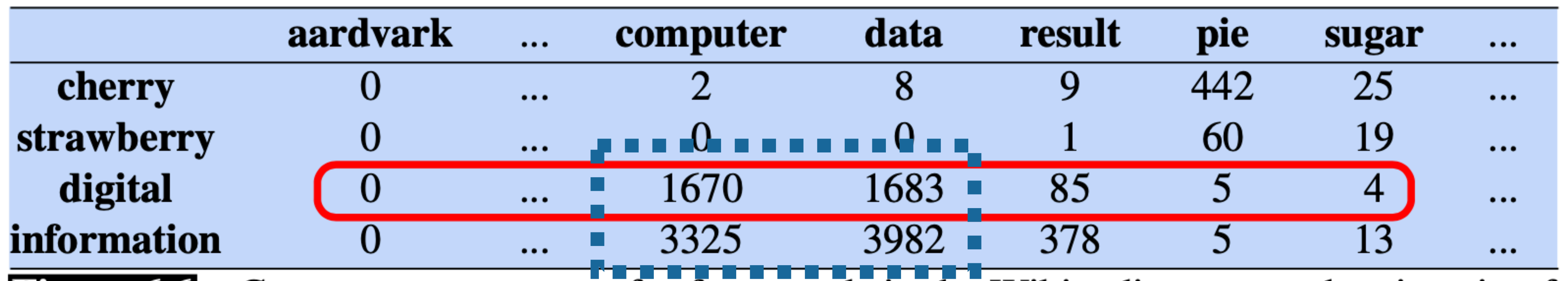
Example training corpus

- Document 1: “Kids like bread and butter”
- Document 2: “Babies like kids”

Co-occurrence matrix, using k=2

	Bread	Butter	Kids	Like	Babies
Bread	1	1	1	1	0
Butter	1	1	0	0	0
Kids	1	0	2	2	1
Like	1	0	2	2	1
Babies	0	0	1	1	1

TERM–TERM MATRIX FOR GENERATING WORD EMBEDDINGS (CONT.)



	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Figure 6.6 Co-occurrence vectors for four words in the Wikipedia corpus, showing six of the dimensions (hand-picked for pedagogical purposes). The vector for *digital* is outlined in red. Note that a real vector would have vastly more dimensions and thus be much sparser.

