Chapter NLP:IV

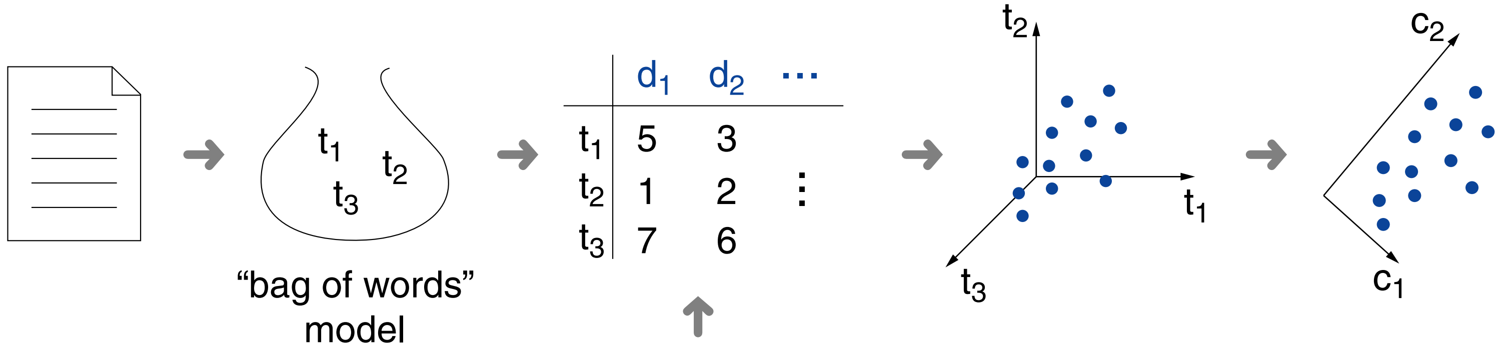
IV. Text Representation Models

- Introduction to Text Representation Models
- Bag of Words / Vector Space Model
- Similarity Measures in Natural Language Processing

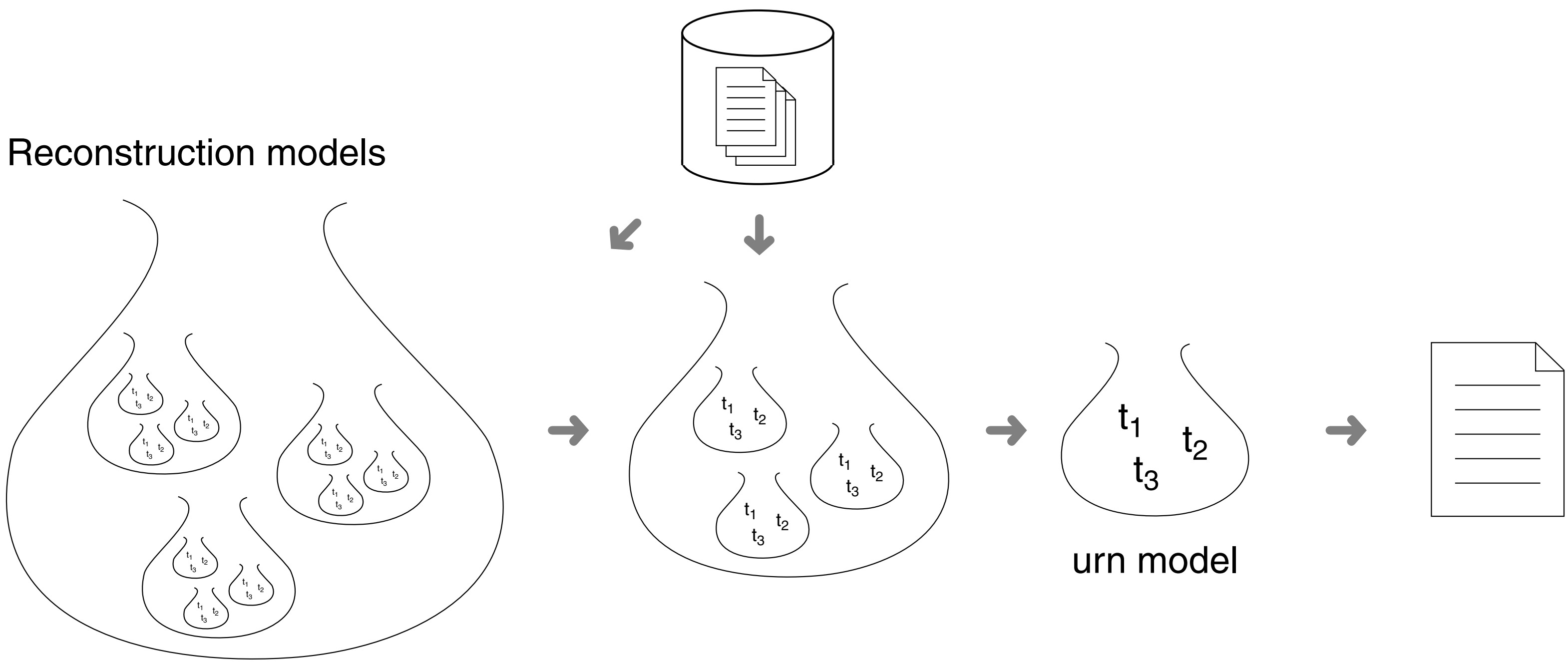
Introduction to Text Representation Models

