Preprocessing: Stopwords, Stemming and Lemmatisation

COMP3009J: Information Retrieval

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

UCD School of Computer Science
Beijing Dublin International College

Preprocessing

- **Aim:**
  - Examine some common commonly-applied preprocessing techniques.
  - Observe some tradeoffs – in “solving” one problem, can we sometimes create others?

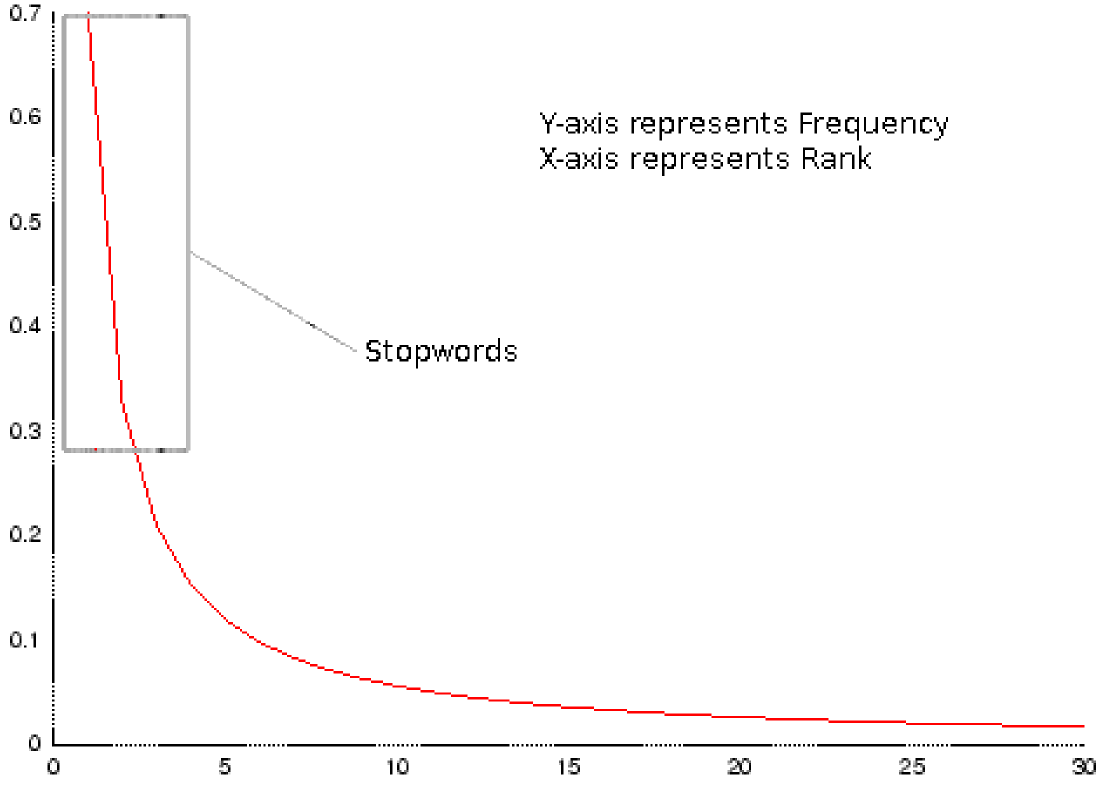
Stopword Removal

- A **stopword** is a commonly occurring term that appears in so many documents that it does not add to the meaning of the document, and appear in most English texts (except for very short ones).
- For example, in English, terms such as *the*, *and*, *of*, etc. tell a reader nothing about what the document is about.
- They are so common that they appear in almost all documents, and of little use when differentiating between documents.
- The fact that they are so common also means that more processing power is required to deal with these terms than others that are less common.
- This additional processing, combined with their relative lack of usefulness means that they are often removed from the documents at the preprocessing stage.

