Information Retrieval (Handout Second Part

Computer Science Tripos Part II

Simone Teufel

Natural Language and Information Processing (NLIP) Group
UNIVERSITY OF CAMBRIDGE
Simone.Teufel@cl.cam.ac.uk

Lent 2012

1. Lecture 1: Introduction
   - Motivation
     - Larger Context
     - Terminology and Definitions
   - Jumping right in
   - History of IR
   - Post-Lecture Exercises
     - Weighting
     - Document–document matrix

Why study IR?

- Many reasons, but if you want a one-word answer:

Why study IR?

- Many reasons, but if you want a one-word answer:

Google . . .

- . . . examines billions of web pages
- . . . returns results in less than half a second
- . . . processes hundreds of Million of queries a day
- . . . valued at gazillions of dollars by the public market

"92% of Internet users say the Internet is a good place for getting everyday information." (Pew Internet Survey, 2004)

How does Google work?

- Only Google know, but ...
- Uses hundreds of thousands of machines
- Uses some sophisticated computer science (efficient storage and searching of large datasets)
- Uses an innovative ranking algorithm (based on the hypertext structure of the web)

How does Google work?

- Underlying Google is basic IR technology
- The Web is indexed
    - an index links terms with pages
- A user's information need is represented as a query
- Queries are matched against web pages
    - Google attempts to return pages which are relevant to the information need

Document Retrieval

Def. of IR:

"Information Retrieval is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers)"
(Manning et al. 2008)

- IR is often used to mean Document Retrieval
- Primary task of an IR system:
  retrieve documents with content that is relevant to a user's information need

IR is more than Web search

- IR is much older than the Web (1950s –)
- The Web has some unique characteristics which make it a special case
- Other "real" applications:
  - Searching literature databases
  - Patent search
  - Information analysts' searches
  - Volume of information stored electronically is growing at ever faster rates
- IR deals with tasks other than searching:
  - categorising information
  - filtering it
  - translating it
  - summarising it
  - drawing conclusions from it
  - ...

Biomedical Information

- Biomedical literature is growing at a startling rate
  - Around 1,000,000 new articles are added to Medline each year
- Tasks:
  - literature search
  - creation and maintenance of biological databases
  - knowledge discovery from text mining

Document Retrieval

- Representation/Indexing
  - Bag of words? stop words, upper/lower case, . . . ?
  - Choice of query language
  - Storing the documents, building the index

Query: "Laws of thought"

Amongst the matches:
"And you thought law enforcement was boring?"
"Savannah NOW: Local News - Mother-in-law thought mechanic was a ..."

- Retrieval Model
  - Is a document relevant to the query?
  - Models of IR: Boolean, Vector-space, Probabilistic, Language Model
  - Efficient algorithms for searching large datasets

What IR is not

- An IR system is not a Database Management System
- A DBMS stores and processes well-defined data
- A search in a DBMS is exact / deterministic
- Search in an IR system is probabilistic
  - inherent uncertainty exists at all stages of IR: information need, formulating query, searching

Uncertainty in Document Retrieval

"The normal presumption in document retrieval is that the user wishes to find out about a topic or subject and is thus, while interested in data or facts, not yet in a position to specify precisely what data or facts are required."
(Sparck Jones and Willett, eds., p.85)

A Simple Retrieval Model

- Bag of Words approach
  - A document is represented as consisting of words as independent units
  - Word order is ignored
  - Syntactic structure is ignored
  - ...
- Relevance is determined by comparing the words in the document with the words in a query
- Simple approach has been very effective

Vector Space Model

- Provides a ranking of documents with respect to a query
- Documents and queries are vectors in a multi-dimensional information space
- Key questions:
  - What forms the dimensions of the space?
    - terms, concepts, ...
  - How is magnitude along a dimension measured?
  - How are document and query vectors compared?

Coordinate Matching

- Document relevance measured by the number of query terms appearing in a document
- Terms provide the dimensions
  - Large vocabulary $\Rightarrow$ high dimensional space
- Length along a dimension is either 0 or 1
- Similarity measure is the dot-product of the query and document vectors

Simple Example

- Documents: $\boxed{\text{d1: Australia collapse as Hoggard takes 6 wickets}}$
  $\boxed{\text{d2: Pietersen's century puts Australia on back foot}}$
- Query: $\boxed{\text{q1: Hoggard, Australia, wickets}}$
- Query–document similarity:

$$\text{Term vocab.}: \begin{pmatrix} \text{England} \\ \text{Australia} \\ \text{Pietersen} \\ \text{Hoggard} \\ \text{run} \\ \text{wicket} \\ \text{catch} \\ \text{century} \\ \text{collapse} \end{pmatrix} \quad q1{\cdot}d1 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = 3 \quad q1{\cdot}d2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = 1$$

