1. Introduction to information retrieval

Prof. Dr. Goran Glavaš

Data and Web Science Group
Fakultät für Wirtschaftsinformatik und Wirtschaftsmathematik
Universität Mannheim

CreativeCommons Attribution-NonCommercial-ShareAlike 4.0 International
After this lecture, you’ll...

- Understand the basic concepts in information retrieval
- Know how to represent and preprocess text for IR
- Understand the general formalization of IR models

- Know what this course is about and be glad you’ve enrolled it
- Know which topics we will cover
- Hopefully be intrigued by some of the topics
- Know what’s your part of the job to earn credits
Outline

**Part one**
- What is information retrieval?
- Text representations and preprocessing
- General information retrieval model

**Part two**
- About the course(s) (IE 663 + IE 691)
- Topics
- Organization
Outline

- **Part one**
  - What is information retrieval?
  - Text representations and preprocessing
  - General information retrieval model

- **Part two**
  - About the course
  - Topics
  - Organization
"Retrieval" and "search"

- What is your first association to "information retrieval"?
- What is your first association to "search" (or "search engine")?
Retrieval and search
What is information retrieval?

- **Information retrieval** is the activity of obtaining information resources **relevant** for an user’s **information need** from a **collection** of **information resources**.

- **Elements of an information retrieval process:**
  1. Information needs (users express them in the form of queries)
  2. Information (re)sources, most often unstructured (text, images, video, audio, etc.)
  3. A system/method/model for identifying (re)sources relevant for a given information need (usually from a large collection of information resources)
Information needs

- **Information need** is an individual or group’s desire to locate and obtain information to satisfy a conscious or unconscious need
  - i.e., needs and interests that call for information

- Information needs (conscious or unconscious) are expressed as queries
  - When **retrieving texts**, queries are words or phrases (e.g., „Olympics in London‟)
  - In image retrieval queries can also be images
Why text information retrieval?

- Because of large repositories of unstructured information sources
  - Companies – technical documentation, business documents, contracts, ...
  - Governments – documentation, regulation, laws, ...
  - Science – publications (e.g., Google Scholar)
  - Personal collections – books, emails, files

- **World Wide Web** – the largest document collection of all
  - Additional challenges due to sheer scale
Why text information retrieval?

- Unstructured sources (text) vs. structured sources (databases)
Text information retrieval

- This course is about retrieval of text, where models differ in:
  - Representations of documents and queries
  - Methods for determining (degree of) relevance of a document for a given query

- In most IR models relevance is expressed as a score and not a binary decision
  - Documents are ranked in decreasing order according to assigned relevance scores
  - Relevance scores usually incorporate an element of uncertainty
Outline

Part one
- What is information retrieval?
- Text representations and preprocessing
- General information retrieval model

Part two
- About the course
- Topics
- Organization
1. **Unstructured representation**
   - Text represented as an **unordered set of terms** (the so-called **bag of words**)
   - Considerable **oversimplification**
     - We are ignoring the syntax, semantics, and pragmatics of text
     - Is this problematic?

   **Q:** “Revenue of Apple”
   **D:** “*Apple* Pencil 2 ‘to launch in March 2017‘... Microsoft faces drop in *revenue* in the 3rd quarter…”

   - Despite oversimplifying, BoW representations yield good IR performance

   - BoW is **de facto** standard IR representation
     - Due to simplicity and speed
2. **Weakly-structured representations**
   - Certain groups of terms given more importance – e.g., nouns or named entities
   - Other terms’ contribution is either downscaled or completely ignored
   - Some natural language processing (NLP) tools required
     - Part-of-speech (POS) tagger to identify nouns or named entity recognizer (NER) to identify named entities
     - Additional preprocessing can be costly

3. **Structured representations**
   - For example, graphs in which nodes represent some terms/concepts and edges semantic relations between them
   - Sophisticated information extraction (IE) and NLP tools needed to induce structure
   - IE models typically not accurate enough and time-costly
   - Structured representations are virtually not used in IR
Text representations in IR

