Introduction to Natural Language Processing

The Bag-of-words approach

Dr. Manuel Pita

DEISI | lusófona informática

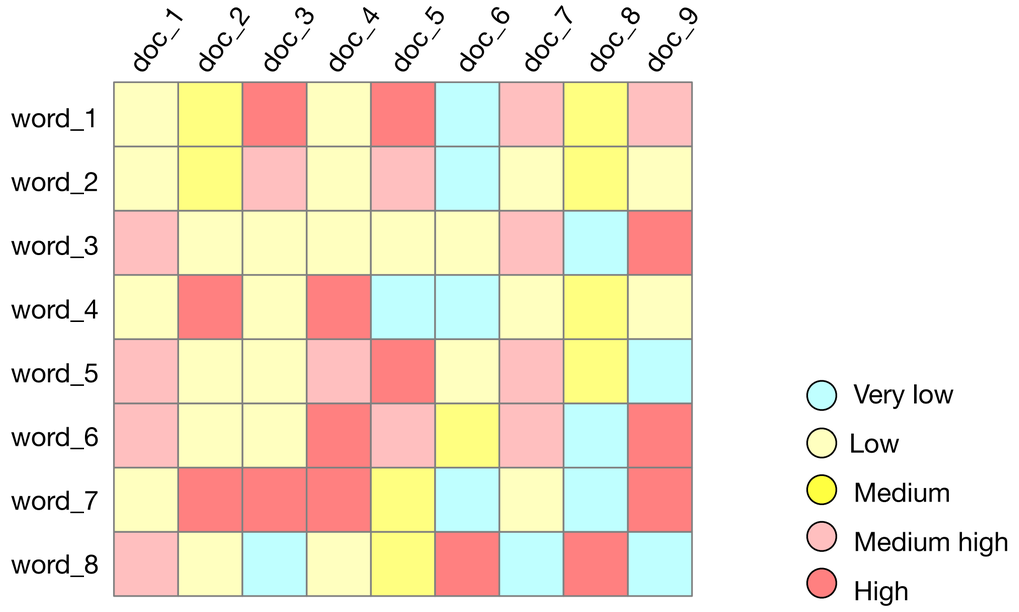
The notion of corpus

And its computational representation

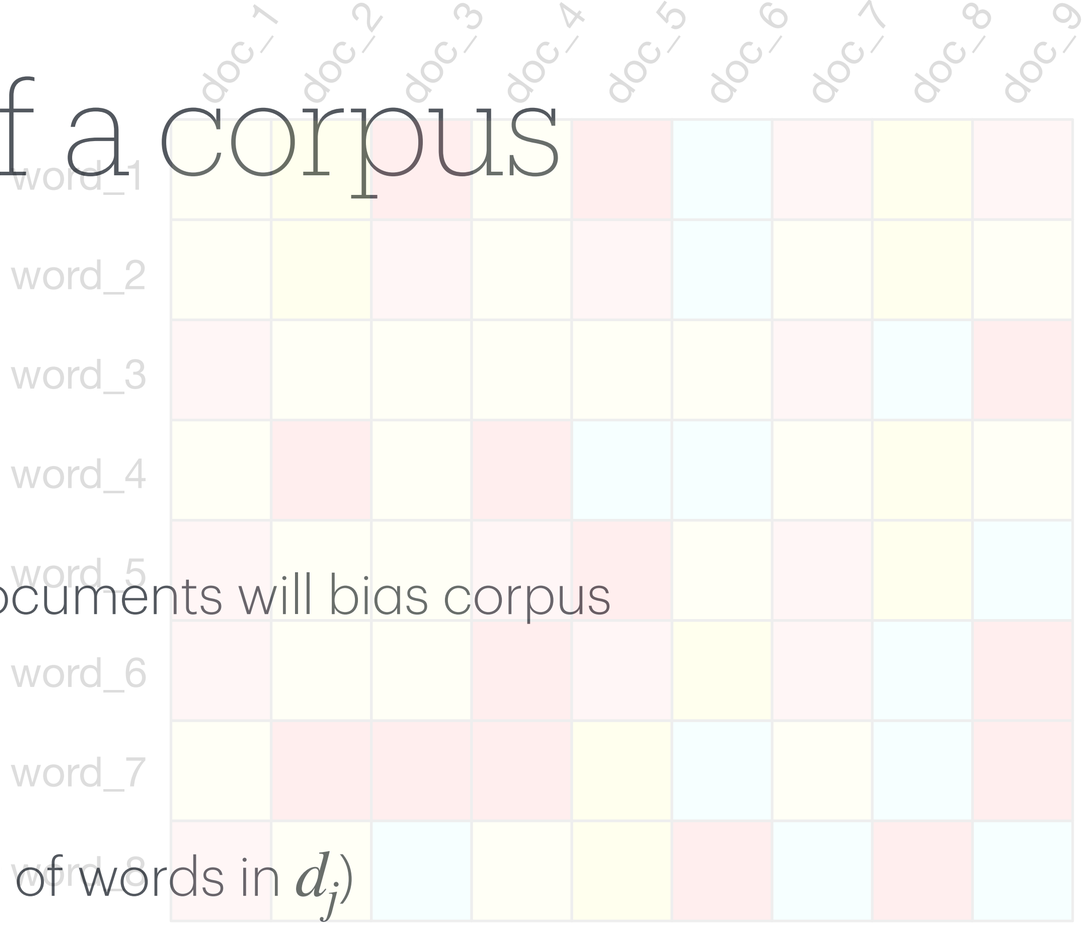
- Coherent **collection of documents** (e.g. recipes, encyclopaedias, news, etc.)
- If we enter a few **keywords**, what is the **most relevant document**?
- This is at the core of e.g. Google search
- One way to solve the problem is **represent documents as vectors over vocabularies**
- This **breaks grammar relationships**, reducing information to **word frequencies**
- This is known as the **Bag of Words approach**

Matrix representation of a corpus

- Each column is a document, d_j
- Rows are the words, w_i in the corpus **vocabulary**
- Matrix cell quantifies w_i importance in d_j
- **But what quantity captures a good notion of importance?**