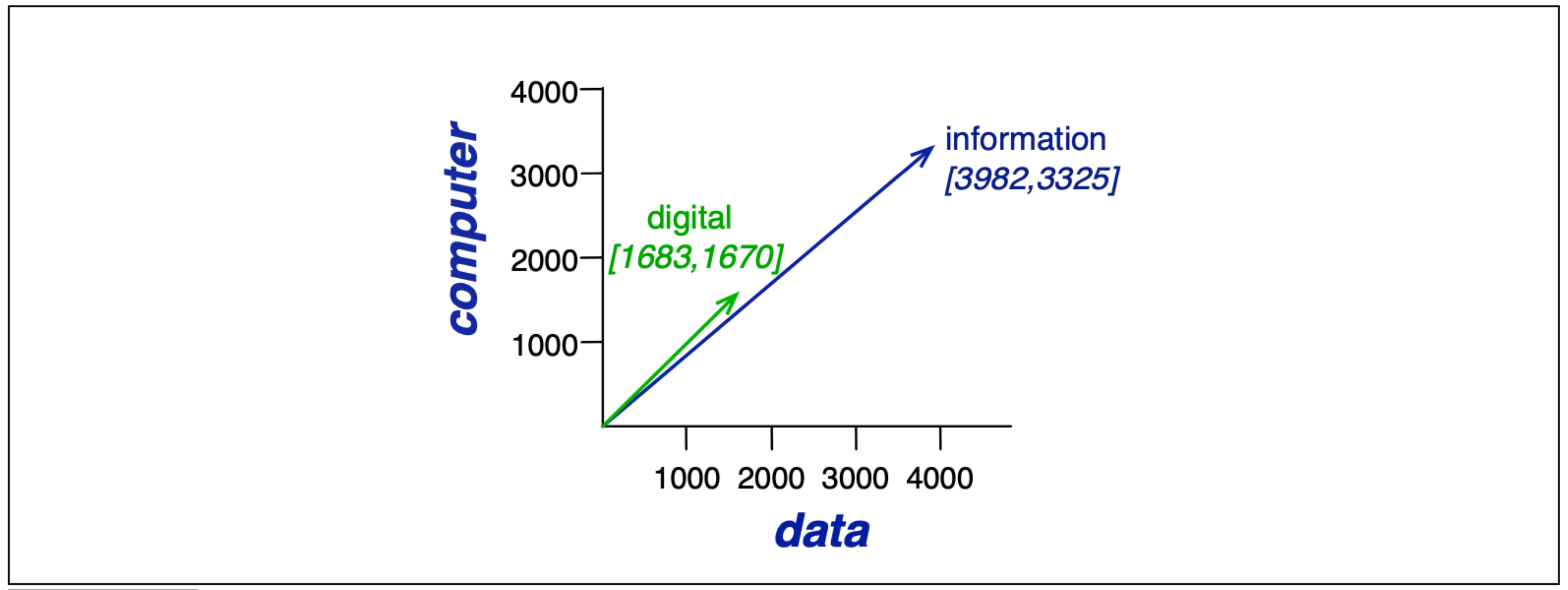


Figure 6.7 A spatial visualization of word vectors for *digital* and *information*, showing just two of the dimensions, corresponding to the words *data* and *computer*.

WEIGHTING WORD FREQUENCIES

- Motivation for introducing weights when considering word frequencies:
  - Very frequent words do not necessarily help differentiate between documents of different types (e.g., words like “the” or “good”)
  - We want to emphasize words that do help us differentiate between documents, e.g., “rocket” and “election” are words that more clearly indicate space vs. politics document
- Example approaches
  - Term Frequency-Inverse Document Frequency (TF-IDF) for word vectors extracted from term-document matrices
  - Positive Pointwise Mutual Information (PPMI) for word vectors extracted from term-term matrices

TF-IDF

- TF = Term Frequency:
  - The number of times a word (term t) appears in a given document (d): $tf_{t,d} = log_{10}(count(t, d) + 1)$
  - Or can be the number of times term appears divided by the number of words in the document
  - Or can be binary, e.g., the word appears or does not appear in a document
- IDF = Inverse Document Frequency: $idf_t = log_{10}\frac{N}{df_t}$
  - Number of documents in the corpus (N) divided by the number of documents in which the particular word appears (df_t)
  - Reflects how much information the word provides (how rare or common it is across the corpus)
- TF-IDF: $tf\_idf(t, d) = tf(t, d) \times idf(t, d)$

TF-IDF INTUITION

IDF goes up as the ratio inside the log goes up (i.e., as the number of documents in which a term appears goes down)

$$idf_t = log_{10}\frac{N}{df_t}$$

$$tf\_idf(t,d) = tf(t,d) \times idf(t,d)$$

10 docs out of 100 contain the word => log(100/10) = 1 (word is rare)

100 docs out of 100 contain the word => log(100/100) = 0 (word is common)

- A high weight in tf-idf is reached by a high term frequency in the given document (high tf value), and a low frequency of the term across the whole collection of documents (high idf value)
    - TF-IDF emphasizes rare words that are strong indicators of a particular document type (e.g., medical content ("patient") vs. space exploration content (e.g., "astronaut"))
    - And de-emphasizes common words that don't help differentiate between documents (e.g., the word "can" will not help us differentiate between medical and space exploration content)

Note: Often +1 is added to the denominator in the idf computation to avoid the tf-idf being 0 for words that appear in all docs; other formulations exist as well.