Term Frequency (TF)

- Coordinate matching does not consider the frequency of query terms in documents

d1: Australia collapsed as Hoggard took 6 wickets. Flintoff praised Hoggard for his excellent line and length.

q1: Hoggard, Australia, wickets

$$\text{Term vocab.}: \begin{pmatrix} \text{England} \\ \text{Australia} \\ \text{Pietersen} \\ \text{Hoggard} \\ \text{run} \\ \text{wicket} \\ \text{catch} \\ \text{century} \\ \text{collapse} \end{pmatrix} \quad q1 \cdot d1 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = 4$$

Term Frequency (TF)

- Coordinate matching does not consider the frequency of query terms in documents

> d2: Flintoff took the wicket of Australia's Ponting, to give him 2 wickets for the innings and 5 wickets for the match.

> q1: Hoggard, Australia, wickets

$$\text{Term vocab.}: \begin{pmatrix} \text{England} \\ \text{Australia} \\ \text{Pietersen} \\ \text{Hoggard} \\ \text{run} \\ \text{wicket} \\ \text{catch} \\ \text{century} \\ \text{collapse} \end{pmatrix} \quad q1 \cdot d2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 3 \\ 0 \\ 0 \\ 0 \end{pmatrix} = 4$$

Inverse Document Frequency

- TF-weighted matching does not consider the number of documents query terms appear in
- Assume wicket appears in 100 documents in total, Hoggard in 5, and Australia in 10 (ignoring IDF of other terms)

d1: Australia collapsed as Hoggard took 6 wickets. Flintoff praised Hoggard for his excellent line and length.
d2: Flintoff took the wicket of Australia's Ponting, to give him 2 wickets for the innings and 5 wickets for the match.

q1: Hoggard, Australia, wickets

Inverse Document Frequency

$$q1 \cdot d1 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \frac{1}{10} \\ 0 \\ \frac{2}{5} \\ 0 \\ \frac{1}{100} \\ 0 \\ 0 \\ 1 \end{pmatrix} = 0.411 \qquad q1 \cdot d2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \frac{1}{10} \\ 0 \\ \frac{0}{5} \\ 0 \\ \frac{3}{100} \\ 0 \\ 0 \\ 0 \end{pmatrix} = 0.13$$

Document Length

- Terms in documents can have high term frequencies simply because the document is long
- Normalise similarity measure, M, by Euclidean length:

$$M(Q, D) = \frac{Q \cdot D}{|Q||D|}$$

Vector Space Similarity

- The terms in the query vector and document vector are weighted:
  $Q \cdot D = \sum_t w_{Q,t} \cdot w_{D,t}$
- $w_{D,t} = \text{TF} \times \text{IDF}$
- Vector of weights determines position of document in the information space

Vector Space Similarity

$$
\begin{aligned}
M(Q,D) &= \frac{Q \cdot D}{|Q||D|} \\
&= \frac{1}{|Q||D|} \sum_t w_{Q,t} \cdot w_{D,t}
\end{aligned}
$$

$$
\textit{where } |D| = \sqrt{\sum_t w_{D,t}^2}
$$

$$
= \textit{cosine}(Q, D)
$$

- Similarity measure is the cosine of the angle between the query and document vectors

Remarks

- TF is typically some monotonically increasing function of term frequency (similarly for IDF)
- TF-IDF scheme determines units of each dimension in the information space
- Many variants for calculating TF · IDF exist
  (Salton and Buckley, 1988, in Sparck-Jones and Willett, eds.)
- Alternative similarity measures to *Cosine* exist
- Vector Space models perform extremely well for general document collections

Language Understanding?

