1 Introduction

1.1. **Define information retrieval**

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

1.2. **Define text mining and explain how it differs from information retrieval and data mining**

- Text mining (TM) is the discovery by computer of new, previously unknown information, by automatically extracting information from different written resources. A key element is the linking together of the extracted information to form new facts or new hypotheses to be explored further by more conventional means of experimentation.
- Different from IR because the goal is to discover unknown information, which could not have yet been written down
- Different from data mining because TM extracts patterns from natural language texts rather than from structured databases of facts

1.3. **Explain what is meant by "unstructured data" and provide an example**

- Data that does not have clear, semantically overt, easy-for-a-computer structure.
- Text does have structure: linguistic structure (syntax, discourse, etc.) and formatting structure (headings, paragraphs, etc.)

1.4. **List the main reasons why text analysis and retrieval (TAR) is challenging**

- Text is written in natural language. It is complex, ambiguous, vague, and relies on common sense knowledge. Full understanding of natural language is an AI-complete problem. Dealing with large amounts of data poses serious technical challenges.

1.5. **List the various types of language ambiguity and give an example for each**

- Categorical ambiguity: Flying planes can be dangerous. Time me on the last lap.
- Word sense ambiguity: I saw her run to the bank. The thief was charged by the police and had to pay a fine.
- Structural ambiguity: I saw a boy on the hill with a telescope.
- Referential ambiguity: Ann and Lisa gave John and Mark some apples because they liked them. Lisa gave Ann a present and she said thanks.

1.6. **Explain what Natural Language Processing (NLP) is**

- Natural language processing (NLP) is a field of computer science, artificial intelligence, and linguistics concerned with the interactions between computers and human (natural) languages. As such, NLP is related to the area of human - computer interaction.

1.7. **Given a short description of a concrete NLP task, recognize what it is**

- Morphological segmentation, Part-of-speech tagging, Parsing, Word sense disambiguation, Coreference resolution, Named entity recognition, Machine translation

1.8. **Clarify the difference between rule-based and statistical NLP**

- Rule-based NLP (knowledge-based, symbolic, logic-based): state machines, formal rule systems, logic. Rule-based NLP is adequate for some tasks but providing exact and complete models is difficult
- Statistical NLP (non-symbolic): probability theory, statistics, machine learning. Statistical NLP addresses this by detecting patterns occurring in texts

---

2 Basics of Natural Language Processing

2.1. **Given several steps in a typical NLP pipeline, state the most meaningful order in which these should be performed**

- Typical NLP tools are: POS tagging (PT), lemmatization (L), sentence segmentation (SS), tokenization (T) and parsing (P)
- SS → T → PT → L → P ?

2.2. **Describe different approaches to sentence segmentation and tokenization**

- Sentence segmentation: finding boundaries of sentences in text. Often done heuristically, using regular expressions. Best performance with supervised machine learning models (prediction of full stop denotion)
- Tokenization: breaking a text up into tokens - words and other meaningful elements. Tokens are words, punctuation marks, and special characters. Rule-based (i.e., heuristic) vs. supervised approaches.

2.3. **Give at least one example of how morphological preprocessing helps in NLP/IR tasks**

- The query house should match against document talking about houses and maybe housing (but probably not about housewives)

2.4**. Explain what stemming and lemmatization are, clarify the difference between them, and provide an example scenario in which you would prefer one over the other**

- Stemming: reduction of word-forms to stems. adjustments → adjust, defensible → defens, revivals → reviv
- Lemmatization: Transformation of a word-form into a linguistically valid base form, called the lemma. nouns → singular nominative form, verbs → infinitive form, adjectives → singular, nominative, masculine, indefinitive, positive form
- Example for stemming over lemmatization: fast, efficient
- Example for lemmatization over stemming: stemming prone to overstemming and understemming errors, imprecise

2.5. **Summarize in your own words how Porter stemmer works**

- Porter stemmer is a popular suffix-stripping stemmer. Each word can be represented as $[C](VC)^m[V]$, where C is a sequence of consonants and V is a sequence of vowels. Each word has a measure m and by using suffix stripping rules it removes suffices in these steps: deals with plurals and past participles, derivation, tidying up.

2.6. **Describe what a language model is and what it is used for**

- Language model is a probabilistic model of text (sequence of words). It determines the probability of a word sequence and a likely candidate for the next word in a sequence.
- It's used for POS (Part-Of-Speech) tagging, parsing, semantics, IR

2.7. **Describe what a corpus is and what it is used for**

- Text corpus (plural: corpora): large and structured set of texts, used for corpus linguistic analyses and for the development of natural language models (primarily machine learning models)

2.8. **Describe Zipf's law and explain why it is relevant for language modeling**

- Zipf's law states that given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.
- Even short sequences of 5-6 words would barely ever appear in a large corpus (we need to approximate each full conditional).

2.9. **Describe what „part-of-speech" means and explain what POS-tagging is**

- Part-of-speech (POS) explains not what the word is, but how it is used. Universal parts across languages: verbs, nouns, adjectives, pronouns, adverbs, prepositions, …
- POS tagging (grammatical tagging, word-category disambiguation) is the process of marking up a word in a text as corresponding to a particular part of speech. POS taggers assign tags from a finite predefined tagset**.** State-of-the-art POS taggers are supervised machine learning models

2.10**. Describe what parsing is and clarify the difference between POS-tagging and parsing**

- Parsing is the task of analyzing the grammatical structure of natural language sentences.
- Given a sequence of words, a parser forms units like subject, verb, object and determines the relations between them according to some grammar formalism**.**

2.11. **Describe what WordNet is and give examples of the data it contains and how NLP can benefit from it**

- WordNet: Manually constructed lexical database, contains nouns, verbs, adjectives and adverbs. Words are organized into synsets - sets of words with the same sense.
- For each synset WordNet provides:
  - a list of words that can be used in that sense
  - a gloss - a short description of the sense
  - semantic relations to other synsets (hyponymy, meronymy, …)
  - Hyperonymy/hyponymy - IS-A relation (chair - furniture)
  - Meronymy - a part of whole relation (finger - hand)
  - Antonymy - opposite meaning (wet - dry)
  - Similarity - similar (but not identical) meaning (warm - hot)
    - Verbs:
  - Troponymy - increasingly specific manner of an event (communicate - talk - whisper, move - jog - run)
  - Entailment - one word entails the other (succeed - try, buy - pay)

2.12. **Clarify the difference between semantic relatedness and similarity**

- Similarity (S) = pragmatic relation
- Relatedness (R) = syntagmatic relation
- Examples: airplane - machine (S), airplane - engine (S), pilot - airplane (R), magic - disappear (R), rich - caviar (R)

2.13**. Explain what polysemy is, give examples, and name the NLP task that deals with polysemy**

- Polysemy - a word has multiple, related meanings, e.g. ring (wedding ring vs. boxing ring)
- Word sense disambiguation (WSD): the task of identifying which sense (meaning) of a word is used in a sentence, when the word has multiple meanings.