POINTWISE MUTUAL INFORMATION

- Pointwise Mutual Information (PMI) measures how often two events, x and y, co-occur relative to how often they would co-occur by chance (i.e., if they were independent)

$$I(x, y) = log_2 \frac{P(x, y)}{P(x)P(y)}$$

- In NLP, the events are word **w**, and context word **c**

$$\underset{\uparrow}{PMI}(w, c) = log_2 \frac{P(w, c)}{P(w)P(c)}$$

$P(w, c)$ ← How often we observe the two words together

$P(w)P(c)$ ← How often the two words are expected to co-occur if their occurrences are independent of each other

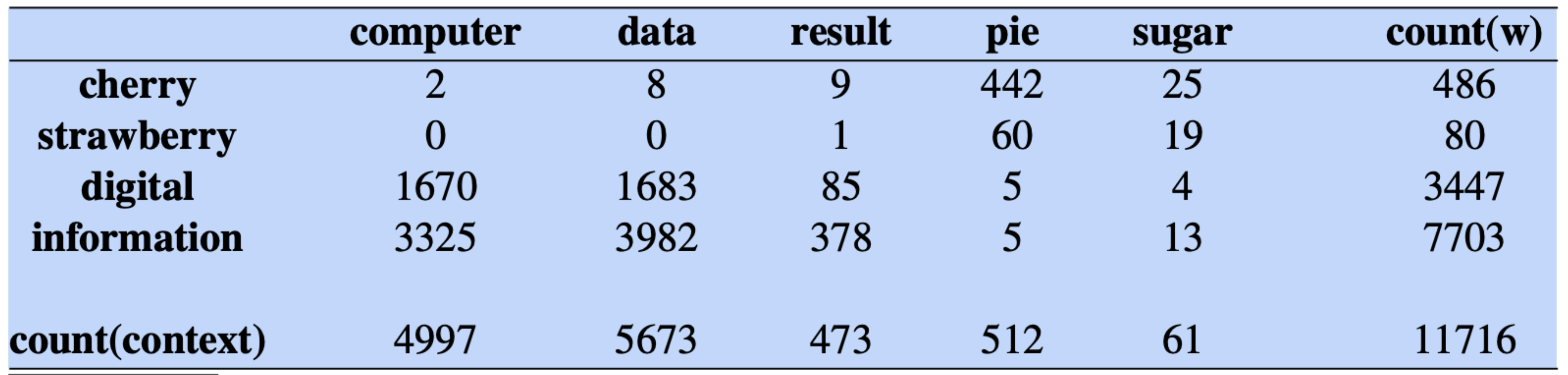
↑ How much more frequently the words actually co-occur compared to their expected co-occurrence if they only co-occurred by chance

- Typically we use only positive values for PMI (Positive PMI or PPMI)

$$PPMI(w, c) = max(log_2 \frac{P(w, c)}{P(w)P(c)}, 0)$$

PPMI EXAMPLE

$$PPMI(w,c) = max(log_2 \frac{P(w,c)}{P(w)P(c)}, 0)$$



	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

Figure 6.10 Co-occurrence counts for four words in 5 contexts in the Wikipedia corpus, together with the marginals, pretending for the purpose of this calculation that no other words/contexts matter.

Using co-occurrence counts in matrix format, we have:

$$P_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W} \sum_{j=1}^{C} f_{ij}}$$

i is the i-th word, and j is the j-th context word

W is the set of all words, C is the set of all context words

$$P(w = information, c = data) = \frac{3982}{11716} = 0.3399$$

$$P(w = information) = \frac{7703}{11716} = 0.6575$$

$$P(c = data) = \frac{5673}{11716} = 0.4842$$

$$PPMI(information, data) = log_2 \frac{0.3399}{0.6575 \times 0.4842} = 0.0944$$

USE OF WORD-BASED EMBEDDINGS FOR REPRESENTING DOCUMENTS

- How do we use the word vectors produced by tf-idf & PMMI to compare documents?
- For a document composed of k word vectors $w_1, w_2, \ldots, w_k$, we take the average of the word vectors to get the centroid document vector d