- Want a system which "understands" documents and query and matches them?
  - use semantic representation and logical inference
- Until recently such technology was not robust / did not scale to large unrestricted text collections
- But:
  - useful for restricted domains
  - now used for some large-scale tasks (QA, IE)
- Is a "deep" approach appropriate for document retrieval?
  - Powerset (Natural Language Search) think so (see `www.powerset.com`)

Other Topics

- Multimedia IR (images, sound, ...)
  - but text can be of different types (web pages, e-mails, ...)
- User-system interaction (HCI)
- Browsing

Evaluation

- IR has largely been treated as an empirical, or engineering, task
- Evaluation has played an important role in the development of IR
- DARPA/NIST Text Retrieval Conference (TREC)
    - began in 1992
    - has many participants
    - uses large text databases
    - considers many tasks in addition to document retrieval

IR in One Sentence

> "Indexing, retrieving and organizing text by probabilistic or statistical techniques that reflect semantics without actually understanding' '
>
> (James Allan, Umass)

Brief History of IR

- 1960s
  - development of basic techniques in automated indexing and searching
- 1970s
  - Development of statistical methods / vector space models
  - Split from NLP/AI
  - Operational Boolean systems
- 1980s
  - Increased computing power
  - Spread of operational systems

Brief History of IR

- 1990s and 2000s
  - Large-scale full text IR systems for retrieval and filtering
  - Dominance of statistical ranking approaches
  - Web search
  - Multimedia and multilingual applications
  - Question Answering
  - TREC evaluations

Reading List

- Course book
  - Introduction to Information Retrieval
    Manning, Raghavan, Schütze
    `http://nlp.stanford.edu/IR-book/html/htmledition/irbook`
- Supplementary books (not required reading)
  - Modern Information Retrieval, Baeza-Yates & Ribeiro-Neto
  - Readings in Information Retrieval, Sparck Jones & Willett eds.
  - Managing Gigabytes, Witten, Moffat & Bell
  - Information Retrieval, van Rijsbergen
    available online:
    `http://www.dcs.gla.ac.uk/Keith/Preface.html`
- Research papers and individual chapters, e.g., from Manning and Schuetze, Foundations of Statistical Natural Language Processing, 1999, MIT press.

Post-Lecture Exercise 1

Googlewhack is a game started in 2002. The task is to find a pair of search terms that return exactly one document in a Google search.

- anxiousness scheduler
- squirreling dervishes

Deceptively hard! Spend some time trying to find a new googlewhack – it will give you an idea what kinds of words might qualify, and why this is so hard.

Post-Lecture Exercise 2

Perform a search for a rare piece of information, e.g. the following information need:

> I read somewhere that (several hundred years ago) a member of the English aristocraty (forgot who) started speaking extremely late in life (3 or 4 years old), but his first statement was a perfectly formed, complicated sentence. He said this sentence after a lady spilt hot coffee over his legs.

Try to find the name of the artistocrat and the quote.

Document Retrieval

- Retrieve documents with content that is relevant to a user's information need
- Document set is fixed
  (size can vary from 10s of documents to billions)
- Information need is not fixed (ad-hoc retrieval)

Goal:

- Documents relevant to query should be returned
- documents not relevant to query should not be returned

Some Terminology

- Document: an item which may satisfy the user's information need
  - task specific: web pages, news reports, emails, . . .
- Query: representation of user's information need
  - initial query formulated by user . . .
  - transformed into final query used for search
- Term: any word or phrase that can serve as a link to a document

Document and Text Representation: the Index

- Manually searching a book for a desired topic is possible, but tedious and time-consuming
  - indexes help a reader quickly locate a topic
- Exhaustive automatic search of very large document collections is expensive
  - indexes greatly speed-up the search process
- Indexing:
  - the process of building the index
    - inverted file, signature file, . . .
  - deciding what will be used to represent the documents
    - need to facilitate good matching of queries and relevant documents

Inverted files

Doc 1: Except Russia and Mexico no country had had the decency to come to the rescue of the government.

Doc 2: It was a dark and stormy night in the country manor. The time was past midnight.

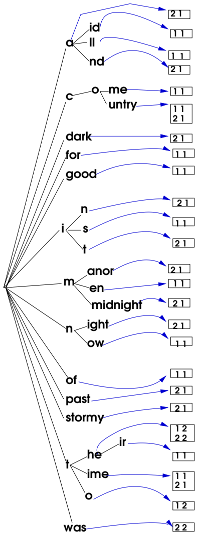
Term	Doc no	Freq	Offset
a	2	1	2
and	1	1	2
and	2	1	4
come	1	1	11
country	1	1	5
country	2	1	9
dark	2	1	3
decency	1	1	9
except	1	1	0
government	1	1	17
had	1	2	6,7
in	2	1	7
it	2	1	0
manor	2	1	10
mexico	1	1	3
midnight	2	1	17
night	2	1	6
no	1	1	4
of	1	1	15
past	2	1	15
rescue	1	1	14
russia	1	1	1
stormy	2	1	5
the	1	2	8,13
the	2	2	8,12
time	2	1	14
to	1	2	10,12
was	2	2	16