- Document snippet

„One evening Frodo and Sam were walking together in the cool twilight. Both of them felt restless again. On Frodo suddenly the shadow of parting had falling: the time to leave Lothlorien was near. “

- Unstructured (bag-of-words) representation

\{(One, 1), (evening, 1), (Frodo, 2), (and, 2), (Sam, 1) (were, 1), (walking, 1), (together, 1), (in, 1), (the, 3), (cool, 1), (twilight, 1), (Both, 1), (of, 2), (them, 1), (felt, 1), (restless, 1), (again, 1), (On, 1), (suddenly, 1), (shadow, 1), (parting, 1), (had, 1), (falling, 1), (time, 1), (to, 1), (leave, 1), (Lothlorien, 1), (was, 1), (near, 1)\}
Text representations in IR

- Weakly-structured representations
  - Bag of nouns
    \{(evening, 1), (Frodo, 2), (Sam, 1), (twilight, 1), (shadow, 1), (parting, 1), (time, 1), (Lothlorien, 1)\}
  
  - Bag of named entities
    \{(Frodo, 2), (Sam, 1), (Lothlorien, 1)\}
„One evening Frodo and Sam were walking together in the cool twilight. Both of them felt restless again. On Frodo suddenly the shadow of parting had falling: the time to leave Lothlorien was near. ”

- Structured representation
  - For example, event-based structure

- Building such structure requires sophisticated natural language processing tools
- Structured document representations have not been shown beneficial for IR
Text preprocessing

- So, in IR, we most often use unstructured text representations
  - Text is represented as unordered set of terms (i.e., \textit{bag of words})

- However, many details about the exact representation are still undefined
  - How do we "split" text into terms? Can this be done in more than one way?
  - Do we consider all terms, or do we want to eliminate some?
    - E.g., functional words that have little meaning like articles and prepositions?
  - How do we treat different forms of the same word?
    - E.g., should "house" be treated the same as "houses"? What about "housing"?
  - What about synonyms or same concepts in different languages?
  - On a more technical side: what about different document formats?
Text preprocessing

- The preprocessing (i.e., preparing text for the retrieval process) usually involves the following steps:
  1. Extracting pure textual content (e.g., from HTML, PDF, Word)
  2. Language detection
     - Optional – if you’re dealing with multilingual document collections
  3. Tokenization (separating text into character sequences)
  4. Morphological normalization (lemmatization or stemming)
  5. Stopword removal

- After preprocessing, the text (i.e., the document) is ready to be indexed
  - More on indexing in the upcoming lectures
Tokens and terms

- **Word** is a delimited string of characters as it appears in the text
- **Term** is a normalized form of the word (accounting for morphology, spelling, etc.)
  - Word and term are in the same equivalence class – in informal speech they are often used interchangeably
- **Token** is an instance of a word or term occurring in a document
  - Tokens are “words” in the general sense
  - But numbers, punctuation, and special characters are also tokens
- **Tokenization** is a process, typically automated, of breaking down the text (one long string) into a sequence of tokens (shorter strings)
Tokenization

- Two types of methods for tokenization
  - Rule-based (i.e., heuristic)
  - Based on supervised machine learning models
    - Learn from manually tokenized texts

- Tokenization might seem simple, but it’s not always unambiguous
  - E.g., a simple rule: split string on all whitespaces
    - “Hewlett-Packard declared losses” -> „Hewlett-Packard”, „declared”, „losses”
    - Would we want to split „Hewlett” from „Packard”? What about „lower-case”?
    - What about „Denmark’s mountains”: „Denmark” and „’s”, or „Denmarks”, or „Denmark”?