$$d = \frac{w_1 + w_2 \ldots + w_k}{k}$$

VECTOR SIMILARITY

SIMILARITY BETWEEN DOCUMENT VECTORS

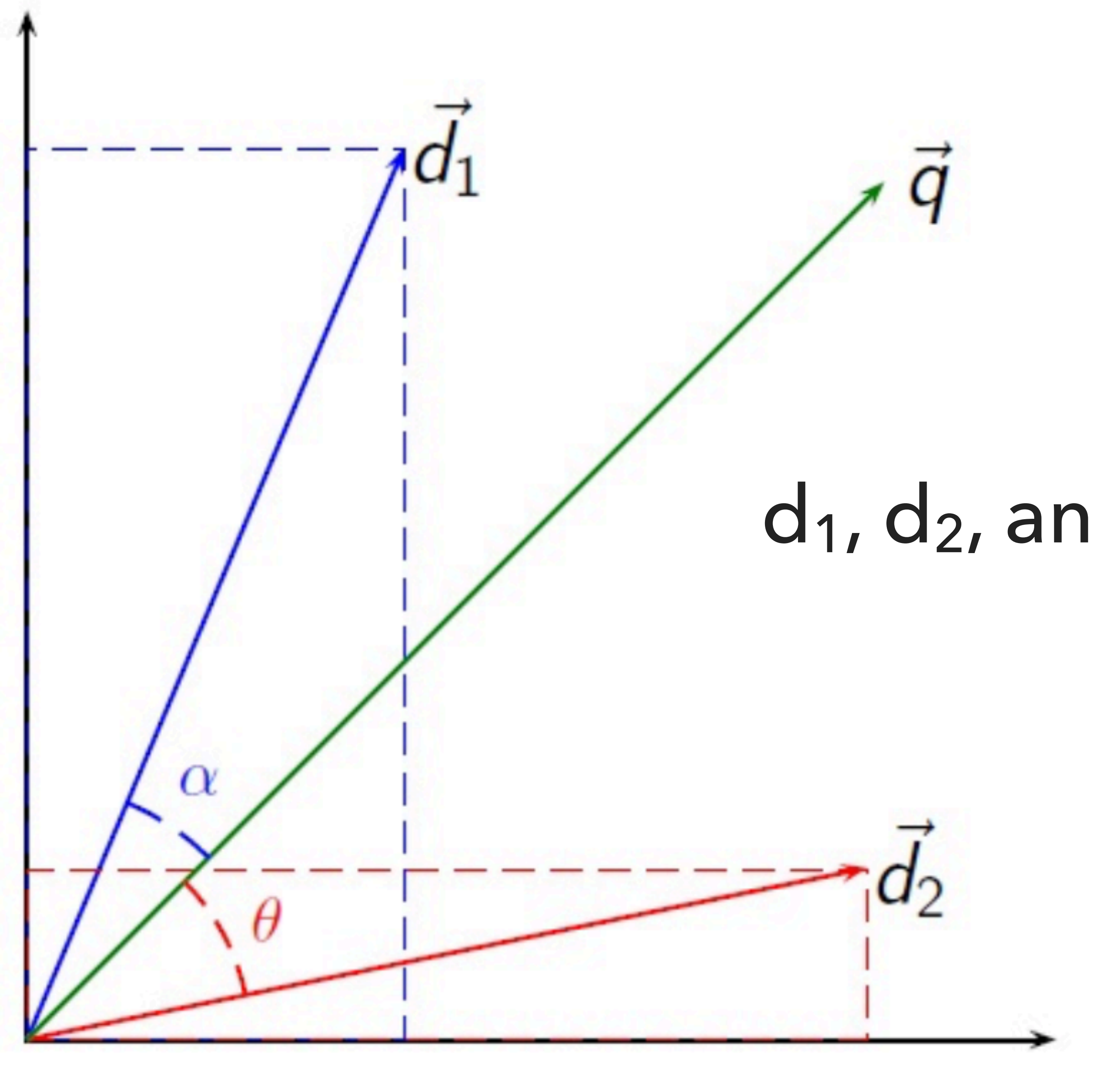
Cosine similarity is the most popular metric for computing document similarity

If d_1 and d_2 are two document vectors, then

$$\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2||\,,$$

where • indicates vector dot product and || *d* || is the length of vector *d*.

Cosine between two vectors is equivalent to normalizing them to unit length, and taking their inner product



https://en.wikipedia.org/wiki/Vector_space_model

d_1, d_2, and q are document vectors

Cosine between two vectors equals:

- 1 when they coincide (angle of 0)
- -1 when they coincide but point in opposite directions (angle of 180)
- 0 when they are orthogonal (angle of 90)

TOPIC MODELING

PROBABILISTIC TOPIC MODELS

- Topic modeling is an unsupervised method for clustering documents
- Enables us to get a general idea of the contents of a large document corpus
- We need to specify a parameter k for the number of "topics" in the dataset
    - k is an example of a hyperparameter (one specified by us, not learned by the model)

PROBABILISTIC TOPIC MODELS (CONT.)

