Preprocessing

COMP3009J: Information Retrieval

Dr. David Lillis (david.lillis@ucd.ie)

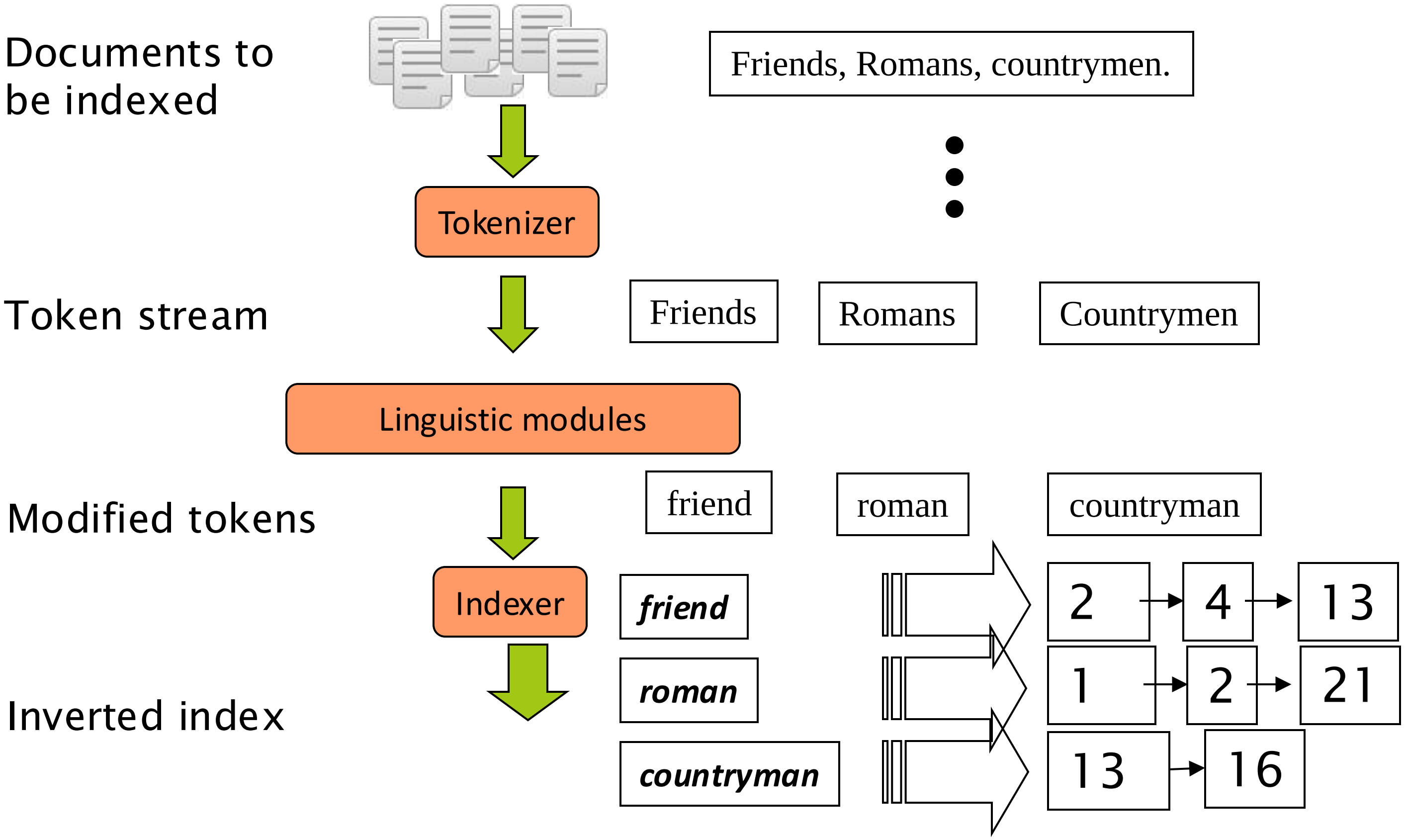
UCD School of Computer Science
Beijing Dublin International College

Some slides adapted from P. Nayak and P. Raghavan

Preprocessing

- **Aim:**
  - Investigate some of the challenges of dealing with natural language when creating index terms.

Remember: creating an Index



Documents to be indexed

Friends, Romans, countrymen.