- What about tokenizing numbers and punctuation?
  - „19/1/2017”, „55 B.C.”, „+49 176 832 40 332”, „IP: 192.168.0.1”
  - Sometimes spaces are not an indication of an end of a token
Issues in tokenization

- What about different languages?
- German has numerous compounds
  - „Lebensversicherungsgesellschaftsangestellter“ (life insurance company employee)
  - Is this a single token or 4 tokens?
  - IR systems for German texts greatly benefit from a compound splitting module
- How about languages that don’t segment text using whitespaces at all?
  - E.g., Chinese
  - „莎拉波娃现在居住在美国东南部的佛罗里达“
Normalization

- Normalization or standardization can involve various changes to the token
  - Error/spelling correction – repairing the incorrect word
  - Case-folding – converting all letters to lower case
    - „Morgen will ich in MIT” – is this German preposition „mit”?  
      - Often best to lower case everything (queries and documents)
  - How does Google do it?
    - „C.A.T.” (information need: Caterpillar Inc.)
      - returns **cat (animal)** as the first result

- Morphological normalization
  - Reducing different forms of the „same” word to a common representative form
Morphological normalization

- **Inflectional normalization** (or *lemmatization*) reduces all lexico-syntactic forms of the same word to one standard form, **lemma** (dictionary headword form)
  - Nouns: singular form in „nominative” case
  - Verbs: infinitive form
  - E.g., „houses” -> „house”, „tried” -> „try”

- **Derivational normalization** reduces all words syntactically derived from some word to the original word (even if the derived word has different meaning)
  - Derivational operators often change the part-of-speech of the word
  - E.g., „destruction” -> „destroy”

- Most IR systems perform inflectional but not derivational normalization
Stemming

- Lemmatization reduces words to dictionary headword entries
  - I.e., the resulting lemma is a string that is again a **valid word** in the language

- **Stemming** is the procedure of reducing the word to its grammatical (morpho-syntactic) root
  - The result of stemming is not necessarily a valid word of the language
    - E.g., „recognized“ -> „recogniz“, „incredibly“ -> „incredibl“
  - Stemming removes suffixes with heuristics
    - E.g., „automates“, „automatic“, „automation“ will all be reduced to „automat“
  - Stemming is „more aggressive“ than lemmatization and „less aggressive“ than derivational normalization
    - „More aggressive“ means more different words are normalized to the same form
  - Stemming is more frequently used in IR systems than lemmatization
Porter’s algorithm

- Most common algorithm for English stemming
- Rule-based algorithm
  - Grammatical conventions and 5 phases of reduction
  - Phases are executed sequentially, one at a time
  - Each phase consists of a set of concurrent suffix-trimming rules
    - If multiple rules apply, use the one that removes the longest suffix

- More on Porter’s stemmer:
  - [http://snowball.tartarus.org/algorithms/porter/stemmer.html](http://snowball.tartarus.org/algorithms/porter/stemmer.html)

- Similar algorithms have been developed for other languages as well
Porter’s algorithm

- Examples of rules
  - “-ing” -> “”
  - “-ly” -> “”
  - “-ses” -> “ss”
  - “-ational” -> “ate”
  - “-tional” -> “tion”

- Rules are sensitive to the measure of “how much of a word” a string is
  - Rules consider sequences of consonants and vowels, e.g., [C][VC]m[V]

- Rules also often take into account the length of the remaining “root”
  - E.g., “ement” -> “” is valid only if the remaining word has more than one syllable
    - “replacement” -> “replac” but “cement” -> “cement”
Expansion vs. normalization

- An alternative to normalization is the expansion of the query words
  - I.e., We search for alternative forms of the query word as well

- Example
  - **Query:** window  **Search:** window, windows
  - **Query:** windows  **Search:** Windows, windows, window
  - **Query:** Windows  **Search:** Windows

- Theoretically more powerful (no need for imperfect normalization)

- In practice less efficient as we need to index all words we will be looking for
  - Some languages are highly inflectional and one word can have many different forms
  - E.g., Finnish can have up to 14 different case forms for nouns
    - *omena* (apple) -> *omenan, omenaa, omenaan, omenat, omenien, omenoiden, omenojen, omenain, omenia, omenoita, omenoja, omeniin, omenoihin*
Stopword removal

- **Stopwords** are semantically poor terms such as articles, prepositions, conjunctions, pronouns, etc.