How to represent Text? Digitally available texts

Deconstruction models



Reconstruction models



Introduction to Text Representation Models

Bag of words

Bag of words hypothesis: “The frequencies of words in a document tend to indicate the relevance of the document to a query” [Turney, Pantel 2010]

Bag metaphor

- frequency is important
- order can be neglected

Example word list from presidential speech: `shall(12), amendment(7), states(7), constitution (6), congress (4), united (4)`

Chapter NLP:IV

IV. Text Representation Models

- Introduction to Text Representation Models
- Bag of Words / Vector Space Model
- Similarity Measures in Natural Language Processing

Vector Space Model [Salton et. al. 1975]

How to represent Text with the Bag of Words assumption?

Idea: Encode textual

- documents in **vectors**
- corpora in **matrices**

Data = event counts for applications like machine learning and statistics

Example corpus:

- D_1: Kim is leaving home.
- D_2: Kim is at home.
- D_3: Karen is leaving.

Dimensionality of vector space

- $|D| \times |V|$ where $|D|$: Number of documents, $|V|$: Vocabulary
- Example matrix dimensions: 3×7

	Kim	is	leaving	home	.	at	Karen
D_1	1	1	1	1	1	0	0
D_2	1	1	0	1	1	1	0
D_3	0	1	1	0	1	0	1

Vector Space Model [Salton et. al. 1975]

Document-Term-Matrix

$$
|D| \begin{pmatrix} 1 & 0 & 5 & 1 & \cdots \\ 0 & 0 & 0 & 2 & \\ 1 & 3 & 0 & 1 & \\ 0 & 0 & 0 & 0 & \\ \vdots & & & & \ddots \end{pmatrix}
$$

$$|V|$$

- DTM’s may get very large
- Events: Frequency counts of word types in each document
- 100% Bag-of-Words
- Very sparse (contains approx. 95% zeros)
- variations:
  - Binary event counts
  - paragraphs as documents
  - sentences as documents
  - additional n-grams ($n > 1$) as events
  - . . .

Vector Space Model [Salton et. al. 1975]

Special case for encoding sequences of text

One Hot coding: A finite sequence of binary numbers where only one number gets the high value (1) and the others are low (0).

- In case of text we have a sequence $\delta_{i,j} = \begin{cases} 1, & \text{if } w_j = w_i \\ 0, & \text{else} \end{cases}$
- where i is the word position, j is the index in the vocabulary.
- Example:

	Kim	is	leaving	home	.	at	Karen
D_1.1	1	0	0	0	0	0	0
D_1.2	0	1	0	0	0	0	0
D_...	...	...	...	...	...	...	...
D_1.5	0	0	0	0	1	0	0

- The Binary encoding for a single word has the dimensionality of the vocabulary.