Tokenizer

Token stream

Friends Romans Countrymen

Linguistic modules

Modified tokens

friend roman countryman

Indexer

Inverted index

friend → 2 → 4 → 13

roman → 1 → 2 → 21

countryman → 13 → 16

1. Split the text into **tokens** (generally words)

2. Tokens must be processed further, according to the language the text is in (more on this **NOW**).

3. Then, we create the inverted index.

Tokenisation (or "Tokenization" if you're American)

- **Input:** "Friends, Romans and Countrymen"
- **Output:** Tokens:
  - *Friends*
  - *Romans*
  - *Countrymen*
- A **token** is an instance of a sequence of characters. In the previous lecture we said they were similar to words, but they are not the same.
- Each such token is now a candidate for storing as an index entry, after **further processing**. We refer to this as **preprocessing** as it occurs before queries are processed by the system.
  - When we store a token in an index, we call it a **term**.
  - **Note:** a term is not always a real word, as we shall see later.
- How can we turn tokens into terms?
  - Libraries are available to deal with most of these situations.

Tokenisation

- Issues in tokenisation:
  - ***Finland's capital*** →

    Finland AND ***s***? ***Finlands***? ***Finland's***?
  - ***Mercedes-Benz*** → ***Mercedes*** and ***Benz*** as two tokens?
    - ***state-of-the-art***: break up hyphenated sequence.
    - ***lowercase***, ***lower-case***, ***lower case*** ?
    - It can be effective to get the user to put in possible hyphens
  - ***San Francisco***: one token or two?
    - How do you decide it is one token?

Numbers

- ***3/20/91 Mar. 12, 1991 20/3/91***
- ***55 B.C.***
- ***B-52***
- ***My PGP key is 324a3df234cb23e***
- ***(800) 234-2333***
  - Often have embedded spaces
  - Older IR systems may not index numbers
    - But often very useful: think about things like looking up error codes/stacktraces on the web
  - Will often index “meta-data” separately
    - Creation date, format, etc.

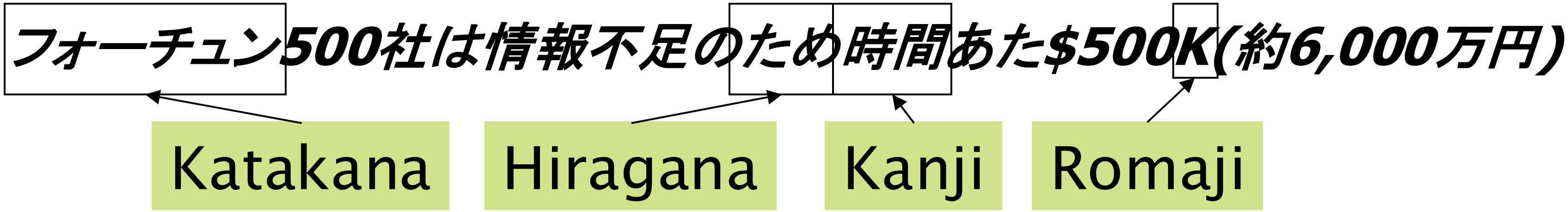
Tokenisation: language issues

- French
  - ***L'ensemble*** → one token or two?
    - ***L*** ? ***L’*** ? ***Le*** ?
    - Want ***l’ensemble*** to match with ***un ensemble***
      - Until at least 2003, it didn’t on Google
        - Internationalisation!

- German noun compounds are not segmented
  - ***Lebensversicherungsgesellschaftsangestellter***
  - ‘life insurance company employee’
  - German retrieval systems benefit greatly from a **compound splitter** module
    - Can give a 15% performance boost for German

Tokenisation: language issues

- Chinese and Japanese have no spaces between words:
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - Not always guaranteed a unique tokenization.
    - e.g. (烘手)机 vs. 烘(手机)
- Further complicated in Japanese, with multiple alphabets intermingled.



フォーチュン500社は情報不足のため時間あた$500K(約6,000万円)

Katakana Hiragana Kanji Romaji

End-user can express query entirely in hiragana!

Tokenisation: language issues

- Arabic (or Hebrew) is basically written right to left, but with certain items like numbers written left to right.
- Words are separated, but letter forms within a word form complex ligatures.

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

← → ← → ← start

- ‘Algeria achieved its independence in 1962 after 132 years of French occupation.’
- With Unicode, the surface presentation is complex, but the stored form is straightforward.

Normalisation

- We may need to “normalise” tokens so that they become terms in the same form.
  - e.g. we want *USA* and *U.S.A.* to match.
- Some common approaches:
  - Changing all tokens to lowercase.
    - Even if something should have uppercase letters in it, users often type in lowercase anyway.
  - Delete full stops to form terms: *USA, U.S.A.* → *usa*
  - *Delete hyphens:*
    anti-discrimininatory, antidiscriminatory → *antidiscriminatory*

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 (i.e. it indexes terms using their sounds rather than their spelling).

Conclusions

- When creating an index from a set of tokens, several challenges are presented.
- These challenges depend on the language that the document collection is written in.
- Several approaches have been proposed to address these problems.
  - For many approaches, a tradeoff is required, as they do not improve retrieval in all circumstances.