---

3 Basics of Information Retrieval

3.1. **Explain what information need is and provide an example**

- Information need: Information need is an individual or group's desire to locate and obtain information to satisfy a conscious or unconscious need.
- (Un)conscious needs for information are expressed via queries: words and phrases in text information retrieval (e.g., "ISIS attacks"), images in image content retrieval

3.2. **List the three types of text representation used for IR and identify the one most typically used in IR**

- Unstructured representation ← most used?
- Weakly-structured representations
- Structured representations ← not used for IR

3.3. **Discuss the pros and cons of the unstructured (bag-of-words) text representation for IR**

- Pros: text represented as an unordered set of terms = simpler?
- Cons: considerable oversimplification = ignoring syntax and semantics (despite oversimplifying, satisfiable retrieval performance)

3.4. **Given a snipped of text, transform it into a specified unstructured or weakly structured representation**

- Text is transformed into pairs word- count: One evening Frodo and … → {(One, 1), (evening, 1), (Frodo, 2), (and, 2),… }

3.5. **List the preprocessing steps commonly performed prior to document indexing in IR**

- Morphological normalization: stemming or lemmatization
- Removal of stop words: keeping just content words - nouns, verbs, adjectives, adverbs

3.6. **Name the three components of every IR system**

- A basic retrieval model is a triple (f_d, f_q, r) where:
    1. f_d is a function that maps documents to their representations for retrieval, i.e., $f_d(d) = p_d$, where p_d is the retrieval representation of the document d
    2. f_q is a function that maps queries to their representations for retrieval, i.e., $f_q(q) = s_q$, where s_q is the retrieval representation of the query q
    3. r is a ranking function

3.7. **Describe the purpose of an index (vocabulary) in IR**

- Index terms are all the terms in the collection (i.e., the vocabulary). Each term k_i is, for document d_j, assigned a weight ω_{ij}. Document d_j is represented by the term vector $[\omega_{1j}, \omega_{2j}, \dots, \omega_{tj}]$, where t is the number of index terms. Let g be the function that computes the weights, i.e., $\omega_{ij} = g(k_i, d_j)$. Different choices for the weight-computation function g and the ranking function r define different IR models

3.8. **List the three main IR paradigms**

- Set theoretic models (Boolean model)
- Algebraic models (Vector space model)
- Probabilistic models (Classic probabilistic model, Language model)

3.9. **Given a document collection, a vocabulary, and a query, apply Boolean retrieval and list the documents that would be retrieved as relevant**

- $L(\text{Frodo}) = \{d_1, d_2, d_3\}$, $L(\text{stab}) = \{d_1, d_2\}$
- $rel(D, q) = L(\text{Frodo}) \cap L(\text{stab}) = \{d_1, d_2, d_3\} \cap \{d_1, d_2\} = \{d_1, d_2\}$

3.10. **Illustrate the use of an inverted file index for Boolean retrieval**

- Inverted file index contains a list of references to documents for all index terms (e.g., $L(\text{Frodo}) = \{d_1, d_2, d_3\}$)

3.11. **Describe the advantages and disadvantages of the Boolean retrieval model**

- Pros: simplicity (computational efficiency)
- Cons: expressing information needs as Boolean expressions is unintuitive, pure model (no ranking - documents are either relevant or non-relevant, relative importance of indexed terms is ignored)

3.12**. Explain how the vector space retrieval model works**

- Documents and queries are represented as vectors of index terms. Weights are real numbers ≥ 0
- $d_j = [\omega_{1j}, \omega_{2j}, \dots, \omega_{tj}]$ – document
- $q = [\omega_{1q}, \omega_{2q}, \dots, \omega_{tq}]$ – query
- The relevance of the document for the query is estimated by computing some distance or similarity metric between the two vectors

3.13. **Name at least one distance and one similarity metric used in vector space retrieval**

- Distance metric: Euclidean, Manhattan
- Similarity metric: Cosine, Dice

3.14. **Explain the intuition behind the use of TF-IDF weighting and derive the TF-IDF weighed vector for a given example document**

- The weight computed as the product of the term frequency (TF) component and the inverse document frequency (IDF) component:
  - $\omega_{ij} = tf(k_i, d_j) \cdot idf(k_i, D)$
  - $tf(k_i, d_j) = 0.5 + \frac{0.5 \cdot freq(k_i, d_j)}{\max(freq(k, d_j) \mid k \in d_j)}$
  - $idf(k_i, D) = log \frac{|D|}{|\{d \in D \mid k_i \in d_j\}|}$

3.15. **Given a document collection, a vocabulary, and a query (along with any relevant equations you may need), apply the vector space model retrieval and list the documents by relevance**

- Weights are calculated for terms, and provided equation is used.

3.16. **Starting from the probability of relevance given a document and a query, derive the expression for ranking documents in probabilistic retrieval models (probability ranking principle)**

- Estimate: $\mathrm{P(R = 1 \mid D = d, Q = q)}$,
- D – Document, Q – Query, R – relevance judgment (1 if D relevant for Q, 0 otherwise)
  - Let r be a shorthand for $R = 1$, and $\bar{r}$ for $R = 0$
  - We apply the logit function to probability
    - $logit(p) = \log\left(\frac{p}{1-p}\right)$
  - This is a rank preserving transformation giving:
    - $log\frac{p(r|D,Q)}{1-p(r|D,Q)} = log\frac{p(r|D,Q)}{p(\bar{r}|D,Q)}$
  - Now we can apply the Bayes rule:
    - $log\frac{p(r|D,Q)}{p(\bar{r}|D,Q)} = log\frac{p(D,Q|r)p(r)}{p(D,Q|\bar{r})p(\bar{r})}$
    - A benefit of having used logit is that $P(D,Q)$ cancels out
  - Finally, we use the chain rule:
    $log\frac{p(D,Q|r)p(r)}{p(D,Q|\bar{r})p(\bar{r})} = log\frac{p(D|Q,r)p(Q|r)p(r)}{p(D|Q,\bar{r})p(Q|\bar{r})p(\bar{r})}$
    - $= log\frac{p(D|Q,r)}{p(D|Q,\bar{r})} + log\frac{p(Q|r)p(r)}{p(Q|\bar{r})p(\bar{r})} \propto log\frac{p(D|Q,r)}{p(D|Q,\bar{r})}$

3.17. **Explain the two assumptions underlying the binary independence model**

- Assumption 1: given relevance terms are statistically independent
- Assumption 2: The presence of a term in a document depends on relevance only when that term is present in the query.