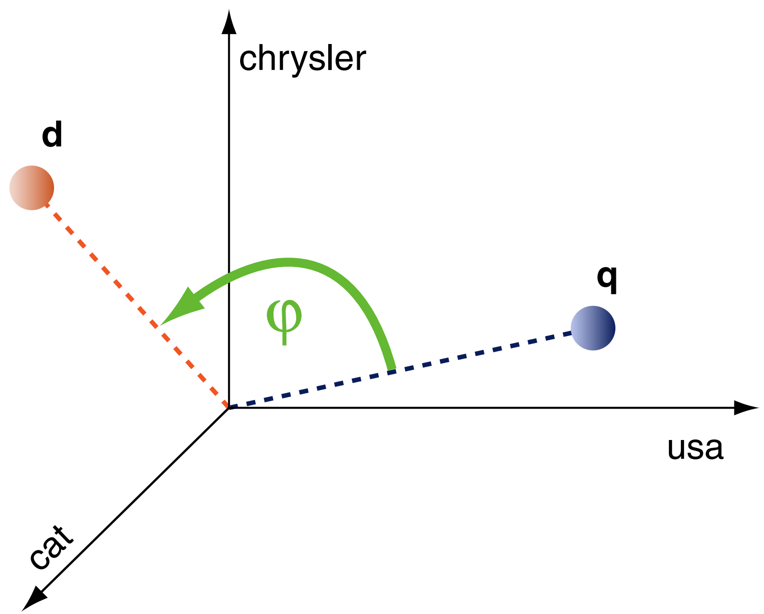
Vector Space Model

Document representations $\mathbf{D}$.

The set of index terms $T = \{t_1, \ldots, t_m\}$ is typically composed of the word stems of the vocabulary of a document collection, excluding stop words.

The representation $\mathbf{d}$ of a document d is a $|T|$-dimensional vector, where the i-th vector component of $\mathbf{d}$ corresponds to a term weight w_i of term $t_i \in T$, indicating its importance for d. Various term weighting schemes have been proposed.

Example Similarity Function ρ for a document: Cosine Similarity



Vector Space Model

Relevance Function ρ: Cosine Similarity

The scalar product $\mathbf{a}^T\mathbf{b}$ between two m-dimensional vectors $\mathbf{a}$ and $\mathbf{b}$, where φ denotes the angle between them, is defined as follows:

$$\begin{aligned}\mathbf{a}^T\mathbf{b} &= ||\mathbf{a}|| \cdot ||\mathbf{b}|| \cdot \cos(\varphi) \\ \Leftrightarrow \ \cos(\varphi) &= \frac{\mathbf{a}^T\mathbf{b}}{||\mathbf{a}|| \cdot ||\mathbf{b}||},\end{aligned}$$

where $||\mathbf{x}||$ denotes the L2 norm of vector $\mathbf{x}$:

$$||\mathbf{x}|| = \left(\sum_{i=1}^{n} x_i^2\right)^{1/2}$$

Let $\rho(\mathbf{q}, \mathbf{d}) = \cos(\varphi)$ be the relevance function of the vector space model.

Vector Space Model

Example

$$\mathbf{d} = \begin{pmatrix} \texttt{chrysler} & w_1 \\ \texttt{usa} & w_2 \\ \texttt{cat} & w_3 \\ \texttt{dog} & w_4 \\ \texttt{mouse} & w_5 \end{pmatrix} = \begin{pmatrix} \texttt{chrysler} & 1 \\ \texttt{usa} & 4 \\ \texttt{cat} & 3 \\ \texttt{dog} & 7 \\ \texttt{mouse} & 5 \end{pmatrix}$$

$$\mathbf{d_i} = \begin{pmatrix} \texttt{chrysler} & 0.1 \\ \texttt{usa} & 0.4 \\ \texttt{cat} & 0.3 \\ \texttt{dog} & 0.7 \\ \texttt{mouse} & 0.5 \end{pmatrix}, \quad \mathbf{d_j} = \begin{pmatrix} \texttt{chrysler} & 0.2 \\ \texttt{usa} & 0.1 \\ \texttt{cat} & 0.5 \\ \texttt{dog} & 0.0 \\ \texttt{mouse} & 0.0 \end{pmatrix}$$

The angle φ between $\mathbf{d_i}$ and $\mathbf{d_j}$ is about $67°$, $\cos(\varphi) \approx 0.38$.

Vector Space Model

Example Term Weighting: *tf* · *idf*

To compute the weight w for a term t from document d under the vector space model, the most commonly employed term weighting scheme $\omega(t)$ is *tf* · *idf*:

- *tf*(t, d) denotes the normalized *term frequency* of term t in document d.
  The basic idea is that the importance of term t is proportional to its frequency in document d. However, t's importance does not increase linearly: the raw frequency must be normalized.

- *df*(t, D) denotes the *document frequency* of term t in document collection D.
  It counts the number of documents that contain t at least once.

