CS317

Information Retrieval

Week 02

Muhammad Rafi

February 19, 2021

Term Vocabulary & Posting Lists

Boolean Model

- Information need has to be translated into a Boolean expression which most users find awkward
- The Boolean queries formulated by the users are most often too simplistic
- The Boolean model imposes a binary criterion for deciding relevance
- The question of how to extend the Boolean model to accomodate partial matching and a ranking has attracted considerable attention in the past
- Two extensions of boolean model:
  - Extended Boolean Model
  - Fuzzy Set Model

Extended Boolean Model

- Proximity Search
- Ranked Retrieval
- Example
  - WestLaw

Westlaw – Commercial Systems

- Largest commercial legal search service in terms of the number of paying subscribers
- Over half a million subscribers performing millions of searches a day over tens of terabytes of text data
- The service was started in 1975.
- In 2005, Boolean search (called “Terms and Connectors” by Westlaw) was still the default, and used by a large percentage of users . . .
- . . . although ranked retrieval has been available since 1992.

Westlaw – Commercial Systems

- Information need: Information on the legal theories involved in preventing the disclosure of trade secrets by employees formerly employed by a competing company
- Query: “trade secret” /s disclos! /s prevent /s employe!

Boolean Retrieval Model

- Last Chapter: Simple Boolean retrieval system
- Our assumptions were:
  - We know what a document is.
  - Documents are only the collection of features.
  - We can “machine-read” each document.
- This can be complex in reality.

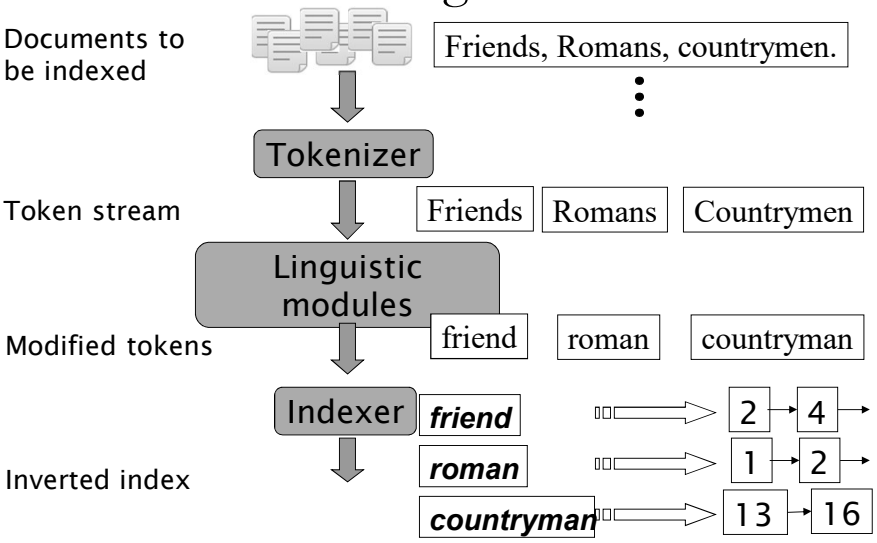
Agenda

- Document Processing

Some Definitions

- Word – A delimited string of characters as it appears in the text.
- Term – A "normalized" word (case, morphology, spelling etc); an equivalence class of words.
- Token – An instance of a word or term occurring in a document.
- Type – The same as a term in most cases: an equivalence class of tokens.

Document Processing



Challenges in Document Processing

- What format is it in?
  - pdf/word/excel/html?
- What language is it in?
- What character set is in use?
  - (CP1252, UTF-8, …)
- Format/Language/Encoding…
- Each of these is a classification problem, which we will study later in the course.