3.18. **Explain the advantage of a two Poisson model over binary independence model, and the advantage of BM models over the two Poisson model**

- Two Poisson Model: A more realistic document representation, a vector of word frequencies
- BM: Removes document length assumptions of the two Poisson model. Matches in longer documents should be less important.

3.19. **Apply language modeling to a snippet of text, including a smoothed language model**

- Unigram language models:
  - Solve the sparseness problem by completely ignoring conditioning
  - Probability of sentence t under the unigram model is:
    - $p(t|M) = p(t_n) \cdots p(t_i) \cdots p(t_1)$
  - The probabilities that define the model are estimated from a collection:
    - $p(t_i) = \frac{n_i}{n_T}$
    - n_i - number of times term t_i occurs in the collection
    - n_T - total number of term occurrences in the collection
  - Example:

- d_1: “Frodo and Sam stabbed orcs.”
- d_2: “Sam chased the orc with the sword.”
- d_3: “Sam took the sword.”

t_i	Frodo	Sam	orc	chased	sword
$P(t_i)$	1/16	3/16	2/16	1/16	2/16

- Bigram language models:
  - We simplify the conditionals by leaving only the previous word
  - A better approximation of reality than unigram models
  - Probability of sentence t under the bigram model:
    - $p(t|M) = p(t_n|t_{n-1}) \cdots p(t_i|t_{i-1}) \cdots p(t_1)$
  - The probabilities that define the model are estimated from a collection:
    - $p(t_i|t_{i-1}) = \frac{n(t_{i-1},t_i)}{n(t_{i-1})}$
    - $n(t_{i-1}, t_i)$ - number of times bigram (t_{i-1}, t_i) occurs in the collection
    - $n(t_{i-1})$ – number of times term t_{i-1} occurs in the collection
  - Example (documents same as above):

t_{i-1}, t_i	Frodo, chased	the, sword	the, orc
$P(t_i|t_{i-1})$	0	2/3	1/3

    - Laplace smoothing: $p'(t_i|M_d) = \frac{n_{i,d}+\alpha}{n_d+|V|\alpha}$
    - Jelinek-Mercer smoothing: $p'^{(t_i|M_d)} = \lambda p(t_i|M_d) + (1-\lambda)p(t_i|M_D)$
    - Dirichlet smoothed unigram model: $p(t_i|M_d) = \frac{n_{i,d}+\mu P(t_i|M_D)}{n_d+\mu}$

3.20. **Clarify the difference between classic probabilistic retrieval and language modeling retrieval**

- Instead of modeling document probability given the query we model the query probability given the document.

3.21. **Explain how language modeling can be used for retrieval**

- Given a document collection D and a query q
- A language model M_d is built for each document
- Documents are scored according to the probability $P(q|M_d)$

3.22. **Given a set of documents, a vocabulary, and a query (along with any relevant equations you may need), apply the Query Likelihood Model (QLM) and rank the documents by their relevance**

- A unigram language model of document d can be estimated by dividing the number of times term i occurs in d by the number of terms in d
  - $P(t_i|M_d) = \frac{n_{i,d}}{n_d}$

3.23. **Explain the pitfalls of language modeling of short documents and explain one smoothing technique**

- Models we've considered so far give probability 0 to queries which contain terms that do not occur in the document
- BM11 (Best Matching):
  - Removes document length assumptions of the two Poisson model
    - matches in longer documents should be less important

- We can correct the frequency $f'_{t,d} = f_{t,d}\left(\frac{l_{avg}}{l_d}\right)$
  - l_{avg} - the average length of a document
  - l_d - the length of document d
  - dampens/boosts word frequencies based on above/below average document length
- Now we can rewrite (6) as:
  - $\sum_{t \in q} \frac{f_{t,d}(k_1+1)}{k_1\left(\frac{l_d}{l_{avg}}\right)+f_{t,d}} \cdot \omega_t$

- BM25:
  - While BM11 removes the problem with assuming equal document length in practice it has problems
    - Long relevant documents are getting too much dampening
    - Short irrelevant documents are getting too much boosting
  - To control the amount of correction we introduce b (often set to 0.75)
    - $\sum_{t \in q} \frac{f_{t,d}(k_1+1)}{\boldsymbol{k_1}(\mathbf{1}-\boldsymbol{b})+k_1\left(\frac{l_d}{l_{avg}}\right)\boldsymbol{b}+f_{t,d}} \cdot \omega_t$
    - This expression represents the famous BM25 ranking function, which gives state-of-the-art results

4 Improved Search, Evaluation and Web Search

4.1. **Explain what query expansion (QE) is**

- Query expansion techniques try to improve retrieval results by adding terms to the user query

4.2. **Explain the workings of QE, and briefly explain the four different ways it is most commonly done**

- The additional terms should be chosen to complement words already present in the query (e.g., synonyms or related words) - If the user searches for accommodation, the search engine should also retrieve the documents that contain only the word hotel
  - 1. a) Explicit query expansion: original query, expanded versions of the query are suggested to the user
  - 1. b) Implicit query expansion: original query is implicitly expanded
  - 2. Controlled vocabulary: terms from the user query are mapped to canonical terms ({hotel, room, apartment} → accommodation)
  - 3. a) Manual thesaurus: Human annotators build sets of synonymous terms for concepts, without designating a Canonical term ({hotel, room, apartment, accommodation})
  - 3. b) Automatically generated thesaurus: statistics in a large corpus
  - 4. Query log mining: query reformulations done by users are logged and used to suggest good expansions to new users with similar queries

4.3. **Explain what relevance feedback is and how it works**

- Techniques that improve results based on feedback from the user.
  - Execute the original query q
  - Present results to the user allowing identification of relevant documents
  - Derive a refined query q' taking into account the relevance information
  - Execute the refined query and present the final results to the user

4.4. **Given results of a probabilistic retrieval model and user feedback as relevance judgments,**
apply relevance re-ranking

- $p(D_t|q,r) = \frac{N_{r,t}+0.5}{N_r+1}$ $\quad p(D_t|q,\bar{r}) = \frac{N_t - N_{r,t}+0.5}{N-N_r+1}$
  - N - total number of documents in the collection
  - N_r - the number of relevant documents
  - N_t - the number of documents containing the term t
  - $N_{r,t}$ - the number of relevant documents containing term t

4.5. **Given results of VSM, and user feedback as relevance judgments, apply relevance re-ranking using the the Rocchio's method**

- $q' = \alpha q + \beta \frac{1}{|D_r|}\sum_{d_j \in D_r} d_j - \gamma \frac{1}{|D_{\bar{r}}|}\sum_{d_j \in D_{\bar{r}}} d_j$
  - D_r and $D_{\bar{r}}$ are sets of known relevant and irrelevant documents.
  - The first term is the original query vector.
  - Other two terms make the result:
    - more similar to the centroid of relevant documents
    - less similar to the centroid of irrelevant documents
  - Constants α, β and γ balance between trusting relevance judgments and trusting the initial query.

