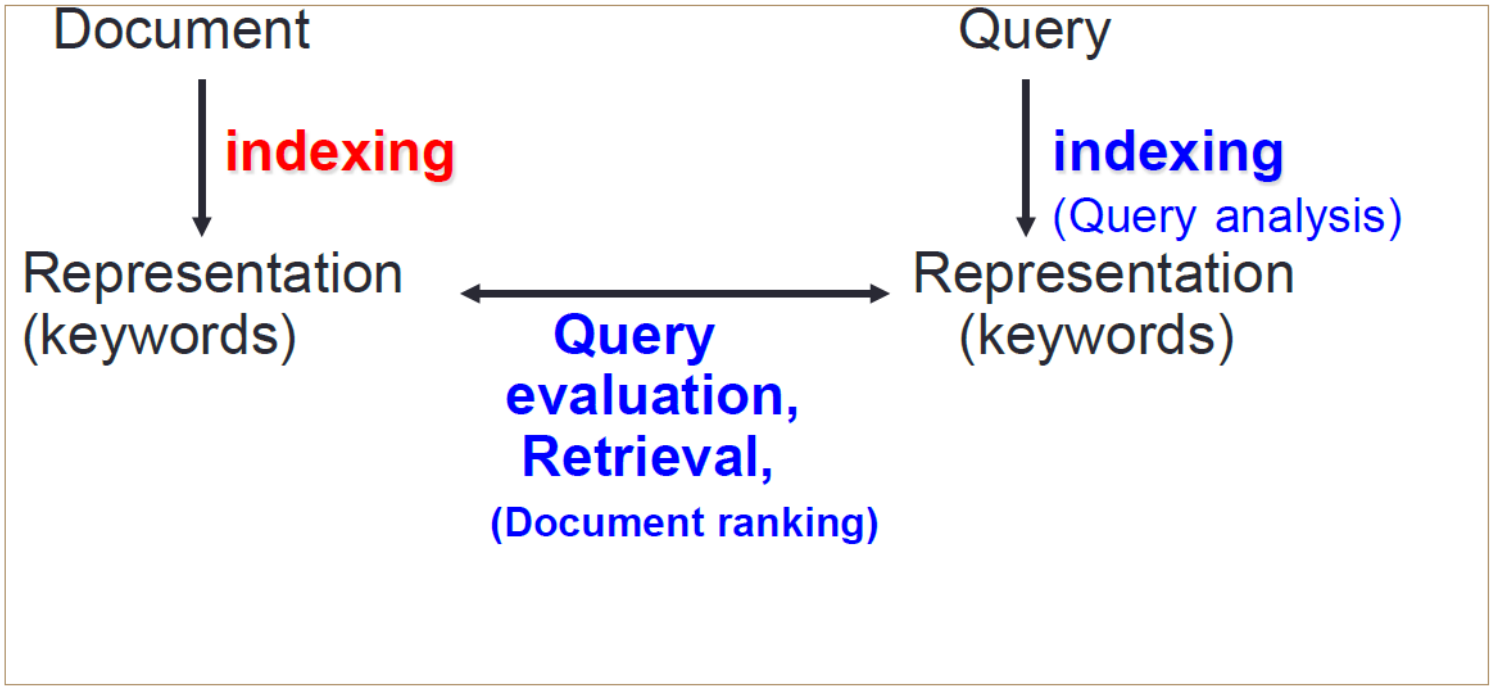
Information retrieval (IR)

By: **Dr. LOUNNAS Bilal**

Indexing-based IR



Basics - Goals

- Recognize the contents of a document
  - What are the main topics of the document?
  - What are the basic units (indexing units) to represent them?
  - How to weight their importance?
- Support fast search given a query
  - Given a query, find the documents that contain the words.