Building inverted files

Information kept for each term:

- Document ID where this term occurs
- Frequency of occurrence of this term in each document
- Possibly: Offset of this term in document (why?)
- Note: post-lecture exercise

Corresponding Trie



Manual Indexing

- Controlled vocabularies can be used to determine index terms
- Examples: MeSH, Library of Congress, Gene Ontology, ...
- e.g. Decubitus Ulcer could also be referred to using *Bedsore, Pressure Ulcer, Pressure Sore* ...
  - MeSH: Bacterial Infections -surgery; Cross Infection -surgery; Decubitus Ulcer -complications; ...
- Single term can describe an ambiguous concept
- Human indexers determine what the important concepts are, and what terms can denote those concepts
- Ontologies (e.g., MeSH, GO) organise concepts hierarchically
  - can contain many relations between concepts, e.g. hyponymy, meronymy
  - Structure of ontology can be used to aid search (specificity)

Automatic Indexing

- Manual creation of controlled vocabulary, and maintenance of associated document collection, is very expensive
- Automatic indexing program decides which words or phrases to use as terms from the documents themselves
- Program may even determine concepts and synonymous terms automatically (automatic thesaurus construction)
- Cranfield experiments in the 60s (Cleverdon papers in Sparck Jones and Willett): automatic indexing can be at least as effective as manual indexing with controlled vocabularies
- perhaps counter-intuitive
- general message: much can be achieved with “shallow” content representations

Document Representation

- Represent a document using index terms – what should these be?
- Example 1: "The colinearity of genes in Hox clusters suggests a role for chromosome structure in gene regulation."
  - should "gene" and "genes" be separate terms?
  - should "the", "of", "in", "a", 'for", "in" be terms?
  - should "chromosome structure", "gene regulation", "Hox clusters" be single terms?
- Example 2: "By using retinoic acid (RA) to induce regulated expression of Hoxb genes in mouse embryonic stem (ES) cells, . . ."
  - should "regulation" and "regulated" be separate terms?
  - should "retinoic acid" and "RA" be terms?
  - should "embryonic stem (ES) cells", 'embryonic stem cells", "stem cells", "ES cells" be single terms?

Tokenisation

- Tokenisation: dividing a character stream into a sequence of distinct word forms (tokens)
- Just separate on white-space?
  - end-of-sentence punctuation:
    "role for chromosome structure in gene regulation."
    "Apple Computer, Inc."
  - bracketing: "By using retinoic acid (RA) ..."
  - hyphenation: "By using this state-of-the-art technique ..."
  - apostrophes: "The biologist's hypothesis doesn't imply ..."
  - slashes: "The body has a potassium/sodium mixture ..."
  - ...
- Languages other than English may need additional processing, e.g. segmentation for Chinese

Stop Words

- A stop word is a high-frequency word which is not useful for distinguishing between documents
- Some of the PubMed stop words:
  a, did, it, perhaps, these, about, do, its, quite, they, again, does, itself, rather, this, all, done, just
- Substantially reduces the size of an inverted file index
  - Statistics for TREC documents: (Witten et al.)
    - 33 terms appear in more than 38% of documents
    - 33 terms account for 30% of all term appearances
    - and acount for 11% of pointers in inverted file
- Use of stop words can be problematic
  - Stop words can be names ("The Who"), or frequent words with other meanings (*may, can, will*)

Terms and Equivalence Classes

- Can be useful to put tokens into equivalance classes, and treat a group of terms as the same term
  - reduces size of index
  - may lead to improved retrieval; e.g. if query is {*gene*, *regulation*} may help to retrieve pages containing *regulate*, *regulated*, *regulates*, ...
  - the combined frequencies of class terms may better reflect the content than the individual frequencies

Morphology and Stemming

- Stem: the core of a word (its main morpheme) to which inflectional and derivational morphology applies
  - inflectional morphology deals with such things as plurality and tense:
    employ → *employed*, *employs*, *employing*, ...
  - derivational morphology deals with obtaining nouns from verbs, adjectives and verbs from nouns, etc.:
    fool → *foolish*, *advert* → *advertise*, ...
- Stemming attempts to remove inflectional (and some) derivational morphology
- Lemmatisation just attempts to remove inflectional morphology
- Morphology is a serious issue for e.g. Arabic, Turkish

Stemming: the Porter stemmer

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

- Removal of suffixes without a stem dictionary, only with a suffix dictionary
- Terms with a common stem have similar meanings:

CONNECT
CONNECTED
CONNECTING
CONNECTION
CONNECTIONS

- Deals with inflectional and derivational morphology
- Conflates relate — relativity — relationship
- Root changes (deceive/deception, resume/resumption) aren't dealt with, but these are rare

Stemming: Representation of a word

$$[C]\ (VC)\{m\}[V]$$

C	one or more adjacent consonants
V	one or more adjacent vowels
[ ]	optionality
( )	group operator
{x}	repetition x times
m	the "measure" of a word

shoe	$[sh]_C[oe]_V$	m=0
Mississippi	$[M]_C([i]_V[ss]_C)([i]_V[ss]_C)([i]_V[pp]_C)[i]_V$	m=3
ears	$([ea]_V[rs]_C)$	m=1

Notation: measure *m* is calculated on the word **excluding** the suffix of the rule under consideration

Porter stemmer: rules and conditions

Rules in one block are run through in top-to-bottom order; when a condition is met, execute rule and jump to next block

Rules express criteria under which suffix may be removed from a word to leave a valid stem: (condition) S1 $\rightarrow$ S2

Possible conditions:

- constraining the measure:
  (m > 1) EMENT $\rightarrow \epsilon$ (ϵ is the empty string)
- constraining the shape of the word piece:
  - *S – the stem ends with S
  - *v* – the stem contains a vowel
  - *d – the stem ends with a double consonant (e.g. -TT, -SS).
  - *o – the stem ends cvc, where the second c is not W, X or Y (e.g. -WIL, -HOP)
- expressions with AND, OR and NOT:
  - (m>1 AND (*S OR *T)) – a stem with m> 1 ending in S or T

Porter stemmer: selected rules

SSES → SS
IES → I
SS → SS
S →

caresses → caress

cares → care

(m>0) EED → EE

feed → feed

agreed → agree

BUT: freed, succeed

Porter Stemmer: selected rules

$\boxed{\text{(*v*) ED} \rightarrow}$

plastered → plaster

bled → bled

Porter stemmer: the algorithm

Step 1: plurals and past participles

Step 1a

SSES	$\rightarrow$ SS	caresses	$\rightarrow$ caress
IES	$\rightarrow$ I	ponies	$\rightarrow$ poni
		ties	$\rightarrow$ ti
SS	$\rightarrow$ SS	caress	$\rightarrow$ caress
S	$\rightarrow$ ϵ	cats	$\rightarrow$ cat

Step 1b

(m>0)	EED	$\rightarrow$ EE	feed	$\rightarrow$ feed
			agreed	$\rightarrow$ agree
(*v*)	ED	$\rightarrow$ ϵ	plastered	$\rightarrow$ plaster
			bled	$\rightarrow$ bled
(*v*)	ING	$\rightarrow$ ϵ	motoring	$\rightarrow$ motor
			sing	$\rightarrow$ sing

Porter Stemmer, Step 1c

If rule 2 or 3 in Step 1b applied, then clean up:

AT	→ ATE	conflat(ed/ing)	→ conflate
BL	→ BLE	troubl(ed/ing)	→ trouble
IZ	→ IZE	siz(ed/ing)	→ size
(*d and not (*L or *S or *Z))	→ single letter	hopp(ed/ing)	→ hop
		hiss(ed/ing)	→ hiss
(m=1 and *o)	→ E	fil(ed/ing)	→file
		fail(ed/ing)	→ fail

Step 1c

(*v*)	Y	→ I	happy	→ happi
			sky	→ sky

Porter Stemmer, Step 2

Step 2: derivational morphology

(m>0)	ATIONAL	→ ATE	relational	→ relate
(m>0)	TIONAL	→ TION	conditional	→ condition
			rational	→ rational
(m>0)	ENCI	→ ENCE	valenci	→ valence
(m>0)	ANCI	→ ANCE	hesitanci	→ hesitance
(m>0)	IZER	→ IZE	digitizer	→ digitize
(m>0)	ABLI	→ ABLE	conformabli	→ conformable
(m>0)	ALLI	→ AL	radicalli	→ radical
(m>0)	ENTLI	→ ENT	differentli	→ different
(m>0)	ELI	→ E	vileli	→ vile
(m>0)	OUSLI	→ OUS	analogousli	→ analogous
(m>0)	IZATION	→ ISE	vietnamization	→ vietnamize
(m>0)	ISATION	→ ISE	vietnamization	→ vietnamize