4.6. **Explain what pseudo-relevance feedback is**

- Relevance feedback without a human user. In absence of user supplied relevance judgments, it is assumed the top k retrieved documents were relevant.

4.7. **List what a typical IR test collection is comprised of**

- Document collection
- Set of information needs (descriptions + queries)
- Set of relevance judgments for each query{document pair
  - Binary relevance or graded relevance judgments

4.8. **Explain why accuracy is not a very good evaluation metric for IR**

- Given a query, most documents (say 99%) are irrelevant. A search engine that retrieves nothing will already have accuracy of 0.99.

4.9. **Given example data, calculate the precision, recall, and F1-score**

$$Precision\ P = \frac{\#relevant\ documents\ retrieved}{\#retrieved\ documents} = \frac{t_p}{t_p+f_p} \qquad Recall\ R = \frac{\#relevant\ documents\ retrieved}{\#relevant\ documents} = \frac{t_p}{t_p+f_n}$$

$$F = \frac{(\beta^2+1)PR}{\beta^2 P + R}, \beta^2 = \frac{1-\alpha}{\alpha}$$

$\beta = 1$ - gives equal weight to precision and recall (F1-score)
$\beta = 0.5$ - emphasizes precision twice as much as recall

$\beta = 2$ - emphasized recall twice as much as precision

4.10. **Given example data, calculate MAP, P@k, and R-Precision**

- Average precision (AP) for query q: $AP(q) = \frac{1}{m}\sum_{k=1}^{m} P(R_k)$, where R_k are the top documents down to the k-th relevant document (both relevant and non-relevant)
- Mean average precision (MAP) is AP averaged over the set of queries Q:

$$MAP(q) = \frac{1}{|Q|}\sum_{j=1}^{|Q|}\frac{1}{m_j}\sum_{k=1}^{m_j} P(R_{jk})$$

- Precision at k (P@k) computes precision for the top k-ranked documents (e.g., P@5, P@10, P@20).
- R-precision computes precision at top-k documents, where k equals the number of relevant documents for the query.
  Example:
  - For a query q, there are four documents in the collection that are relevant (R), while all other are not relevant (N).
    Given q, the system returns a ranked list of eight documents:
    1 2 3 4 5 6 7 8
    N R N R N N N R
    Compute AP, P@5, and R-precision.

4.11. **Explain the motivation for combining popularity/importance score with document relevance for web search**

- Content scores not enough for efficient retrieval on the web due to web's massive size and spam websites.

4.12. **State the PageRank hypothesis**

- Web is a massive directed graph in which edges denote hyperlinks between web pages. Hyperlinks are recommendations. A page with more recommendations is more important. Recommendations from more important recommenders are worth more; the overall number of recommendation issued by the recommender also matters → A web page is important if it is pointed to by other important pages that do not point to too many other pages. ← Web page importance

4.13. **Define the PageRank iterative update formula in matrix form and explain the vectors and matrices used**

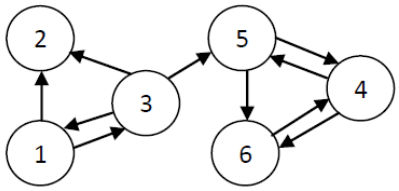
4.14. **Explain the conditions under which the iterative PageRank algorithm converges**

4.15. **Given a directed graph and initial relevance scores of the nodes, apply the PageRank**
algorithm and derive PageRank scores after several iterations

(combined answer)

- The PageRank of a page P_i is the sum of importances of all pages that have hyperlinks to P_i, each normalized with the total number of hyperlinks on that page.
  - $r(P_i) = \sum_{P_j \in B_{P_i}} \frac{r(P_j)}{|P_j|}$ (2)
  - B_{P_i} is the set of all pages that link to P_i
  - $|P_j|$ is the number of outgoing links from page P_j
  - PageRank scores of pages P_j linking to P_i are unknown

- Iterativeness over original PageRank summation:
  - Assign the same initial score to all pages: $P_i = \frac{1}{n}$, where n is the total number of pages.
  - Run the equation (2) iteratively, until the PageRank scores for all pages converge (remain the same in two consecutive iterations):
    - $r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|}$ (3)
  - PageRank summation in the matrix form



Iteration 0	Iteration 1	Iteration 2	Rank
$r_0(P_1) = 1/6$	$r_1(P_1) = 1/18$	$r_2(P_1) = 1/36$	5
$r_0(P_2) = 1/6$	$r_1(P_2) = 5/36$	$r_2(P_2) = 1/18$	4
$r_0(P_3) = 1/6$	$r_1(P_3) = 1/12$	$r_2(P_3) = 1/36$	5
$r_0(P_4) = 1/6$	$r_1(P_4) = 1/4$	$r_2(P_4) = 17/72$	1
$r_0(P_5) = 1/6$	$r_1(P_5) = 5/36$	$r_2(P_5) = 11/72$	3
$r_0(P_6) = 1/6$	$r_1(P_6) = 1/6$	$r_2(P_6) = 14/72$	2

- In equation (3), PageRank scores are computed for one page at a time
- Using a matrix form, all PageRank scores can be updated at once

$\boldsymbol{\pi}$ – the vector of PageRank scores: $\boldsymbol{\pi}_i = r(P_i)$

$\mathbf{H}$ – the row normalized Web graph adjacency matrix

$$\mathbf{H}_{ij} = \begin{cases} 1/|P_i|, & \text{if there is a link from page } P_i \text{ to page } P_j \\ 0, & \text{otherwise.} \end{cases}$$

$$\mathbf{H} = \begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

- The iterative update formula can now be rewritten in matrix form
  - $\pi_{k+1}^T = \pi_k^T H$ (4)
- Each iteration requires one vector-matrix multiplication: $O(n^2)$. Eq. (4) is power method applied to matrix H.
- Power method applied to some matrix P converges if P is:
  - Stochastic - each row is a probability distribution
  - Irreducible - there is a probability assigned to transition from each node to every other node
  - Aperiodic - no such cycle of length k for which $P = P^k, P^1 = P^{k+1}, P^2 = P^{k+2}, \ldots$

4.16. **State the HITS hypothesis and explain the concepts of hubs and authorities**

- HITS (Hypertext Induced Topic Search) hypothesis:
  - A page is a good hub if it points to good authorities, and a good authority if it is pointed to by good hubs.
- A page is considered a hub if it contains many hyperlinks.
- A page is considered an authority if many hyperlinks point to it.

