Lecture 1b: Text, terms, and bags of words

Trevor Cohn
(based on slides by William Webber)

COMP90042, 2015, Semester 1
Corpus, document, term

- Body of text referred to as corpus
- Corpus regarded as a collection of discrete documents
- Document regarded as a collection of discrete terms
- Term not necessarily a “word” (or token) in natural language
Complexities

- Corpus may be open-ended, poorly defined, growing
- May be in a strange format or formats (pdfs, html, text, spreadsheets)
- What language(s) and character sets?
- Document boundaries may be unclear:
  - Non-trivial dependencies between documents (e.g. email in a thread of emails)
  - Internal subdivisions (e.g. chapter in a book; paragraphs in chapter)
- Terms may be “typed”, have internal structure
  - What are the terms in the email headers:

  From: "Peter Pan" <peter@pan.com>
  Date: Sun, 02 Mar 2014 23:26:33 +0000
What is a document?

Question of granularity...

- Is it a file on disk?
- Linked group of files?
- Emails, but what about threading and attachments?

Generally a corpus specific question, here we will assume these questions have been resolved, and focus on subsequent processing.
Tokenising: What is a word?

- Split document up into “tokens”
- Input: “To be, or not to be ...”
- Output:
  - To
  - be
  - or
  - ...
- A token is an instance of a word, stripped of punctuation
- Tokens form the candidates for subsequent search
- In English and other similar scripts:
  - Strip formatting (e.g. of HTML)
  - Strip punctuation
  - Break at whitespace
- What about ideographic languages (e.g. CJK)?
  Compounding?
Tokenising complexities

- Even for English, it’s not completely straightforward
- *Peter’s yard* –
  *Peter* and ’s or s, or just *Peter’s*?
- shouldn’t –
  is the ’ punctuation? Or *should* and *n’t*
  remove internal punctuation?
- *one-day-at-a-time, well-respected* –
  splitting makes sense here
- *merry-go-round* –
  splitting makes less sense
- *New Zealand* – merge together into one token?
Tokenising complexities – numbers, dates etc

- 03 8344 1700
- 3/5/2015, May 3rd ’15
- 2020 A.D.
- V2.0
- “FYI #MyHeartAge is 61 . . . oh dear. Poor old ticker.”
  @stephenfry
- also, URLs, emails and other specific markup

Must deal with internal spaces and punctuation.
- common approach is to discard these tokens
- often useful though, for specific queries (e.g., software error codes)
- or can deal with them as n-grams / phrases

Dates often indexed as metadata, e.g., date of page creation.
Token processing

- Case folding: in bicameral (two-case) scripts (e.g. Latin, Greek), reduce all letters to one (e.g. lower) case

  \[\text{United States} \rightarrow \text{united states}\]

- Stemming: in inflected languages, reduce words to base forms:

  \[\text{stemming, stemmed, stems} \rightarrow \text{stem}\]

- Stopping: remove common words that do not carry meaning:

  \[\text{the, a, and, of}\]

- Other tokens may be suppressed (e.g. long numbers)

Notes

- All of the above are optional; pros and cons for each
- Different languages present different issues
Stemming vs. Lemmatisation

Collapse multiple forms of same word to single form

- Lemmatisation reduces inflected forms to dictionary entry *lemma*
  - is, are, was, were, will → be
  - sit, sat, sits → sit
- Requires a dictionary and set of processing rules (e.g., for suffixes)
- Stemming is a simpler, just affix chopping
  - automate, automatic, automation → automat
- Often doesn’t result in a dictionary entry

Overall lead to better retrieval performance (more matches), but can conflate different concepts
Porter Stemmer

Most popular stemmer for English

- Apply reductions in several stages
- Attempt to match longest suffix possible
- List of pre-defined rules, e.g.,
  - SSES $\rightarrow$ SS
  - IES $\rightarrow$ I
  - ED $\rightarrow$
  - ING $\rightarrow$
  - IVITI $\rightarrow$ IVE
- some rules e.g., $(m > 1)$ EED $\rightarrow$ EE
  select for length, agreed $\rightarrow$ agree but not applied to feed.

E.g., Whether tis nobler in the mind to suffer the slings and arrows of outrageous fortune $\rightarrow$ Whether ti nobler in the mind to suffer the sling and arrow of outrag fortun
Welcome to COMP90042!