Porter Stemmer, Step 2, Continued

Step 2 ctd

(m>0)	ATION	$\rightarrow$ ATE	predication	$\rightarrow$ predicate
(m>0)	ATOR	$\rightarrow$ ATE	operator	$\rightarrow$ operate
(m>0)	ALISM	$\rightarrow$ AL	feudalism	$\rightarrow$ feudal
(m>0)	IVENESS	$\rightarrow$ IVE	decisiveness	$\rightarrow$ decisive
(m>0)	FULNESS	$\rightarrow$ FUL	hopefulness	$\rightarrow$ hopeful
(m>0)	OUSNESS	$\rightarrow$ OUS	callousness	$\rightarrow$ callous
(m>0)	ALITI	$\rightarrow$ AL	formaliti	$\rightarrow$ formal
(m>0)	IVITI	$\rightarrow$ IVE	sensitiviti	$\rightarrow$ sensitive
(m>0)	BILITI	$\rightarrow$ BLE	sensibiliti	$\rightarrow$ sensible

Porter Stemmer, Step 3

Step 3: more derivational morphology

(m>0)	ICATE →	IC	triplicate	→ triplic
(m>0)	ATIVE →	ϵ	formative	→ form
(m>0)	ALIZE →	AL	formalize	→ formal
(m>0)	ALISE →	AL	formalise	→ formal
(m>0)	ICITI →	IC	electriciti	→ electric
(m>0)	ICAL →	IC	electrical	→ electric
(m>0)	FUL →	ϵ	hopeful	→ hope
(m>0)	NESS →	ϵ	goodness	→ good

Porter Stemmer, Step 4

Step 4: even more derivational morphology

(m>1)	AL $\rightarrow$	ϵ	revival	$\rightarrow$ reviv
(m>1)	ANCE $\rightarrow$	ϵ	allowance	$\rightarrow$ allow
(m>1)	ENCE $\rightarrow$	ϵ	inference	$\rightarrow$ infer
(m>1)	ER $\rightarrow$	ϵ	airliner	$\rightarrow$ airlin
(m>1)	IC $\rightarrow$	ϵ	gyroscopic	$\rightarrow$ gyroscop
(m>1)	ABLE $\rightarrow$	ϵ	adjustable	$\rightarrow$ adjust
(m>1)	IBLE $\rightarrow$	ϵ	defensible	$\rightarrow$ defens
(m>1)	ANT $\rightarrow$	ϵ	irritant	$\rightarrow$ irrit
(m>1)	EMENT $\rightarrow$	ϵ	replacement	$\rightarrow$ replac
(m>1)	MENT $\rightarrow$	ϵ	adjustment	$\rightarrow$ adjust
(m>1)	ENT $\rightarrow$	ϵ	dependent	$\rightarrow$ depend
(m>1 and (*S or *T))	ION $\rightarrow$	ϵ	adoption	$\rightarrow$ adopt
(m>1)	OU $\rightarrow$	ϵ	homologou	$\rightarrow$ homolog
(m>1)	ISM $\rightarrow$	ϵ	communism	$\rightarrow$ commun
(m>1)	ATE $\rightarrow$	ϵ	activate	$\rightarrow$ activ
(m>1)	ITI $\rightarrow$	ϵ	angulariti	$\rightarrow$ angular
(m>1)	OUS $\rightarrow$	ϵ	homologous	$\rightarrow$ homolog
(m>1)	IVE $\rightarrow$	ϵ	effective	$\rightarrow$ effect
(m>1)	ISE $\rightarrow$	ϵ	bowdlerize	$\rightarrow$ bowdler
(m>1)	IZE $\rightarrow$	ϵ	bowdlerize	$\rightarrow$ bowdler

Porter Stemmer, Step 5

Step 5: cleaning up

Step 5a

(m>1)	E $\rightarrow \epsilon$	probate	$\rightarrow$	probat
		rate	$\rightarrow$	rate
(m=1 and not *o)	E $\rightarrow \epsilon$	cease	$\rightarrow$	ceas

Step 5b

(m $>$ 1 and *d and *L)	$\rightarrow$ single letter	controll	$\rightarrow$	control
		roll	$\rightarrow$	roll

Example Porter Stemmer

- Example original sentence: Document will describe marketing stategies carried out by U.S. companies for their agricultural chemicals, report predictions for market share of such chemicals, or report market statistics for agrochemicals, pesticide, herbicide, fungicide, insecticide, fertilizer, perdicted sales, market share, stimulate demand, price cut, volume of sales
- Porter-stemmed (minus stop words): market strateg carr compan agricultur chemic report predict market share chemic report market statist agrochem pesticid herbicid fungicid insecticid fertil predict sale stimul demand price cut volum sale