5 Machine Learning for NLP

5.1. **List at least three disadvantages of rule based NLP systems**

- We just need too many rules to cover all the cases
- There are many exceptions that need to be handled
- We need expert knowledge (a linguist)

5.2. **List at least three advantages and disadvantages of machine-learning-based NLP Systems**

- Advantages
  - It is often much easier to manually label something than to figure out an algorithm that does the same automatically
  - Often labeling does not require (much) expert knowledge
  - We don't care about how complex/subjective the task is: we let the „data speak for itself" and we let the ML algorithm do the work
- Disadvantages
  - Data labeling can be expensive, especially if large amounts of data are required and/or expert (linguistic) knowledge
  - Data labeling can be tedious (slow, error-prone)
  - Sometimes, it requires quite of lot of training/coaching and many discussion rounds to settle the annotation disagreements

5.3. **Clarify the difference between supervised and unsupervised ML, as well as the difference between classification and regression**

- Supervised ML: we have labeled data as input
  - Classification: output is a discrete label (but there is no ordering between the labels)
  - Regression: output is a real-valued or integer number (obviously there is an ordering)

- Unsupervised ML: we have no labeled data
  - Clustering

5.4. **Given a short description of a concrete NLP task, recognize to what ML task it maps To**
Huh? ☹

5.5. **Define a binary, multi-class, and a multi-label classification task as a function mapping**
and clarify the difference between these

- Binary classification: just two output labels (yes/no, 0/1)
  - $\boldsymbol{h : X \rightarrow \{0, 1\}}$
- Multi-class classification: each instance has one of K labels
  - $\boldsymbol{h : X \rightarrow y}$, $\boldsymbol{y = \{1, \dots, K\}, K > 2}$
- Multi-label classification: an instance can have many labels at once

5.6. Explain what classifier training and testing are

- Training: adjustment of model parameters so that the classification error is minimized
  - The error is computed on a labeled training set, so we call this a training error
  - Training error is minimized with an optimization method
  - ML algorithms differ in the exact optimization criteria and the optimization method they use some algorithms use an approximate (iterative) optimization, others use exact optimization
  - Some algorithms work better on certain tasks. Generally, you need to experiment a lot to find out what works best.

- Testing
  - To test how well the classifier generalizes, we split the labeled test set into a training set and a test set (held-out data) The classifier does not use the test set for training. Once the classifier is trained, we run it on the test set and evaluate its performance
  - Generally, the performance on the test set will be (a bit) worse than on the training set, but this is a more realistic performance estimate

5.7. Explain what overfitting is and how it is prevented using cross-validation

- If the model is too complex, if will fit the training data very well, but perform poorly on new data. We call this overfitting.

- Cross-validation: to optimize the hyperparameters, we train the model h with various hyperparamer values, and test what model performs best. We cannot test on the training set nor on the test set, as this would lead to model overfitting

5.8. Outline the select-train-evaluate loop

Select – Train – Evaluate

1. Split the data set into a training set and a test set (e.g.,70–30%)
2. **Model selection:** for each hyperparameter value
3. Cross-validate the model on the training set (e.g., 5-fold CV)
4. Choose the best-performing hyperparameters
5. **Train** the model with these hyperparameters on the training set
6. **Evaluate** the trained model on the test set

5.9. Explain the motivation for one-hot encoding of feature values

- Some models (SVM, logistic regression) only work with real-valued vectors
- Why not encode categorical features as numbers from $\{1, \dots, K\}$?
  - Not a good idea because values should remain unordered (equidistant)
- Categorical features must be encoded using one-hot encoding
- K-dimensional binary vector with only one component set to 1

$$value\ 1 \rightarrow (1,0,0,\dots,0,0)$$
$$value\ 2 \rightarrow (0,1,0,\dots,0,0)$$
$$value\ n \rightarrow (0,0,0,\dots,0,1)$$

- Vector dimension equals the number of values the feature can take. For words, this will often be on the order of ten thousands

5.10. Explain what a sequence labeling problem is

- An issue with the standard classification problems. Assumption that individual classification decisions are independent. Many NLP problems do not satisfy this assumption. Because texts are sequences of words, labels for individual words depend on the labels assigned to its predecessor words
- Sequence labeling problem: each token in a sequence is assigned a label. Labels of tokens are dependent on the labels of other tokens in the sequence. In general, more sophisticated learning and inference techniques are needed

5.11. Explain how to frame sequence labeling as a classification task and describe the disadvantages of such an approach

- Sequence labeling as classification with outputs as inputs. Output labels of surrounding tokens are usually very good features. However, these are not available for all the surrounding tokens at classification time for the current token. We can use labels from either:
    - Preceding tokens, if we are moving from left to right (typically) E.g., in prev. ex. POS-tag of „Mark" as feature for classifying the POS-tag of „saw"
    - Succeeding tokens, if we are moving from right to left (less usual) E.g., in prev. ex. POS-tag of „saw" as feature for classifying the POS-tag of „Mark"
    - Shortcomings of sequence labeling as classification
        - Dificult to integrate labels from both side surrounding tokens as features.
        - The probability/uncertainty of token-wise decisions is not propagated.
        - The set of locally most probable decisions does not necessarily correspond to the jointly most probable sequence of labels

5.12. Explain the basic idea behind the HMM approach to sequence labeling

- Assumption that the next state only depends on the current state and is independent of previous history. A finite state machine with probabilistic state transitions.

5.13. Given HMM parameters and a word sequence, calculate the most likely state sequence

Given the following HMM parameters

$$\Pi = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{pmatrix} 0.3 \\ 0.3 \\ 0.4 \end{pmatrix} \quad A = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{array}{c} \begin{array}{ccc} s_1 & s_2 & s_3 \end{array} \\ \begin{pmatrix} 0.1 & 0.3 & 0.6 \\ 0.3 & 0.3 & 0.4 \\ 0.5 & 0.3 & 0.2 \end{pmatrix} \end{array} \quad B = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{array}{c} \begin{array}{ccc} o_1 & o_2 & o_3 \end{array} \\ \begin{pmatrix} 0.5 & 0.3 & 0.2 \\ 0.8 & 0.1 & 0.1 \\ 0.6 & 0.2 & 0.2 \end{pmatrix} \end{array}$$

and an output sequence $\mathbf{O} = \{o_3, o_1, o_2\}$, what is the most likely hidden state sequence $\mathbf{X}$ that generated $\mathbf{O}$?