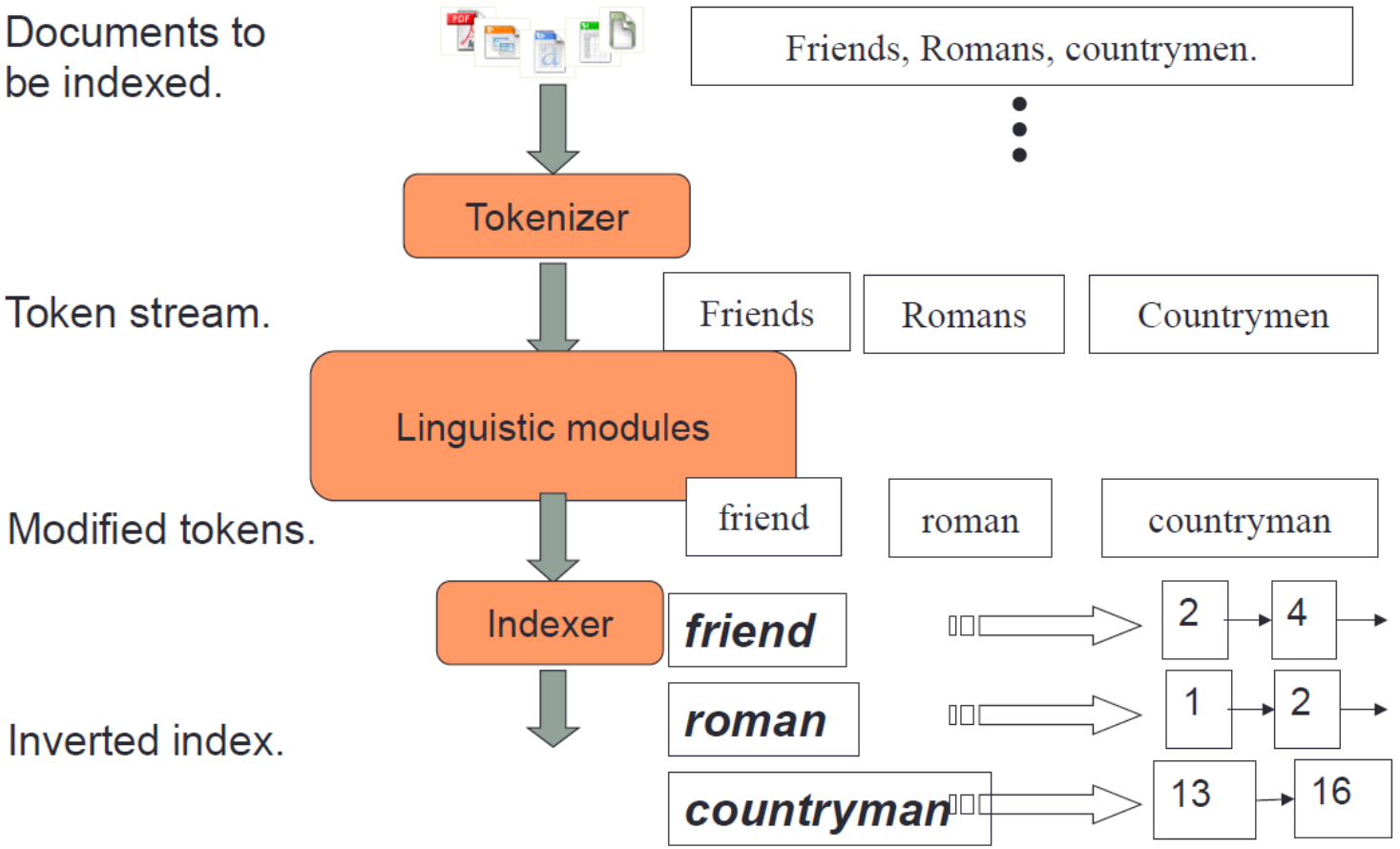
Basics - document

- Document: anything which one may search for, which contains information in different media (text, image, …)
  - This course: text
- Text document = description in a natural language
- Human vs. computer understanding
  - Read the text and understand the meaning
  - A computer cannot (yet) understand meaning as a human being, but can quickly process symbols (strings, words, …).
- Indexing process:
  - Let a computer “read” a text
  - Select the symbols to represent what it believes to be important
  - Create a representation (index) in order to support fast search

Basics – "Reading" a document

- Let us use words as the representation units
- Document parsing
  - Identify document format (text, Word, PDF, …)
  - Identify different text parts (title, text body, …) (note: often separate index for different parts)
  - Go through a text, and recognize the words
    - Tokenization
    - The elements recognized = tokens
  - Statistics for weighting
- Create index structures
- Search: Query words → corresponding documents

Basic indexing pipeline



Tokenization

- Input: "***Friends, Romans and Countrymen***"
- Output: Tokens
  - ***Friends***
  - ***Romans***
  - ***and***
  - ***Countrymen***
- Usually use space and punctuations
- Each such token is now a candidate for an index entry, after further processing

Tokenization: issues

- ***Finland's capital*** $\rightarrow$

  Finland? Finlands? Finland's?

- ***Hewlett-Packard*** $\rightarrow$

  Hewlett and ***Packard*** as two tokens?

  - ***State-of-the-art***: break up hyphenated sequence.
  - co-education ?
  - the hold-him-back-and-drag-him-away-maneuver ?

- ***San Francisco***: one token or two? How do you decide it is one token?

Tokenization: Numbers

- ***3/12/91***
- ***Mar. 12, 1991***
- ***55 B.C.***
- ***B-52***
- ***My PGP key is 324a3df234cb23e***
- ***100.2.86.144***
  - Generally, don't index as text.
  - Will often index "meta-data" separately
    - Creation date, format, etc.