(Example from James Allan, Umass)

Porter Stemmer

- Output appears non-sensical – why does it work?
  - representation of root word only needs to be unique for the relevant class of words
  - and transformation needs to be repeatable
- Porter stemmer is sometimes too agressive
  - e.g. *policy/police, execute/executive, organize/organic*
- Porter stemmer sometimes misses good conflations
  - e.g. *European/Europe, matrices/matrix, machine/machinery*
- Literature gives contrasting evidence about whether stemming helps
  - but improving stemmers still an active research area

Models of Retrieval

- A model is an abstraction of a process
- A retrieval model can be a description of the human process or the computational process of retrieval
  - the process by which information needs are articulated and refined
  - the process of choosing documents for retrieval
- Here we focus on the description of the computational process

Models of the Retrieval Process

- Boolean Model
  - simple, but common in commercial systems
- Vector Space Model
  - popular in research; becoming common in commercial systems
- Probabilistic Model
- Statistical language models
- Bayesian inference networks

shared by all models: both document content and query can be represented by a bag of terms

Boolean Model

- Boolean model is simple, but popular in commercial specialist systems (e.g. bibliographic databases)
- Queries consist of terms connected by:
  AND ($\wedge$), OR ($\vee$), NOT ($\neg$)
- Key assumptions (and weaknesses)
  - Terms are either present or absent in a document (frequency not taken into account)
  - Terms are all equally informative when determining relevance
  - A document is either relevant or not relevant (no partial matches)

Example Boolean Retrieval

- User need: I'm interested in learning about vitamins other than vitamin e that are antioxidants
- User query:
  vitamin AND antioxidant AND NOT vitamin e
- Suppose there's a document which discusses the antioxidant properties of vitamin e and vitamin c
  - does the user want it?
  - does the user get it?

Advantages and Disadvantages of the Boolean Model

Advantages:

- Simple framework; semantics of a Boolean query is well-defined
  - can be implemented efficiently
  - works well with well-informed user

Disadvantages:

- Complex queries often difficult to formulate
- difficult to control volume of output
- no ranking facility
  - may require trained intermediary
  - may require many reformulations of query

Reading for Today (L2)

- Course Textbook Chapters 1; 2.1; 2.2

Post-Lecture Exercise: Porter Stemmer

1. Show which stems *rationalisations, rational, rationalizing* result in, and which rules they use.
2. Explain why *sander* and *sand* do not get conflated.
3. What would you have to change if you wanted to conflate them?
4. Find five different examples of incorrect stemmings.
5. Can you find a word that gets reduced in every single step (of the 5)?
6. Exemplify the effect that stemming (eg. with Porter) has on the Vector Space Model, using your example from before.

Post-Lecture Exercise 2

- Build your own Boolean index and search engine – as an added thrill, using only unix tools on the command line (if you want)...
- Instructions and some sample data on IR course website (this used to be an exercise for MPhil students who don't necessarily have a CS background)

3. Lecture 3: The Vector Space Model
   - Basis Vectors
   - Term Similarity metrics
     - Single Link
     - Complete Link
     - Hierarchical Clustering Examples

Vector Space Model

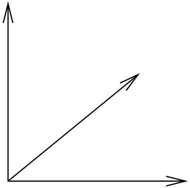
- Documents, Queries, Terms are all vectors in some high-dimensional vector space
- Key questions:
  - What are the basis vectors (dimensions)?
  - What is magnitude along a dimension?
  - How can objects in the space be compared?

Basis Vectors

- A Vector Space is defined by a linearly independent set of basis vectors
  - A set of vectors $\{\overline{v}_1, \overline{v}_2, \ldots, \overline{v}_n\}$ is linearly independent if the only solution to the equation $\lambda_1\overline{v}_1 + \lambda_2\overline{v}_2 + \ldots + \lambda_n\overline{v}_n = 0$ is $\lambda_i = 0$ for all i
  - Each vector in the set cannot be expressed as a linear combination of the remaining vectors
- Any vector in the space can be expressed as a linear combination of the basis vectors
  - basis vectors determine what objects can be described in the space

Orthogonal Basis Vectors

- If $\overline{v} \cdot \overline{w} = 0$ then $\overline{v}$ and $\overline{w}$ are orthogonal
  - $\overline{v} \cdot \overline{w} = \sum_i v_i . w_i$
  - $\overline{v} \cdot \overline{w} = |\overline{v}||\overline{w}| \cos \theta$
- If a set of vectors is pairwise orthogonal then it is linearly independent