- Removal of stopwords is one of the most common steps of IR text preprocessing

- **Q:** Why would we want to remove the stopwords?
  - **A:** Because stopwords have very little meaning, they do not determine whether a document is relevant or not
  - **A:** Removing stopwords reduces the size of vocabulary (and index) and makes retrieval process more efficient
  - **A:** Including stopwords may lead to **false positives** because of stopword matches between query and documents

- Stopword lists for a number of languages:
  - [http://www.ranks.nl/stopwords](http://www.ranks.nl/stopwords)
Outline

- **Part one**
  - What is information retrieval?
  - Text representations and preprocessing
  - General information retrieval model

- **Part two**
  - About the course
  - Topics
  - Organization
General information retrieval model

- We’ve seen what information retrieval is and how to preprocess text
- Now, let’s formalize the general information retrieval model
  - Consider this as a „placeholder” for all concrete IR models we will cover later

- Each functional retrieval system implements the following three components
  1. Representation of a raw query text
     - To be used for matching against documents in the collection
  2. Representation of a raw document text
     - To be used for matching against the query
     - May or may not be the same representation as the one used for query
  3. A function for determining the relevance of documents for the query
     - Taking as input document and query representations – (1) and (2)
Formally, a general retrieval model is a triple of functions \((f_d, f_q, r)\):

1. \(f_d\) is a function that maps documents (raw text) to their representation 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 (raw text) to their representation for retrieval, i.e., \(f_q(q) = s_q\), where \(s_q\) is the retrieval representation of the document \(q\);
   
   - Depending on the IR model, \(f_d\) and \(f_q\) may or may not be the same function

3. \(r\) is a ranking function which computes a real number indicating the potential relevance of document \(d\) for query \(q\), using representations \(p_d\) and \(s_q\):

\[
\text{rel}(d,q) = r(f_d(d), f_q(q)) = r(p_d, s_q)
\]
Index terms and term weights

- **Index terms** are all terms in the collection (i.e., the vocabulary)
  - Except those we ignore in preprocessing (like stopwords)
  - The set of all index terms: \( K = \{ k_1, k_2, \ldots, k_t \} \)
  - Each term \( k_i \) is, for each document \( d_j \), assigned a weight \( w_{ij} \)
  - The weight of the index terms not appearing in the document is 0
- Document \( d_j \) is represented by term vector \( [w_1j, w_2j, \ldots, w_tj] \) where \( t \) is the number of index terms
- Let \( g \) be the function that computes the weights, i.e., \( w_{ij} = g(k_i, d_j) \)
- Different choices for the weight-computation function \( g \) and the ranking function \( r \) define different IR models
IR paradigms

- Information retrieval models roughly fall into following paradigms:
  1. Set theoretic models
     - Boolean model
     - Extended Boolean model
  2. Algebraic models
     - Vector space model
     - Latent models
       - Latent semantic indexing (LSI), Random indexing, Topic modelling for IR
  3. Probabilistic retrieval
     - Classic probabilistic retrieval: Binary independence model, BM11, BM25
     - Language models for IR
  4. Semantic ad-hoc retrieval
     - Embedding models
IR paradigms

- Different models are used in the Web search
  - Due to sheer size of the Web
  - Because users have no control over the content of the collection
    - Q: What is the problem if only content is considered for relevance?
      - A: Easy to create spam documents that would be very relevant for certain queries
  - Ranking algorithms also exploit the linked structure of the Web
    - PageRank
    - HITS
Outline

- **Part one**
  - What is information retrieval?
  - Text representations and preprocessing
  - General information retrieval model

- **Part two**
  - About the course
  - Topics
  - Organization
Course description