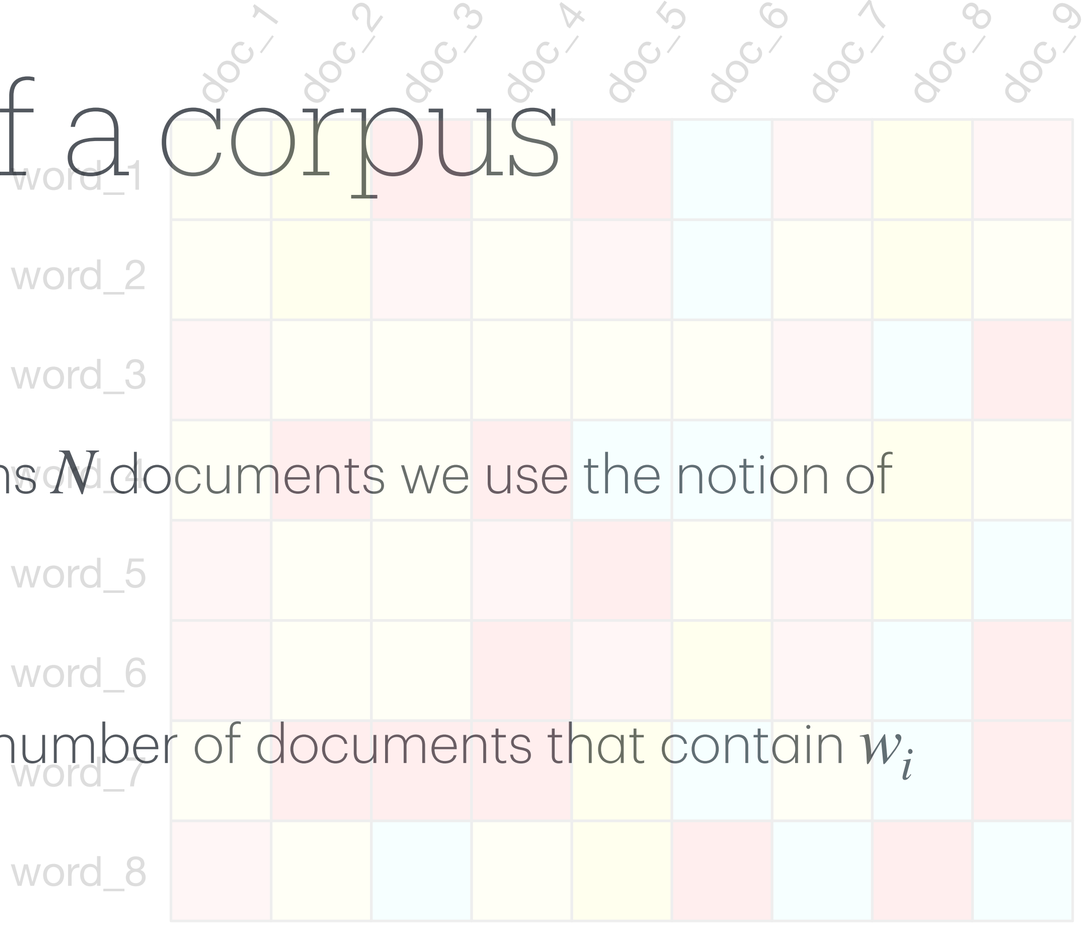


Matrix representation of a corpus



- **Word counts?** Yes but not enough
- **Problem 1:** documents have different sizes; bigger documents will bias corpus
- **Solution 1:** normalise word counts by document size
- $TF(w_i, d_j) = \text{count}(w_i, d_j)/|d_j|$ ($|d_j|$ is total number of words in d_j)
- **Problem 2:** TF captures relevance of word in document, but not in corpus
- **Example:** A high TF word that appears everywhere in the corpus is not informative

Matrix representation of a corpus



- To quantify importance of w_i in corpus C that contains N documents we use the notion of **Inverse Document Frequency** (IDF)

  $IDF(w_i, C) = log\left(\frac{N}{df(w_i)}\right)$, where $df(w_i)$ means number of documents that contain w_i

- How do we interpret this quantity?