Stopword Removal

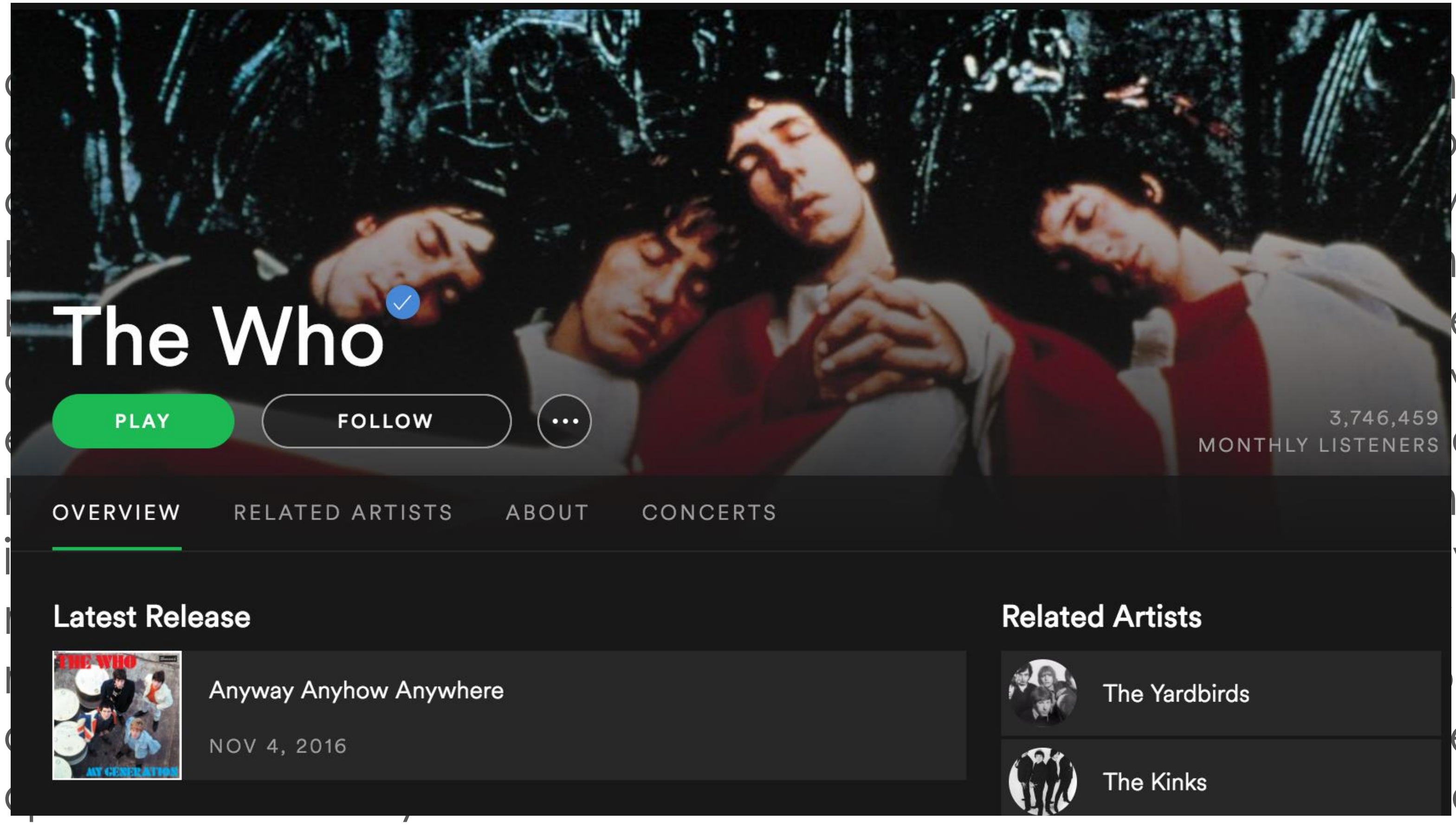
- Stopwords can be estimated using **Zipf's Law**.
- George Kingsley Zipf analysed the statistical occurrence of words (terms) in text.
- He noted that while only a few terms are used very often, more are used only rarely.
- If we rank terms from the most commonly used to the least commonly-used in a large text collection, the second term is approximately ½ as common as the first, the third is approximately ⅓ as common as the first, etc.
- We use this to identify what terms count as stopwords in various languages.

Stopword Removal: Zipf's Law



Y-axis represents Frequency
X-axis represents Rank

a about above across adj after again against all almost alone along also although always am among an and another any anybody anyone anything anywhere apart are around as aside at away be because been before behind being below besides between beyond both but by can cannot could deep did do does doing done down downwards during each either else enough etc even ever every everybody everyone except far few for forth from get gets got had hardly has have having her here herself him himself his how however i if in indeed instead into inward is it its itself just kept many maybe might mine more most mostly much must myself near neither next no nobody none nor not nothing nowhere of off often on only onto or other others ought our ours out outside over own p per please plus pp quite rather really said seem self selves several shall she should since so some somebody somewhat still such than that the their theirs them themselves then there therefore these they this thorough thoroughly those through thus to together too toward towards under until up upon v very was well were what whatever when whenever where whether which while who whom whose will with within without would yet young your yourself