Q: Why this course?
- A: Because large collections of unstructured documents from which we retrieve information are all around
- A: Because there are many efficient models to retrieve information, some more suitable than others in different settings
- A: Because as information workers and data scientists you are likely to sooner or later have to design/implement a system that retrieves some information from unstructured data collections

Course purpose
- Provide a systematic overview of both traditional and advanced methods for text retrieval and web search
Course description

- **Target audience** are students who want to
  - Gain **theoretical understanding** of basic and advanced information retrieval models
  - Obtain **practical (hands-on) experience** implementing IR & WS techniques

- **Prerequisites**
  - Fundamental knowledge of
    - Linear algebra
    - Probability theory
    - Algorithms and data structures
  - **For IE 681**: Programming skills in a higher-level programming language
    - E.g., Java, Python, C#, C++
    - Necessary for homeworks and project
  - Helpful, **but not necessary**:
    - Knowledge of natural language processing
    - Knowledge of machine learning
Course description

- What this course covers
  - Basic theoretical concepts in information retrieval
  - Several traditional information retrieval models
  - Some advanced/recent IR models and techniques
  - IR evaluation
  - Web search and web ranking algorithms
Course description

- What this course doesn’t cover
  - Natural language processing / Computational linguistics
    - We’ll cover only as much as needed for IR, but won’t go into much depth
  - Machine learning
    - We’ll cover basics needed for IR, but won’t explain the inner workings of ML algorithms
  - Multimedia retrieval (search for images, video, audio)
    - Out of focus, we are interested primarily in text
Textbooks


Course content and schedule

- **Lecture 01**: Introduction to Information Retrieval (Feb 17)
- **Lecture 02**: Boolean Retrieval and Term Indexing (Feb 24)
- **Lecture 03**: Data Structures in IR and Tolerant Retrieval (Mar 2)
- **Lecture 04**: Term Weighting and Vector Space Model (Mar 9)
- **Lectures 05 & 06**: Probabilistic IR and Language Modeling for IR (Mar 16)
- **Lecture 07**: Relevance Feedback and Query Expansion (Mar 23)
- **Lecture 08**: Latent and Semantic Information Retrieval Models (Mar 30)
- **Easter break**: Apr 6 & Apr 13
- **Lecture 09**: Classification, Clustering, Learning to Rank, Evaluation (Apr 20)
- **Lecture 10**: Web Search and Link Analysis (Apr 27)
Examination and grading

- **IE 663: Final exam**
  - Exam will assess both *theoretical* and *practical* knowledge
  - **Preparation** for the exam:
    - Exercises
  - 50% of points necessary to pass to course
Communication

- This course is powered by the **Data and Web Science (DWS)** group
- Your IR & WS teachers
  - Prof. Dr. Goran Glavaš (lecturer)
  - Robert Litschko (teaching assistant)
- Office hours (Goran)
  - **Fridays at 10:00** *(in lecture weeks only)*
  - B6 29, building C, Room C1.08
  - Visits should be previously **announced via email**
- E-mail communication
  - Only for **really urgent matters**, otherwise come in office hours
  - If you’re wondering whether your matter is urgent or not, it probably isn’t 😊
- All relevant information will be posted **timely** in ILIAS
Is this course hard?

- To an extent, this depends
  - On your previous knowledge (linear algebra, probability theory, NLP, ML, ...)
  - On your programming skills (for the Project course)

- But primarily this depends on
  - Your interest in the IR & WS topics
  - Your enthusiasm and willingness to learn new stuff
  - The amount of time and effort you invest into this course

- This course is **6 (3+3) ECTS** credits
  - One credit should amount to **25-30** hours of your time
  - Our job is to make sure that this is the amount of effort you put in the course
Now you...

- Understand the basic concepts in information retrieval
- Know how to represent and preprocess text for IR
- Understand the general formalization of IR models

- Know what this course is about and be glad you’ve enrolled it
- Know which topics we will cover
- Are hopefully intrigued by some of the topics
- Know what’s your part of the job to earn credits
Can I pass this course?

YES!!
YOU CAN DO IT!!