Challenges in Document Processing

- Documents –a general term for IR
- Size of document
  - A file / An e-mail / A blog
  - A group of files.
- Tokenization
  - A process through with documents are parsed and a sequence of characters separated, as a feature for document processing.
  - A token is an instance of a sequence of characters

Challenges in Document Processing

- Tokenization
  - Tokenization process decide when to emit a token.
  - Input: “Friends, Romans and Countrymen”
  - Output: Tokens
    - Friends
    - Romans
    - Countrymen
- Issues in Tokenization
  - Finland’s capital

Challenges in Document Processing

- Issues in Tokenization
  - Finland’s capital
  - Hewlett-Packard
  - co-education
  - San Francisco: one token or two?
  - Numbers
    - 3/20/91
    - Mar. 12, 1991
    - 55 B.C.
    - B-52
    - (800) 234-2333

Challenges in Document Processing

- Issues in Tokenization
  - Languages
    - French
    - German
    - Urdu & Arabic
    - Korean
    - Chinese & Japanese

莎拉波娃现在居住在美国东南部的佛罗里达。今年４月９日，莎拉波娃在美国第一大城市纽约度过了１８岁生日。生日派对上，莎拉波娃露出了甜美的微笑。

استقلت الجزائر في سنة 1962 بعد 132 عاما من الاحتلال الفرنسي.

← START ← → ← →

'Algeria achieved its independence in 1962 after 132 years of French occupation.'

Sec. 2.2.2

Stop words

- With a stop list, you exclude from the dictionary entirely the commonest words. Intuition:
  - They have little semantic content: *the, a, and, to, be*
  - There are a lot of them: ~30% of postings for top 30 words
- But the trend is away from doing this:
  - Good compression techniques (lecture 5) means the space for including stopwords in a system is very small
  - Good query optimization techniques (lecture 7) mean you pay little at query time for including stop words.
  - You need them for:
    - Phrase queries: "King of Denmark"
    - Various song titles, etc.: "Let it be", "To be or not to be"
    - "Relational" queries: "flights to London"

Sec. 2.2.3

Normalization to terms

- We need to "normalize" words in indexed text as well as query words into the same form
  - We want to match ***U.S.A.*** and ***USA***
- Result is terms: a term is a (normalized) word type, which is an entry in our IR system dictionary
- We most commonly implicitly define equivalence classes of terms by, e.g.,
  - deleting periods to form a term
    - ***U.S.A., USA***
  - deleting hyphens to form a term
    - ***anti-discriminatory, antidiscriminatory antidiscriminatory***

Sec. 2.2.3

Normalization: other languages

- Accents: e.g., French ***résumé*** vs. ***resume.***
- Umlauts: e.g., German: ***Tuebingen*** vs. ***Tübingen***
- Cedilla/diacritic
- Most important criterion:
  - How are your users like to write their queries for these words?
- Even in languages that standardly have accents, users often may not type them
  - Often best to normalize to a de-accented term
    - ***Tuebingen, Tübingen, Tubingen*** ⎩ ***Tubingen***

Morphological Analysis

- Inflections: adding a suffix to a word, that doesn't change its grammatical category, such as tenses in verbs (-ing, -ed, -s), plural in nouns (s).
- Derivations - adding a suffix to a word, that changes its grammatical category, such as nation (noun) => national (adjective) => nationalize (verb).

Sec. 2.2.3

Case folding

- Reduce all letters to lower case
  - exception: upper case in mid-sentence?
    - e.g., General Motors
    - Fed vs. fed
    - SAIL vs. sail
  - Often best to lower case everything, since users will use lowercase regardless of 'correct' capitalization…

Sec. 2.2.3

Normalization to terms

- An alternative to equivalence classing is to do asymmetric expansion
- An example of where this may be useful
  - Enter: ***window*** Search: ***window, windows***
  - Enter: ***windows*** Search: ***Windows, windows, window***
  - Enter: ***Windows*** Search: ***Windows***
- Potentially more powerful, but less efficient

Thesauri and soundex

- Do we handle synonyms and homonyms?
  - E.g., by hand-constructed equivalence classes
    - ***car*** = ***automobile*** ***color*** = ***colour***
  - We can rewrite to form equivalence-class terms
    - When the document contains ***automobile***, index it under ***car-automobile*** (and vice-versa)
  - Or we can expand a query
    - When the query contains ***automobile***, look under ***car*** as well
- What about spelling mistakes?
  - One approach is Soundex, which forms equivalence classes of words based on phonetic heuristics

Lemmatization

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

Lemmatization Vs. Stemming

- Lemmatization is the algorithmic process of determining the lemma for a given word.
  - The process may involve complex tasks such as
    - understanding context and determining the part of speech of a word in a sentence (requiring, for example, knowledge of the grammar of a language)
    - it can be a hard task to implement a lemmatizer for a new language.
  - For example, in English, the verb 'to walk' may appear as 'walk', 'walked', 'walks', 'walking'. The base form, 'walk', that one might look up in a dictionary, is called the *lemma* for the word.