- Low values correspond to very common words, high values to very rare words
- So, what we put in our matrix is $a_{i,j} = TF(w_i, d_j) \times IDF(w_i)$

Basic corpus pre-processing

1. **Instantiate the Corpus:** a collection of strings, each corresponding to a document
2. **Preprocess corpus part 1**: remove unwanted characters (e.g. punctuation), make the text lower case, map accented chars to non-accented (see notebook)
3. **Tokenise corpus**: all string docs become lists of tokens (words)
4. **Preprocess corpus part 2**: remove stop words, lemmatise
5. **Extract corpus vocabulary and compute IDF for each term** (token)

Simple example

Compute the TF matrix (ignore IDF for now)

Doc1: {tomate,cebola, alho}
Doc2: {manteiga, azeite, sal, pimenta}

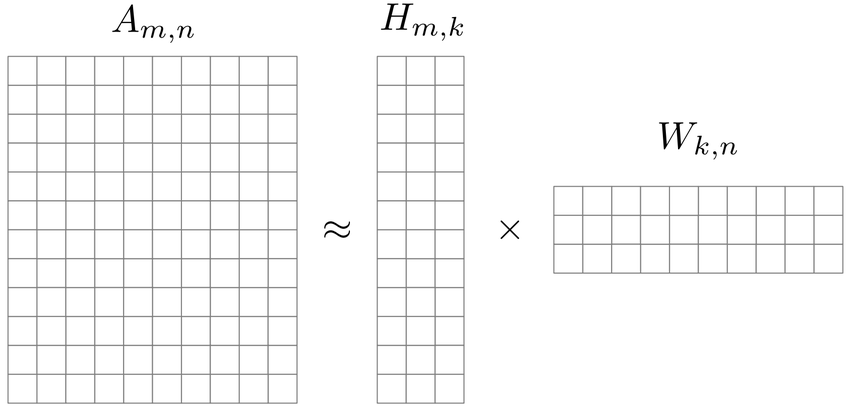
	Doc 1	Doc2
Tomate	1/3	0
Cebola	1/3	0
Alho	1/3	0
Manteiga	0	1/4
Azeite	0	1/4
Sal	0	1/4
Pimenta	0	1/4