- Topic modeling assumes that each document contains multiple latent/hidden ideas/themes/topics
- That each topic is characterized by a word distribution, which the algorithm finds
- Then documents can be classified

  based on the topics they contain

  which in turn are based on the word distributions contained in each document that match those topics

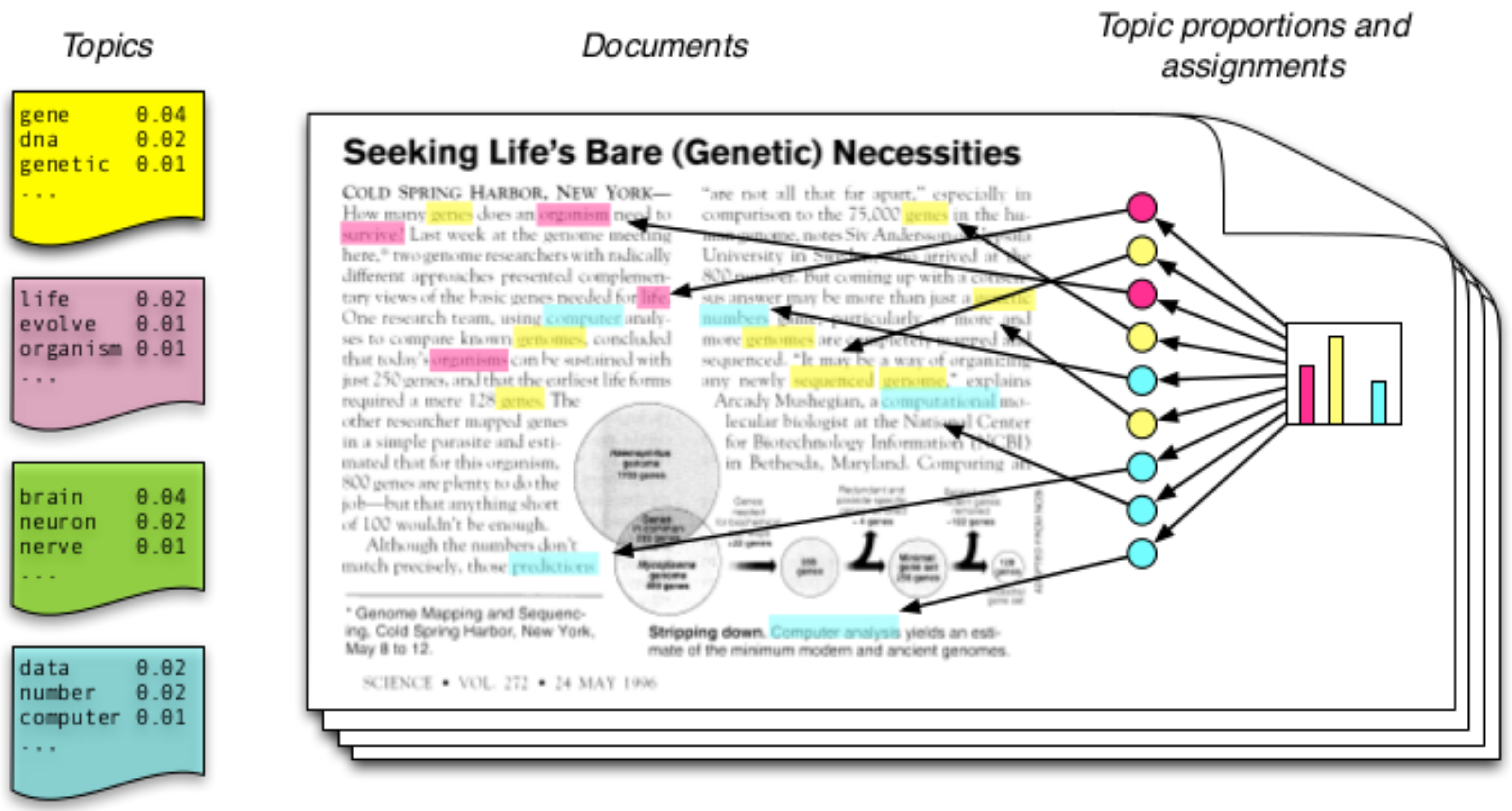
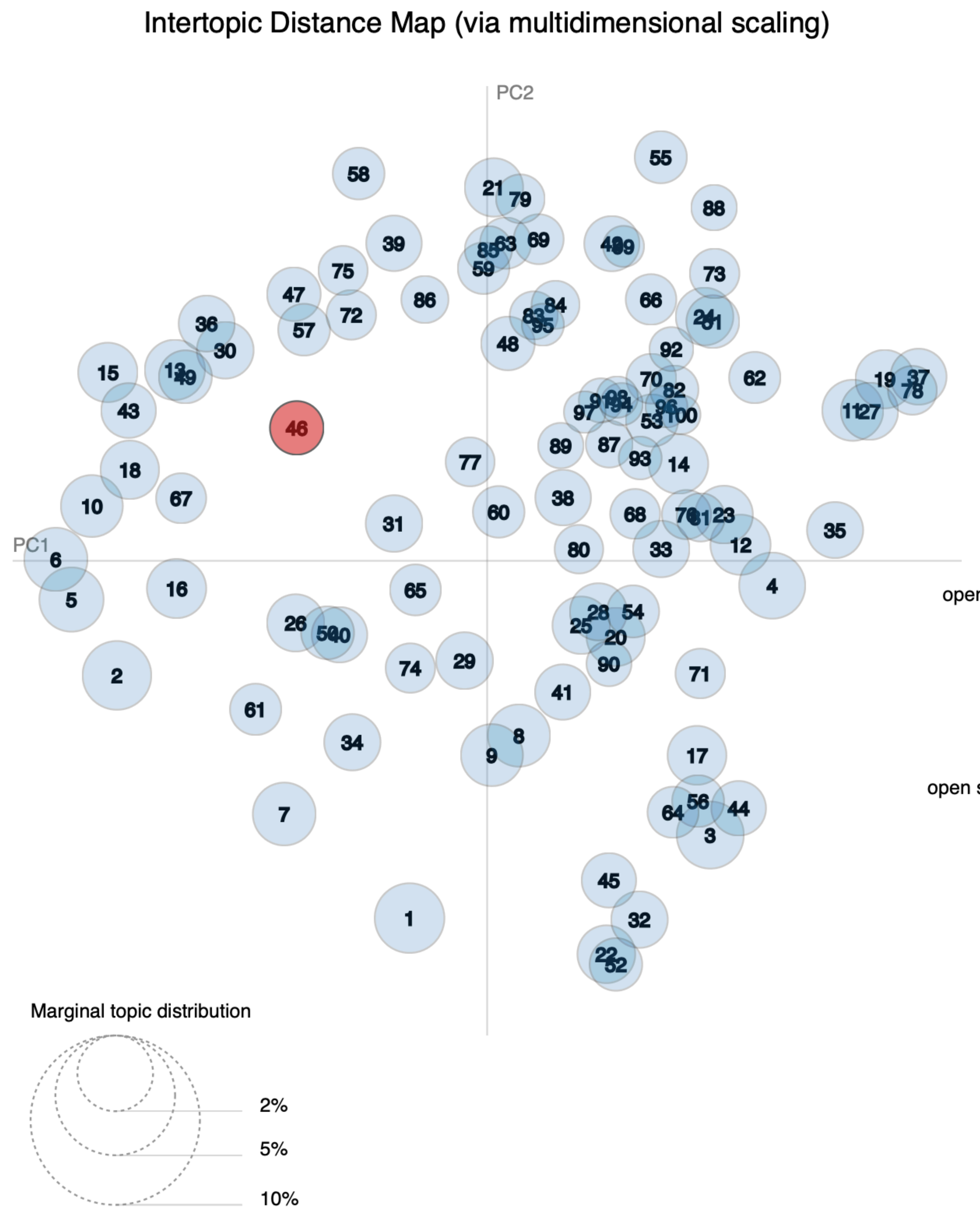
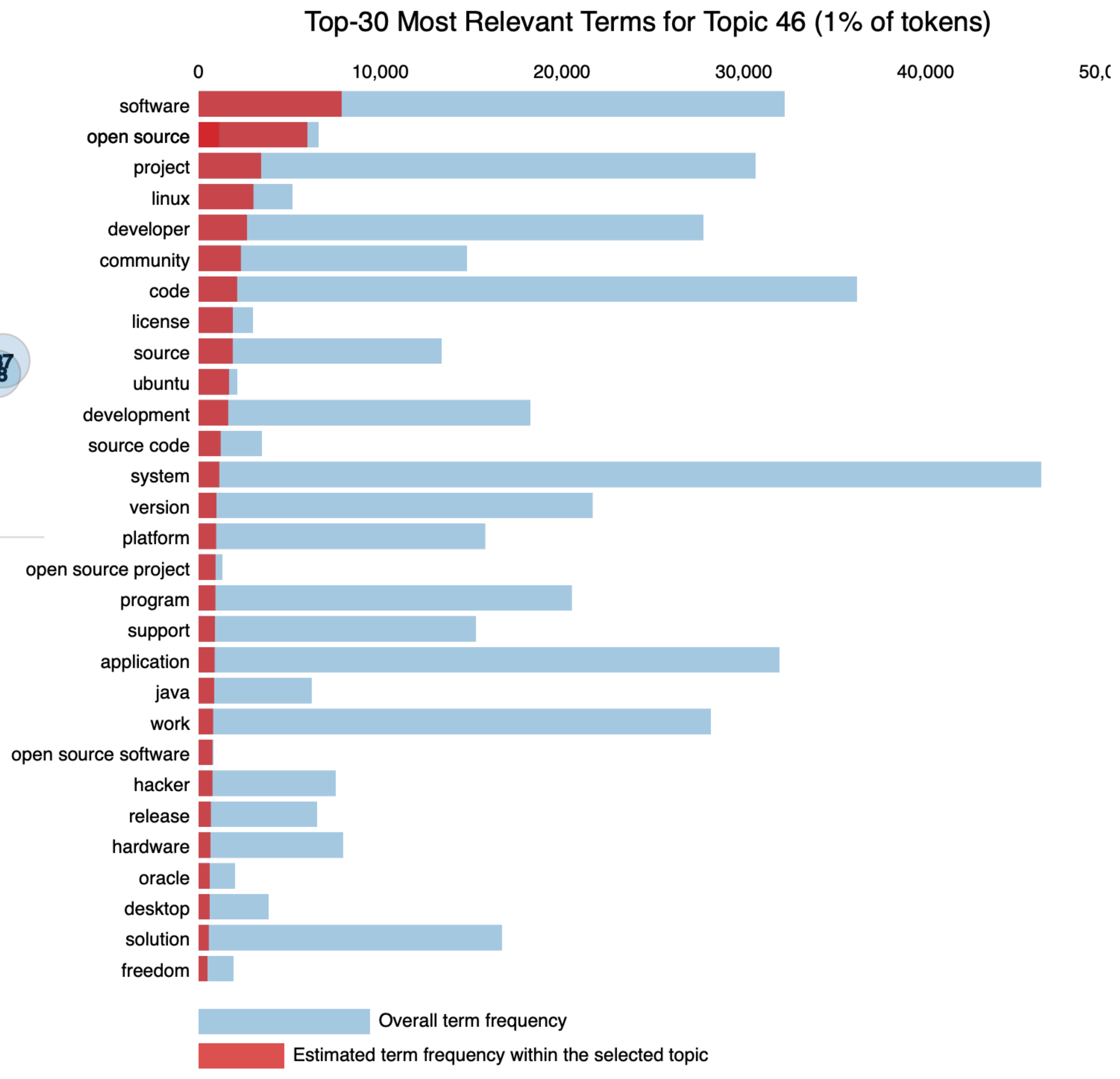


Fig. from Blei, David M., Probabilistic Topic Models presentation, Sept. 2, 2012.

TOPIC MODELING VISUALIZATION





1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

https://nbviewer.jupyter.org/github/bmabey/hacker_news_topic_modelling/blob/master/HN%20Topic%20Model%20Talk.ipynb#topic=46&lambda=1&term=