- Step 1

$$\delta_1(1) = 0.2 \cdot 0.3 = 0.06$$
$$\delta_2(1) = 0.03 \cdot 0.1 = 0.03$$
$$\delta_3(1) = 0.4 \cdot 0.2 = 0.08$$

Step 2

$$\begin{aligned} \delta_1(2) &= \max(0.06 \cdot 0.1, 0.03 \cdot 0.3, 0.08 \cdot 0.5) \cdot 0.5 \\ &= \max(0.006, 0.009, 0.04) \cdot 0.5 = 0.02 \\ \psi_1(2) &= 3 \end{aligned}$$

$$\begin{aligned} \delta_2(2) &= \max(0.06 \cdot 0.3, 0.03 \cdot 0.3, 0.08 \cdot 0.3) \cdot 0.8 \\ &= \max(0.018, 0.009, 0.024) \cdot 0.8 = 0.0192 \\ \psi_2(2) &= 3 \end{aligned}$$

$$\begin{aligned} \delta_3(2) &= \max(0.06 \cdot 0.6, 0.03 \cdot 0.4, 0.08 \cdot 0.2) \cdot 0.6 \\ &= \max(0.036, 0.012, 0.016) \cdot 0.6 = 0.0216 \\ \psi_3(2) &= 1 \end{aligned}$$

- Step 3

$$\delta_1(2) = \max(0.02 \cdot 0.1, 0.0192 \cdot 0.3, 0.0216 \cdot 0.3) \cdot 0.5$$
$$= \max(0.002, 0.00576, 0.0108) \cdot 0.3 = 0.00324$$
$$\psi_1(2) = 3$$

$$\delta_2(2) = \max(0.02 \cdot 0.3, 0.0192 \cdot 0.3, 0.0216 \cdot 0.3) \cdot 0.1$$
$$= \max(0.006, 0.00576, 0.00648) \cdot 0.1 = 0.024$$
$$\psi_2(2) = 3$$

$$\delta_3(2) = \max(0.02 \cdot 0.6, 0.0192 \cdot 0.4, 0.0216 \cdot 0.2) \cdot 0.2$$
$$= \max(0.012, 0.00768, 0.00432) \cdot 0.2 = 0.0024$$
$$\psi_3(2) = 1$$

- using ψ we can reconstruct the optimal path $s_1 \rightarrow s_3 \rightarrow s_1$

5.14. Explain the main advantages of a CRF model over a HMM model

- CRF solves HMM problems:
  - Large number of parameters
  - Inability to adjust well to the concrete problem at hand
  - They model $P(X, Y)$ while $P(Y|X)$ is enough for prediction
  - The label bias problem

5.15. Explain the relation between annotation correctness and inter-annotator agreement

- Instead of measuring correctness, we measure annotation reliability: do humans consistently make the same decisions?
- Have two or more annotators annotate the same data independently but according to the same guidelines. Measure the inter-annotator agreement (IAA):
  - Agreement %
  - Cohen's Kappa
  - Fleiss' Kappa
  - P, R, F1 (results of one annotator taken as true labels)

5.16. Explain why Cohen's kappa is better than raw agreement score

5.17. Given example data with multi-class annotation, calculate and interpret Cohen's Kappa

---

6 Text Classification and Clustering

6.1. Give examples of text classification tasks

- Given document d as input, we wish to predict its class (category,label) from a fixed set of labels: $h(d) = \{c_1, \dots, c_k\}$

6.2. Explain the advantage of machine-learning-based text classification over its rulebased counterpart

- Handcrafted rule-based systems:
  - Classification rules based on occurrence/non-occurrence of certain words or word combinations in the document
  - Pros: Can be quite precise if de_ned by a domain expert
  - Cons: expensive to design and maintain, fragile
- Supervised ML:
  - Uses a classifier trained on manually labeled dana. Typically: NB, SVM, k-NN
  - Requires labeled data, but labeling is simpler than designing rules .More robust than rule-based systems: relies on statistics of all words in a document (a single occurrence of „money" does not imply that the document is about Finance)

6.3. List the text classification preprocessing steps

- Tokenization
- Stop-words removal
- Morphological normalization (stemming or lemmatization)
- Feature selection

6.4. Define the naive Bayes classifier

$$P(c_j|\mathbf{d}) = \frac{P(\mathbf{d}|c_j)P(c_j)}{\sum_k P(\mathbf{d}|c_j)P(c_k)}$$

$$P(c_j|\mathbf{d}) \propto P(\mathbf{d}|c_j)P(c_j)$$

$$P(c_j|\mathbf{d}) \propto P(c_j)\prod_{i=1}^{n} P(w_i|c_j)P(c_j)$$

$$h(\mathbf{d}) = \arg\max_j P(c_j)\prod_{i=1}^{n} P(w_i|c_j)$$

- To prevent underflows, we move to log-space:

$$h(\mathbf{d}) = \arg\max_j \left(\log P(c_j) + \sum_{i=1}^{n} \log P(w_i|c_j)\right)$$

6.5. Describe the multinomial document classification modeling

- At each position k in a document, any of the words w_i from the vocabulary can occur
- W_k is a categorical ("multinomial") variable with $|V|$ possible values: $W_k = w_i$ iff w_i occurs at position k
- $P(W_k = w_i|C = c_j)$ models the probability of word w_i occurring at position k in a document from class c_j
- Modeling separately every position would require too many parameters, thus we simplify and treat all positions as being equal. This gives us one categorical variable W for all word positions
- Then, $P(W = w_i|C = c_j)$ is the probability of word w_i occurring at any given position in document from class c_j
- Estimation:

$$P(w_i|c_j) = \frac{N(W = w_i, C = c_j) + 1}{N(W, C = c_j) + |V|}$$

$\Rightarrow$ Fraction of times in which word w_i appears in documents from c_j

6.6. Describe the Bernoulli document classification modeling

- Each word w_i from a vocabulary can either occur (perhaps many times) in the document or not occur at all
- W_i is a binary variable: $W_i = 1$ iff word w_i occurred in the document
- $P(W_i = 1|C = c_j)$ models the probability of word w_i occurring in a document from class c_j
- Estimation:

$$P(w_i|c_j) = \frac{N(W_i = 1, C = c_j) + 1}{N(C = c_j) + 2}$$

$\Rightarrow$ Fraction of documents from class c_j in which the word w_i occurs

6.7. Explain the commonalities and differences between the multinomial and the Bernoulli document classification modeling

- Both Multinomial and Bernoulli completely ignore the position of words in the document.
- Bernoulli ignores the counts, while Multinomial doesn't.

6.8. Given example documents constituting a train set, calculate the probabilities of a multinomial or a Bernoulli document model and classify a test document

DocID	Words in document	$c =$ China
d_1	Chinese Beijing Chinese	1
d_2	Chinese Chinese Shangai	1
d_3	Chinese Macao	1
d_4	Tokyo Japan Chinese	0
d_5	Chinese Chinese Chinese Tokyo Japan	?