The TF-IDF matrix

1. **Unified corpus representation:** documents represented in the same vocabulary vector
2. **Basic semantic inference becomes possible**: using e.g. cosine similarity (angle)
3. **Word-level representation** as matrix rows
4. **TF-IDF matrix can be factorised** $A \approx H \times W$
5. **Factorisation is useful** if we can interpret (extract semantics) from H and W, and we can!

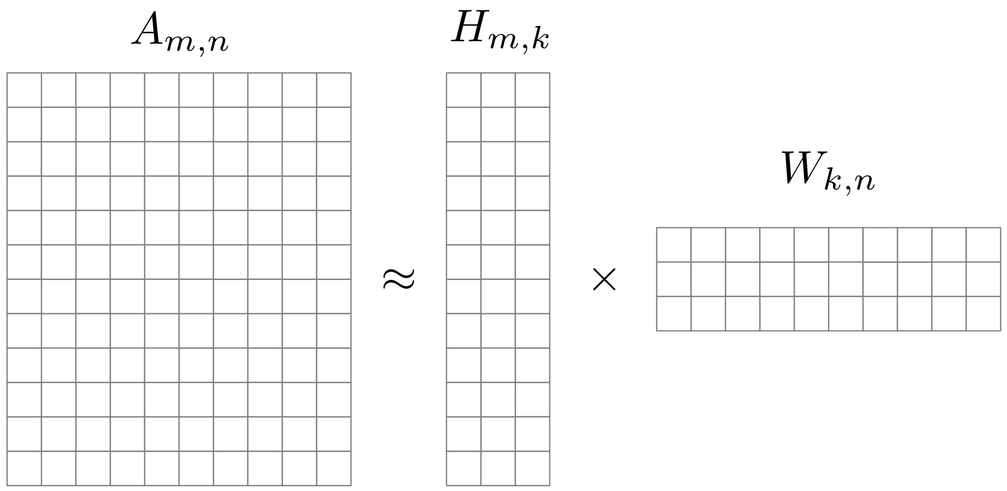
Topic modelling as matrix factorisation

1. **Start with matrix A (TF-IDF)**
2. **Use some matrix factorisation method**. For topic models we use the Latent Dirichlet Allocation (LDA)
3. LDA factorises a matrix with **special statistical rules**.
4. These rules are a good match for **how words are naturally distributed over topics**
5. By **distribution we mean term frequencies in topics**, e.g. onion appears more often in recipes topic than in general conversation, and this "more often" has statistical properties LDA matches.
6. LDA requires we say in advance the number of topics (k)



$$A_{m,n} \approx H_{m,k} \times W_{k,n}$$

Topic modelling as matrix factorisation



$$A_{m,n} \approx H_{m,k} \times W_{k,n}$$

1. **Start with matrix A (TF-IDF)**
2. **H is the loadings matrix:** how important are words in topic
3. **W is the scores matrix:** what proportion of each topic is in a document
4. We must be able to interpret these numbers correctly

Topic model interpretation

Through and example

	T1	T2	T3
milk	0.3	0	0
flour	0.3	0	0
lentils	0	0	0.4
tomato	0	0.6	0.4
beef	0	0.6	0
mozzarella	0.3	0	0

X

D1	D2	D3	D4	
0.6	0	0	…	T1
0.4	0	0.3	…	T2
0	1	0.7	…	T3

Topic T1 has high loadings for for milk, flour, mozzarella, T2 has high tomato and beef...

We can label topics considering the terms with high loadings. For example T1 is highly associated with “pizza”.

Document D1 is 60% T1 and 40% T2 (it could be pasta bolognaise, while document D2 is 100% T3 (maybe lentil tomato soup)