- *idf*(t, D) denotes the *inverse document frequency*:

$$\mathit{idf}(t, D) = \log \frac{|D|}{\mathit{df}(t, D)}$$

  The importance of term t in general is inversely proportional to its document frequency.

A term weight w for term t in document $d \in D$ is computed as follows:

$$\omega(t) = \mathit{tf}(t, d) \cdot \mathit{idf}(t, D).$$

Vector Space Model

Example Term Weighting: *tf* $\cdot$ *idf*

To compute the weight w for a term t from document d under the vector space model, the most commonly employed term weighting scheme $\omega(t)$ is *tf* $\cdot$ *idf*:

- *tf* (t, d) denotes the normalized *term frequency* of term t in document d.
  The basic idea is that the importance of term t is proportional to its frequency in document d. However, t's importance does not increase linearly: the raw frequency must be normalized.

- *df* (t, D) denotes the *document frequency* of term t in document collection D.
  It counts the number of documents that contain t at least once.

- *idf* (t, D) denotes the *inverse document frequency*:

$$\mathit{idf}(t, D) = \log \frac{|D|}{\mathit{df}(t, D)}$$

  The importance of term t in general is inversely proportional to its document frequency.

A term weight w for term t in document $d \in D$ is computed as follows:

$$\omega(t) = \mathit{tf}(t, d) \cdot \mathit{idf}(t, D).$$

Vector Space Model

Example Term Weighting: *tf* $\cdot$ *idf*

To compute the weight w for a term t from document d under the vector space model, the most commonly employed term weighting scheme $\omega(t)$ is *tf* $\cdot$ *idf*:

- *tf* (t, d) denotes the normalized *term frequency* of term t in document d.
  The basic idea is that the importance of term t is proportional to its frequency in document d. However, t's importance does not increase linearly: the raw frequency must be normalized.

- *df* (t, D) denotes the *document frequency* of term t in document collection D.
  It counts the number of documents that contain t at least once.

- *idf* (t, D) denotes the *inverse document frequency*:

$$\mathit{idf}(t, D) = \log \frac{|D|}{\mathit{df}(t, D)}$$

  The importance of term t in general is inversely proportional to its document frequency.

A term weight w for term t in document $d \in D$ is computed as follows:

$$\omega(t) = \mathit{tf}(t, d) \cdot \mathit{idf}(t, D).$$

Vector Space Model

Example Term Weighting: *tf* · *idf*

To compute the weight w for a term t from document d under the vector space model, the most commonly employed term weighting scheme $\omega(t)$ is *tf* · *idf*:

- *tf* (t, d) denotes the normalized *term frequency* of term t in document d.
  The basic idea is that the importance of term t is proportional to its frequency in document d. However, t's importance does not increase linearly: the raw frequency must be normalized.

- *df* (t, D) denotes the *document frequency* of term t in document collection D.
  It counts the number of documents that contain t at least once.

- *idf* (t, D) denotes the *inverse document frequency*:

$$\mathit{idf}(t, D) = \log \frac{|D|}{\mathit{df}(t, D)}$$

  The importance of term t in general is inversely proportional to its document frequency.