Lemmatization Vs. Stemming

- Stemmer:
  - The difference between lemmatization and stemming is that a stemmer operates on a single word ***without* knowledge of the context**
  - Stemmers are typically easier to implement and run faster, and the reduced accuracy may not matter for some applications

Porter Stemmer

- An incoming word is cleaned up in the initialization phase, one prefix trimming phase then takes place and then five suffix trimming phases occur.
- Note: The entire algorithm will not be covered -- we will leave out some obscure rules.

Initialization

- First the word is cleaned up. Converted to lower case only letters or digits are kept.
- F-16 is converted to f16.

Porter Stemming

- Remove prefixes:
  - "kilo", "micro", "milli", "intra", "ultra", "mega", "nano", "pico", "pseudo"
- So megabyte, kilobyte all become “byte”.

Porter Step 1

- Examples:
- Remove “es” from words that end in “sses” or “ies”
  - passes --> pass, cries --> cri
- Remove “s” from words whose next to last letter is not an “s”
  - runs --> run, fuss --> fuss
- If word has a vowel and ends with “eed” remove the “ed”
  - agreed --> agre, freed --> freed
- Replace trailing “y” with an “I” if word has a vowel
  - satisfy --> satisfi, fly --> fli

Porter Step 2

- With what is left, replace any suffix on the left with suffix on the right ...

tional	tion	conditional --> condition
ization	ize	nationalization --> nationalize
iveness	ive	effectiveness --> effective
fulness	ful	usefulness --> useful
ousness	ous	nervousness --> nervous
ousli	ous	nervously --> nervous
entli	ent	fervently --> fervent
iveness	ive	inventiveness --> inventive
biliti	ble	sensibility --> sensible

Porter Step 3

- With what is left, replace any suffix on the left with suffix on the right ...

icate	ic	fabricate --> fabric (*Think about this one)*
ative	--	combativ --> comb (*another good one)*
alize	al	nationalize --> national
iciti	ic	
ical	ic	tropical --> tropic
ful	--	faithful --> faith
iveness	ive	inventiveness --> inventive
ness	--	harness --> har

Porter Step 4

- Remove remaining standard suffixes
  - *al, ance, ence, er, ic, able, ible, ant, ement, ment, ent, sion, tion, ou, ism, ate, iti, ous, ive, ize, ise*

Porter Step 5

- Remove trailing “e” if word does not end in a vowel
  - hinge --> hing
  - free --> free

Porter Stemmer: Experimental Results

Suffix stripping of a vocabulary of 10,000 words

Number of words reduced in	step 1:	3597
	step 2:	766
	step 3:	327
	step 4:	2424
	step 5:	1373
Number of words not reduced:		3650

The resulting vocabulary of stems contained 6370 distinct entries. Thus the suffix stripping process reduced the size of the vocabulary by about one third.

Example

Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Lovins stemmer: such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Paice stemmer: such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

Porter Summary

- Do stemming and other normalizations help?
  - English: very mixed results. Helps recall but harms precision
    - operative (dentistry) ⇒ oper
    - operational (research) ⇒ oper
    - operating (systems) ⇒ oper
  - Definitely useful for Spanish, German, Finnish, …
    - 30% performance gains for Finnish!
- Full morphological analysis – at most modest benefits for retrieval

Implementation Issues

- Inverted Index
  - Lists
  - Hashmap
  - Trees
- SkipList

Problem

- Are the following statements true or false?
  - In a Boolean retrieval system, stemming never lowers precision.
  - In a Boolean retrieval system, stemming never lowers recall.
  - Stemming increases the size of the vocabulary.
  - Stemming should be invoked at indexing time but not while processing a query.

Problem (Solution)

- Are the following statements true or false?
  - In a Boolean retrieval system, stemming never lowers precision. (False)
  - In a Boolean retrieval system, stemming never lowers recall. (True)
  - Stemming increases the size of the vocabulary. (False)
  - Stemming should be invoked at indexing time but not while processing a query. (False)