- Multinomial

$$
\begin{aligned}
P(c) &= \tfrac{3}{4} \\
P(\neg c) &= \tfrac{1}{4} \\
P(\text{Chinese}|c) &= \tfrac{5+1}{8+6} = \tfrac{3}{7} \\
P(\text{Tokyo}|c) &= \tfrac{0+1}{8+6} = \tfrac{1}{14} \\
P(\text{Japan}|c) &= \tfrac{0+1}{8+6} = \tfrac{1}{14} \\
P(\text{Chinese}|\neg c) &= \tfrac{1+1}{3+6} = \tfrac{2}{9} \\
P(\text{Tokyo}|\neg c) &= \tfrac{1+1}{3+6} = \tfrac{2}{9} \\
P(\text{Japan}|\neg c) &= \tfrac{1+1}{3+6} = \tfrac{2}{9}
\end{aligned}
$$

Prediction for d_5:

$$
\begin{aligned}
P(c|d_5) &\propto P(d_5, c) = \tfrac{3}{4} \cdot (\tfrac{3}{7})^3 \cdot \tfrac{1}{14} \cdot \tfrac{1}{14} = 0.0003 \\
P(\neg c|d_5) &\propto P(d_5, \neg c) = \tfrac{1}{4} \cdot (\tfrac{2}{9})^3 \cdot \tfrac{2}{9} \cdot \tfrac{2}{9} = 0.0001
\end{aligned}
$$

- Bernoulli:

$$
\begin{aligned}
P(\text{Chinese}|c) &= \frac{3+1}{3+2} = \frac{4}{5} \\
P(\text{Japan}|c) = P(\text{Tokyo}|c) &= \frac{0+1}{3+2} = \frac{1}{5} \\
P(\text{Beijing}|c) = P(\text{Macao}|c) = P(\text{Shangai}|c) &= \frac{1+1}{3+2} = \frac{2}{5} \\
P(\text{Chinese}|\neg c) &= \frac{1+1}{1+2} = \frac{2}{3} \\
P(\text{Japan}|\neg c) = P(\text{Tokyo}|\neg c) &= \frac{1+1}{1+2} = \frac{2}{3} \\
P(\text{Beijing}|c) = P(\text{Macao}|\neg c) = P(\text{Shangai}|\neg c) &= \frac{0+1}{1+2} = \frac{1}{3}
\end{aligned}
$$

Prediction for d_5:

$$
\begin{aligned}
P(c|d_5) &\propto P(d_5, c) = \tfrac{3}{4} \cdot \tfrac{4}{5} \cdot \tfrac{1}{5} \cdot \tfrac{1}{5} \cdot (1 - \tfrac{2}{5}) \cdot (1 - \tfrac{2}{5}) \cdot (1 - \tfrac{2}{5}) = 0.005 \\
P(\neg c|d_5) &\propto P(d_5, \neg c) = \tfrac{1}{4} \cdot \tfrac{2}{3} \cdot \tfrac{2}{3} \cdot \tfrac{2}{3} \cdot (1 - \tfrac{1}{3}) \cdot (1 - \tfrac{1}{3}) \cdot (1 - \tfrac{1}{3}) = 0.022
\end{aligned}
$$

6.9. Explain what clustering is

- Cluster analysis is a multivariate statistical technique that allows automated generation of groupings in data

6.10. Describe single pass clustering

- Single pass clustering - each document to its closest cluster If maximal similarity score is beneath a certain threshold, a new cluster is started

6.11. Describe hierarchical agglomerative clustering

- Agglomerative clustering starts from n singleton clusters which are iteratively grouped together to form larger and larger clusters Divisive clustering starts from a single cluster with all n documents, iteratively divided into smaller and smaller clusters

6.12. Describe the k-means algorithm

- Let us have K clusters, let b_k^i be and indicator variable that equals 1 when document i belongs to cluster k and μ_k be the centroid of the k-th cluster
- The cost function being optimized is $J = \sum_{k=1}^{K} \sum_{i=1}^{N} b_k^i ||x^i - \mu_k||^2$
- We want to find the grouping that minimizes the cost function. Analytical minimization is not possible because the parameters b_k and μ_k are mutually dependent iterative minimization of the cost function J

1. Select random centroids $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$
2. The cost J is minimized if the examples are assigned to the cluster represented by the nearest centroid

$$b_k^{(i)} = \begin{cases} 1, & \text{if } k = \arg\min_j \|\mathbf{x}^{(i)} - \boldsymbol{\mu}_j\| \\ 0, & \text{otherwise.} \end{cases}$$

3. With indicator variables $b_k^{(i)}$ fixed, we can get the new centroids

$$\boldsymbol{\mu}_k = \frac{\sum_i b_k^{(i)} \mathbf{x}^{(i)}}{\sum_i b_k^{(i)}}$$

4. With the change of centroids, some indicator variables $b_k^{(i)}$ have changed. Repeat steps 2–3 until convergence

7 Latent Semantic Models in NLP and IR

7.1. Explain what is a latent representation of a document and what it is used for

- Main problems of classical retrieval models:
  - Synonymy - multiple words with the same meaning
  - Polysemy - same word with multiple meanings
- Most classical models derive a document representation by analyzing words appearing in a particular document. Global structure of the document collection is ignored
- Idea: derive document representations which find and exploit latent (hidden) structure and patterns present in the entire collection
- We would like to map a text to a lower dimensional representation while preserving general meaning (the topic structure of documents)
- Latent models usually handle synonymy and polysemy better

7.2. Outline the basic idea behind LSA

- Assume a collection of N documents with a vocabulary of M words
- We begin by constructing a $M\ x\ N$ word-document matrix A
- Rows correspond to words, Columns correspond to documents
- Element $A[i,j]$ contains the count of word i in document j. Instead of counts, other weighting techniques such as tf-idf can be used (e.g., $A[i,j] = tf_{ij} * idf_{ij}$)

7.3. Given an SVD decomposition of a (small) word-document matrix, calculate the latent semantic representations of documents