Normalization

- Need to “normalize” terms in indexed text as well as query terms into the same form
  - We want to match ***U.S.A.*** and ***USA***
- We most commonly implicitly define equivalence classes of terms
  - e.g., by deleting periods in a term
  - U.S. → US

Case folding

- Reduce all letters to lower case
  - exception: upper case (in mid-sentence?)
    - e.g., ***General Motors***
    - ***Fed*** vs. ***fed***
    - ***SAIL*** vs. ***sail***
    - ***AIDS*** vs. ***aids***
  - Often best to lower case everything, since users will use lowercase regardless of 'correct' capitalization
  - Problems
    - Simple tokenization: U.S. → U, S
    - U.S. → us
- Question: Other problems in tokenization?

Lemmatization

- Reduce inflectional/variant forms to base form (lemma)
- E.g.,
  - *am, are, is* → *be*
  - *computing* → compute
  - *car, cars, car's, cars'* → *car*
- *the boy's cars are different colors* → *the boy car be different color*
- Lemmatization implies doing "proper" reduction to dictionary headword form

Stemming

- Reduce terms to their "roots"/stems before indexing
- "Stemming" suggest crude affix chopping
  - language dependent
  - e.g., ***automate(s), automatic, automation*** all reduced to ***automat***.

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

Porter's algorithm

- Commonest algorithm for stemming English
  - Results suggest at least as good as other stemming options
- Conventions + 5 phases of reductions
  - phases applied sequentially
  - each phase consists of a set of commands

Porter algorithm

(Porter, M.F., 1980, An algorithm for suffix stripping, *Program*, **14**(3) :130-137)

- Step 1: plurals and past participles
  - SSES -> SS — caresses -> caress
  - (*v*) ING -> — motoring -> motor
- Step 2: adj->n, n->v, n->adj, …
  - (m>0) OUSNESS -> OUS — callousness -> callous
  - (m>0) ATIONAL -> ATE — relational -> relate
- Step 3:
  - (m>0) ICATE -> IC — triplicate -> triplic
- Step 4:
  - (m>1) AL -> — revival -> reviv vital -> vital
  - (m>1) ANCE -> — allowance -> allow
- Step 5:
  - (m>1) E -> — probate -> probat
  - (m > 1 and *d and *L) -> single letter — controll -> control

Other stemmers

- Other stemmers exist, e.g., Lovins stemmer
  http://www.comp.lancs.ac.uk/computing/research/stemming/general/lovins.htm
  - Single-pass, longest suffix removal (about 250 rules)
  - Motivated by Linguistics as well as IR
- Krovetz stemmer (R. Krovetz, 1993: "Viewing morphology as an inference process," in R. Korfhage et al., *Proc. 16th ACM SIGIR Conference*, Pittsburgh, June 27-July 1, 1993; pp. 191-202.)
  - Use a dictionary – if a word is in the dict, no change, otherwise, suffix removal
- Full morphological analysis – at most modest benefits for retrieval
- Do stemming and other normalizations help?
  - Often very mixed results: really help recall for some queries but harm precision on others
- Question:
  - Stemming usually remove suffixes. Can we also remove prefixes?

Example (from Croft et al.'s book)

- Original

Document will describe marketing strategies carried out by U.S. companies for their agricultural chemicals, …

- Porter

document describ market strategi carri compani agricultur chemic …

- Krovetz

document describe marketing strategy carry company agriculture chemical …

Stopwords / Stoplist

- Stopword = word that is not meaning bearer
- Function words do not bear useful information for IR
  of, in, about, with, I, although, …
- Stoplist: contain stopwords, not to be used as index
  - Prepositions: of, in, from, …
  - Articles: the, a, …
  - Pronouns: I, you, …
  - Some adverbs and adjectives: already, appropriate, many, …
  - Some frequent nouns and verbs: document, ask, …
- The removal of stopwords usually improves IR effectiveness in TREC experiments
- But more conservative stoplist for web search
  - "To be or not to be"
- A few "standard" stoplists are commonly used.

Stoplist in Smart (571)

a	all	and	are
a's	allow	another	aren't
able	allows	any	around
about	almost	anybody	as
above	alone	anyhow	aside
according	along	anyone	ask
accordingly	already	anything	asking
across	also	anyway	associated
actually	although	anyways	at
after	always	anywhere	available
afterwards	am	apart	away
again	among	appear	awfully
against	amongst	appreciate	
ain't	an	appropriate	

Discussions

- What are the advantages of filtering out stop words?
  - Discard useless terms that may bring noise
  - Reduce the size of index
- What problems this can create?
  - Difficult to decide on many frequent words: useful in some area but not in some others
  - A too large stoplist may discard useful terms
  - Stopwords in some cases (specific titles) may be useful
  - Silence in retrieval – document cannot be retrieved for this word