The Who

3,746,459 MONTHLY LISTENERS

Latest Release

Anyway Anyhow Anywhere

NOV 4, 2016

Related Artists

The Yardbirds

The Kinks

e along
dy
ay be
beyond
e down
very
ot had
owever i
ybe
next no
nto or
e plus pp
d since so

some somebody somewhat still such than that **the** their theirs them themselves then there therefore these they this thorough thoroughly those through thus to together too toward towards under until up upon v very was well were what whatever when whenever where whether which while **who** whom whose will with within without would yet young your yourself

Stopword Removal

- There is a **tradeoff** required when using stopwords.
- The trend is away from removing them:
  - Good compression means that space required is small.
  - Good optimisation means processing them takes little time.
- Some queries will become impossible if all stopwords are removed.
  - Phrases: “King **of** Denmark”
  - Song titles, etc. “Let **it be**”, “**To be or not to be**”
  - Relational queries: “Flights **to** London”
- Some solutions:
  - Don’t remove stopwords. Use all words as terms.
  - Recognise when combinations of stopwords are meaningful, and include these as terms in the index.

Stemming

- Many terms in natural language appear different to a computer, but represent the same concept.
- In English, terms such as “connect”, “connecting”, “connects”, “connected”, etc. represent a similar concept.
- Many IR systems believe that a search for “computing” should also include documents that contain the word “computer”, for example.
- **Stemming** is the process that maps these terms to a **common root**.
  - e.g. “computing”, “computer” → “comput”
  - Note: a stem does not need to be a real word.
- The end of a term is called a **suffix** (-er, -s, -ing, -ed, etc.)
- Hence stemming is also known as **suffix stripping**.

Stemming

- While there have been many algorithms to accomplish stemming, the most famous (for English text) is **Porter's Stemming Algorithm**.
- The rules that the follows are detailed in:
  - Martin Porter, An Algorithm for Suffix Stripping, Program, 14 no. 4, pp. 130-137, July 1980.
- Numerous implementations are available at:
  - http://tartarus.org/~martin/PorterStemmer
- It should be noted that Porter's stemming algorithm is not necessarily the most accurate, though it is the most widely used. Snowball and Porter2 are often considered to be more effective.

Stemming Algorithms

- Stemming Example: each of the following news headlines include words with a common stem:
  - accept: FAI decides to **accept** rescue deal.
  - acceptable: FAI deems rescue deal **acceptable**.
  - acceptance: Owens pleased after rescue deal **acceptance**.
  - accepted: The FAI has **accepted** the government rescue deal.
  - accepts: FAI **accepts** rescue deal.
- Clearly, these would be relevant to the same query.
  - The stem for all of these is "**accept**".
- If stemming is applied to the documents, it is also applied to queries, so a user can find these documents using any term related to the stem **accept** (accept, acceptable, acceptance, accepted, accepts, etc.)

Stemming: Problems

- Sometimes suffixes can be removed from words that are **not related**, but that end up being the same afterwards.
- This is known as **overstemming**.
- Stemming is a very useful process to help words of similar meanings to be matched, but it is not a perfect process.
- **Example:** consider the words “general” and “generate”
- Both stem to "gener", even though they are not related.
- Using a dictionary can help avoid this type of problem

Stemming: Problems

- Although not specifically a problem with stemming, there is also an issue with words that do not have the same meaning, but are spelled the same (known as **homographs**).
    - At the Battle of Waterloo, Napoleon's forces were **routed**.
        - The army was heavily defeated in battle.
    - The cars were **routed** off the motorway.
        - The cars were sent in a different direction.
- It is very difficult to tell the difference between these without using Natural Language Processing, which is a much slower process that often needs to see the word in context.

Lemmatisation

- Lemmatisation is a Natural Language Processing (NLP) technique for converting words into **lemmas** (i.e. base words found in dictionaries).
  - Unlike a **stem**, a **lemma** is always a real word.
- This is slower than stemming, but is usually more effective, because it can deal with words that stemming can't handle.
- To find the right lemma, a lemmatiser will often need to know the **part of speech** that the word is being used as (verb, noun, etc.). This requires more text analysis than stemming.

Lemmatisation vs. Stemming

Verbs

Word	Lemma	Stem
running	run	run
runs	run	run
ran	run	ran
be	be	be
am	be	am
are	be	ar
brought	bring	brought
brings	bring	bring

Nouns

Word	Lemma	Stem
wolves	wolf	wolv
wolf	wolf	wolf
church	church	church
churches	church	church
mouse	mouse	mous
mice	mouse	mice
radius	radius	radiu
radii	radius	radii