Students taking this subject must have taken the subject COMP30018. All student work — including projects — must be programmed in Python. Email tcohn@unimelb.edu.au with questions.
Welcome to COMP90042!

Students taking this subject must have taken the subject COMP30018. All student work—including projects—must be programmed in Python. Email tcohn@unimelb.edu.au with questions.
Welcome to COMP90042

Students taking this subject must have taken the subject COMP30018. All student work including projects must be programmed in Python. Email tcohn@unimelb.edu.au with questions

- Strip formatting
- Strip punctuation
Welcome to COMP90042 Students taking this subject must have taken the subject COMP30018. All student work including projects must be programmed in Python. Email tcohn@unimelb.edu.au with questions.

- Strip formatting
- Strip punctuation
- Break at whitespace
welcome to comp90042 students taking this subject must have taken the subject comp30018 all student work including projects must be programmed in python email tcohn unimelb edu au with questions

- Strip formatting
- Strip punctuation
- Break at whitespace
- Case fold
Processing example

welcome to comp90042 student
take this subject must have
take the subject comp30018
all student work include
project must be program in
python email tcohn unimelb
edu au with question

- Strip formatting
- Strip punctuation
- Break at whitespace
- Case fold
- Stem
Processing example

Welcome comp90042 student
Take subject must
take subject comp30018
student work include
project must program
python email tcohn unimelb edu au question

- Strip formatting
- Strip punctuation
- Break at whitespace
- Case fold
- Stem
- Stop
Processing example

- Strip formatting
- Strip punctuation
- Break at whitespace
- Case fold
- Stem
- Stop
- Bag of words
Term occurrences

Real-world vocabulary not closed, fixed (not just “dictionary words”), due to:

▶ Proper names
▶ Abbreviations, acronyms
▶ Serial numbers, URLs, other identifiers
▶ Terms from other languages
▶ Misspellings
▶ Slang
▶ . . .

One rule of thumb for processing web documents:

*The size of your vocabulary approximately equals the number of documents you have*
Distribution of term frequencies

- Distribution of term frequencies in collection (either total count, or document count) follows a power law (Zipfian, long-tailed) distribution
- Rank terms (or any other items) by decreasing frequency
- Zipf’s law: The frequency $f$ of the term at rank $r$ is proportional to $r^{-b}$, with $b \approx 1$; i.e.:
\[ f \sim r^{-b} \]  
- Zipfian distributions will look approximately straight in log–log graphs
Distribution of term frequencies (cont.)

- Collection of 804,414 news articles (RCV1v2)
- Has (stemmed, stopped, case-folded) vocabulary of 288,062 terms. (The 20-volume OED lists 171,146 words in current use.)
- Count term frequency by number of documents a term appears in
Distribution of term frequencies in Brown Corpus

- Regular graph almost perfect L shape.
Distribution of term frequencies in Brown Corpus

- Regular graph almost perfect L shape.
- In log-log graph, a “bent” straight line (typical for such distributions)
Consequences of Zipf’s law

Infrequent terms

- Most of the vocabulary will occur very infrequently
- For RCV1v2, 40% of vocabulary occurs only in single document
- Includes “terms” like t04301, hrebenciuc, tluszczoweg
- For some applications, such terms are uninformative

Frequent terms

- A few terms will appear in much of the collection (even if stopping is applied).
- For RCV1v2, year appears in 46% of documents, percent in 37%, and million in 35%
- For some applications, such terms are only weakly informative
Looking back and forward

- The “Bag of Words” representation of a document is widely used, and will be used for most of the IR component of the course.
- Vocabularies are open-ended, and term frequencies long-tailed (Zipfian).
- Further modelling step required to make BOW useful.
- We will look at several models of IR building on this text and document representations.
Further reading

▶ Chap 2, secs 2.1 & 2.2 of Manning, Raghavan, and Schutze, *Introduction to Information Retrieval* (on tokenization and term normalization)

▶ Introduction to Manning and Schutze, *Foundations of Statistical Natural Language Processing* (§1.4 on token and type distributions Zipf’s law; §4.2 on tokenisation)

▶ The Porter Stemmer, a standard English stemming algorithm
http://tartarus.org/martin/PorterStemmer/