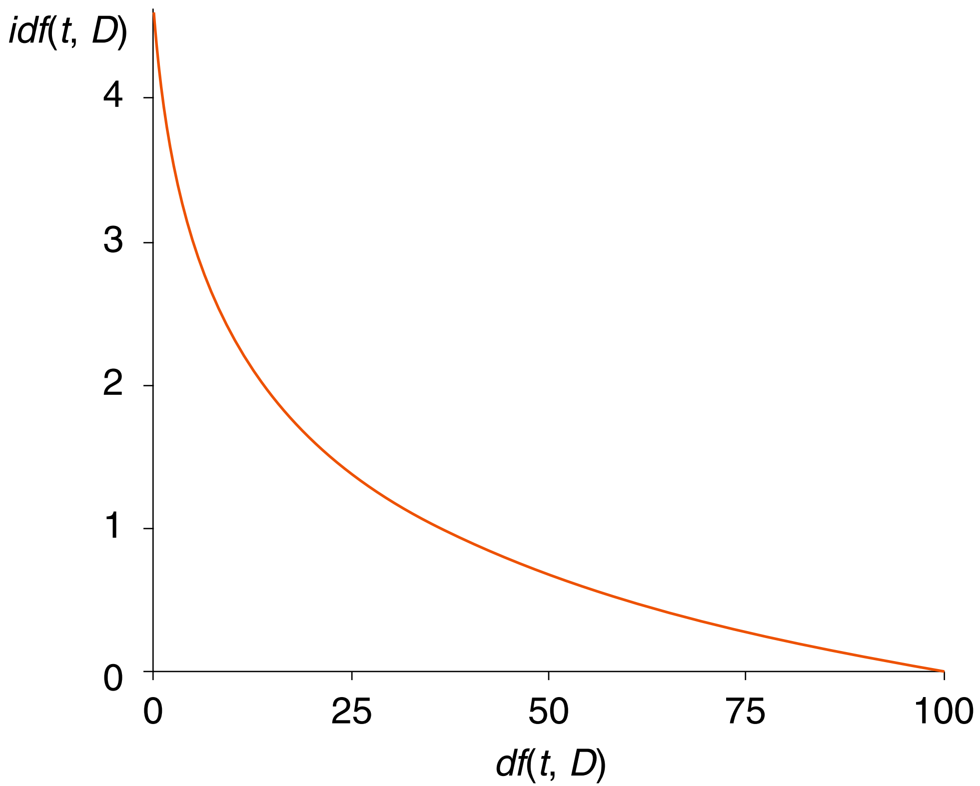
A term weight w for term t in document $d \in D$ is computed as follows:

$$\omega(t) = \mathit{tf}(t, d) \cdot \mathit{idf}(t, D).$$

Vector Space Model

Term Weighting: *tf* · *idf*

Plot of the function $\textit{idf}(t, D) = \log \dfrac{|D|}{\textit{df}(t, D)}$ for $|D| = 100$.



Remarks:

- Term frequency weighting was invented by Hans Peter Luhn: “There is also the probability that the more frequently a notion and combination of notions occur, the more importance the author attaches to them as reflecting the essence of his overall idea.” [Luhn 1957]

- The importance of a term t for a document d is not linearly correlated with its frequency. Several normalization factors have been proposed [Wikipedia]:

  – $\mathit{tf}(t,d)/|d|$

  – $1+\log(\mathit{tf}(t,d))$ for $\mathit{tf}(t,d)>0$

  – $k+(1-k)\dfrac{\mathit{tf}(t,d)}{\max_{t'\in d}(\mathit{tf}(t',d))}$, where k serves as smoothing term; typically $k=0.4$

- Inverse document frequency weighting was invented by Karen Spärck Jones: “it seems we should treat matches on non-frequent terms as more valuable than ones on frequent terms, without disregarding the latter altogether. The natural solution is to correlate a term’s matching value with its collection frequency.” [Spärck Jones 1972]

- Spärck Jones gives little theoretical justification for her intuition. Given the success of *idf* in practice, over the decades, numerous attempts at a theoretical justification have been made. A comprehensive overview has been compiled by [Robertson 2004].

Vector Space Model

Pruning of vocabulary in Vector Space Model

Vocabularies even of small collections can get very large (5.000 German newspaper documents $\rightarrow |V|$ > 300.000 types)

- performance issue in machine learning tasks
- meaningful semantics

Pruning = filtering the vocabulary of a collection by minimum / maximum thresholds of token occurrence

Very useful preprocessing step to reduce vocabulary size:

- Count occurrence of types in the complete corpus
- keep only those terms which occur above / below a well-defined threshold

Term Frequency

- absolute pruning
- sum all term occurrences in all documents filter terms which occur e.g. $count(term) > 1$ AND $count(term) < 1000$

Document Frequency

- relative pruning
- for each term count number of documents in which it is contained allows for filters like: terms which occur e.g. in
  - more than 99% OR
  - less than 1%

  of all documents

Remarks:

- Linguistic Preprocessing shall reduce / unify data for application specific purpose (See slides about text preprocessing)
- May contain various steps in row:
  - Data cleaning: encoding, spelling correction, duplicate filtering
  - Removing uninformative data: noise, duplicates, stopwords, dictionary lists
  - Unification: punctuation, capitalization, stemming, lemmatization
  - Pruning: remove low/high frequent terms
- Best setup dependent on final analysis task requirements
  - to be derived experimentally
  - or by analyst experience
- **Caution: order of steps influences results!**