- Formula : $A = U\Sigma V^T$

$$A = \begin{array}{c} \\ \textit{president} \\ \textit{minister} \\ \textit{speech} \\ \textit{law} \\ \textit{ball} \\ \textit{score} \\ \textit{player} \\ \textit{run} \\ \textit{person} \\ \textit{piano} \\ \textit{mouse} \end{array} \begin{array}{c} \begin{array}{cccccc} d1 & d2 & d3 & d4 & d5 & d6 \end{array} \\ \begin{pmatrix} 3 & 2 & 0 & 1 & 0 & 0 \\ 4 & 1 & 3 & 0 & 0 & 0 \\ 2 & 5 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 4 & 0 & 2 \\ 0 & 0 & 0 & 3 & 2 & 3 \\ 0 & 0 & 1 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

$$U = \begin{bmatrix} -0.43 & 0.13 & 0.22 & -0.01 & -0.55 & -0.09 & \\ -0.53 & 0.25 & -0.28 & 0.62 & -0.09 & -0.07 & \\ -0.58 & 0.33 & 0.18 & -0.56 & 0.37 & 0.06 & \\ -0.12 & -0.05 & -0.19 & 0.28 & 0.64 & 0.26 & \\ -0.22 & -0.51 & 0.53 & 0.17 & 0.10 & -0.32 & \\ -0.26 & -0.62 & 0.08 & -0.05 & -0.03 & 0.41 & \cdots \\ -0.22 & -0.40 & -0.69 & -0.25 & -0.12 & -0.21 & \\ -0.03 & -0.06 & -0.18 & -0.11 & -0.12 & -0.07 & \\ -0.11 & -0.03 & 0.02 & 0.13 & -0.18 & 0.60 & \\ -0.10 & -0.02 & -0.12 & -0.29 & 0.01 & -0.06 & \\ -0.09 & -0.08 & 0.01 & 0.16 & 0.26 & -0.47 & \end{bmatrix} \begin{array}{l} \textit{president} \\ \textit{minister} \\ \textit{speech} \\ \textit{law} \\ \textit{ball} \\ \textit{score} \\ \textit{player} \\ \textit{run} \\ \textit{person} \\ \textit{piano} \\ \textit{mouse} \end{array}$$

$$diag(\Sigma) = \begin{bmatrix} 7.66 & 6.81 & 3.99 & 3.72 & 2.34 & 1.54 \end{bmatrix}$$

$$V^T = \begin{bmatrix} -0.60 & -0.56 & -0.35 & -0.30 & -0.19 & -0.21 \\ 0.29 & 0.31 & 0.07 & -0.62 & -0.42 & -0.49 \\ -0.01 & 0.23 & -0.42 & 0.47 & -0.72 & 0.11 \\ 0.38 & -0.66 & 0.47 & 0.11 & -0.40 & 0.09 \\ -0.61 & 0.29 & 0.65 & -0.03 & -0.27 & 0.19 \\ 0.12 & 0.01 & -0.20 & -0.52 & -0.10 & 0.80 \end{bmatrix}$$

$$\Sigma V^T = \begin{bmatrix} -4.66 & -4.37 & -2.71 & -2.37 & -1.51 & -1.65 \\ 2.01 & 2.12 & 0.49 & -4.23 & -2.93 & -3.35 \\ -0.06 & 0.92 & -1.70 & 1.90 & -2.90 & 0.44 \\ 1.45 & -2.48 & 1.75 & 0.43 & -1.51 & 0.34 \\ -1.44 & 0.68 & 1.53 & -0.09 & -0.64 & 0.46 \\ 0.19 & 0.02 & -0.32 & -0.82 & -0.16 & 1.25 \\ & & \cdots & & & \end{bmatrix}$$

- From diagonal matrix we can see first and second topics play a much bigger role in reconstructing. We pick first 2 columns in V^T and first 2 rows in ΣV^T:

$$U \qquad\qquad \Sigma V^T$$

$$\begin{bmatrix} -0.43 & 0.13 \\ -0.53 & 0.25 \\ -0.58 & 0.33 \\ -0.12 & -0.05 \\ -0.22 & -0.51 \\ -0.26 & -0.62 \\ -0.22 & -0.40 \\ -0.03 & -0.06 \\ -0.11 & -0.03 \\ -0.10 & -0.02 \\ -0.09 & -0.08 \end{bmatrix} \cdot \begin{bmatrix} -4.66 & -4.37 & -2.71 & -2.37 & -1.51 & -1.65 \\ 2.01 & 2.12 & 0.49 & -4.23 & -2.93 & -3.35 \end{bmatrix}$$

$$\begin{array}{c} \begin{matrix} d1 & d2 & d3 & d4 & d5 & d6 \end{matrix} \\ \begin{pmatrix} 2.3 & 2.1 & 1.2 & 0.5 & 0.3 & 0.3 \\ 3.0 & 2.8 & 1.6 & 0.2 & 0.1 & 0.0 \\ 3.3 & 3.2 & 1.7 & 0.0 & -0.1 & -0.1 \\ 0.5 & 0.4 & 0.3 & 0.5 & 0.3 & 0.4 \\ 0.0 & -0.1 & 0.3 & 2.7 & 1.8 & 2.1 \\ 0.0 & -0.2 & 0.4 & 3.2 & 2.2 & 2.5 \\ 0.2 & 0.1 & 0.4 & 2.2 & 1.5 & 1.7 \\ 0.0 & 0.0 & 0.0 & 0.3 & 0.2 & 0.3 \\ 0.4 & 0.4 & 0.3 & 0.4 & 0.2 & 0.3 \\ 0.4 & 0.4 & 0.3 & 0.3 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.5 & 0.4 & 0.4 \end{pmatrix} \end{array}$$

7.4. Outline the basic idea behind LDA

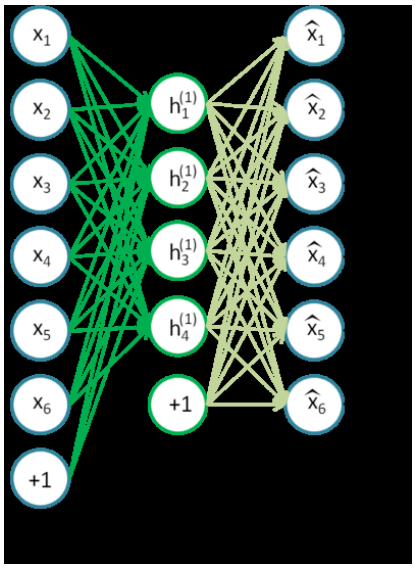
- A multinomial (categorical) distribution is a probability distribution over a discrete set of possible events.
- Dirichlet distribution: a distribution over all vectors of length K summing up to 1 (also called the K-1 simplex)
- LDA assumes a collection of M documents was generated as follows (K - number of topics, V - number of possible words)
- The goal of inference is to determine values of the hidden variables

7.5. Explain what ϕ, θ and Z stand for in the LDA model

- ϕ **-** prior assumptions about topic word distributions
- θ - document topic distributions
- Z - topic for the word in document

7.6. Outline the basic idea of how autoencoders generate latent semantic representations

- Idea: use neural networks to perform a non-linear projection into a lower dimensional space
- Networks are trained to minimize error when reconstructing original data from the reduced representation
- Training is often done by gradient descent to make $\hat{x}$ as close as possible to original x
- Outputs of hidden units represent the document in the latent semantic space



7.7. Outline the basic idea how word2vec generates word vectors

- (from wikipedija)These models are shallow, two-layer neural networks, that are trained to reconstruct linguistic contexts of words: the network is shown a word, and must guess which words occurred in adjacent positions in an input text. The order of the remaining words is not important (bag-of-words assumption).