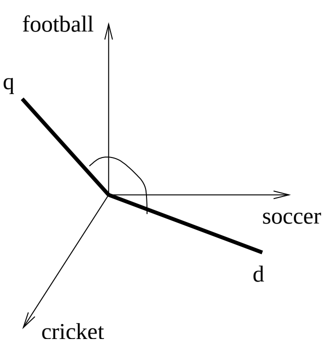


Basis vectors for 3 dimensions

Terms as Basis Vectors

- Typically terms from the document set are chosen as orthogonal basis vectors
- But terms are clearly not orthogonal (?)
- If we believe terms are "not orthogonal", we must have some pre-defined notion of a space in which terms exist
- What are the basis vectors of this space?
  - concepts?
    - documents, queries, terms are linear combinations of concepts?
  - Latent Semantic Indexing is an example of a technique which considers this question (cf. lecture 5; advanced retrieval models)

The Problem with Orthogonal Terms

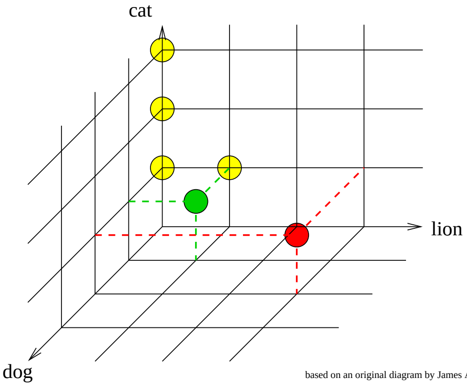


- Document d mentions soccer and cricket; query q mentions football and cricket
- Relaxing the orthogonality assumption brings d closer to q in the space

The Problem with Orthogonal Terms

- Synonymy: two documents with similar content can contain different words and be far apart in the space
  - problem of synonymy may adversely affect recall
- Polysemy: two documents can share many words, and hence be close in the space, but have very different content
  - problem of polysemy may adversely affect precision
- However, despite many attempts to improve upon the orthogonality assumption, systems which make this assumption are hard to beat

Document Representation using Term Frequency



based on an original diagram by James Allan, Umass

- yellow: {cat}, {cat cat}, {cat cat cat}, {cat lion}
- green: {cat lion dog}
- red: {cat dog dog lion lion lion}

Weighting Schemes

- Weighting scheme determines position of documents and queries in the space
  - ideally we want similar objects clustered together, and dissimilar objects far apart
- Individual vector components for an object determine:
  - the degree to which the object embodies that dimension
  - possibly the usefulness of that dimension in distinguishing the object

$$\text{e.g. TF} \times \text{IDF}$$

- Small IDF for a term effectively shrinks the space in that dimension, making it less important
- This weighting scheme is based on Zipf's law

Zipf's Law

Most frequent words in a large language sample, with frequencies:

Rank	English		German		Spanish		Italian		Dutch	
1	the	61,847	der	7,377,879	que	32,894	non	25,757	de	4,770
2	of	29,391	die	7,036,092	de	32,116	di	22,868	en	2,709
3	and	26,817	und	4,813,169	no	29,897	che	22,738	het / 't	2,469
4	a	21,626	in	3,768,565	a	22,313	è	18,624	van	2,259
5	in	18,214	den	2,717,150	la	21,127	e	17,600	ik	1,999
6	to	16,284	von	2,250,642	el	18,112	la	16,404	te	1,935
7	it	10,875	zu	1,992,268	es	16,620	il	14,765	dat	1,875
8	is	9,982	das	1,983,589	y	15,743	un	14,460	die	1,807
9	to	9,343	mit	1,878,243	en	15,303	a	13,915	in	1,639
10	was	9,236	sich	1,680,106	lo	14,010	per	10,501	een	1,637

Zipf's Law: The frequency rank of a word is reciprocally proportional to its frequency:

$$freq(word_i) = \frac{1}{i^\theta} freq(word_1)$$

Zipf's law: Rank $\times$ Frequency $\sim$ Constant

English:			
Rank R	Word	Frequency f	$R \times f$
10	he	877	8770
20	but	410	8200
30	be	294	8820
800	friends	10	8000
1000	family	8	8000

German:			
Rank R	Word	Frequency f	$R \times f$
10	sich	1,680,106	16,801,060
100	immer	197,502	19,750,200
500	Mio	36,116	18,059,500
1,000	Medien	19,041	19,041,000
5,000	Miete	3,755	19,041,000
10,000	vorläufige	1.664	16,640,000

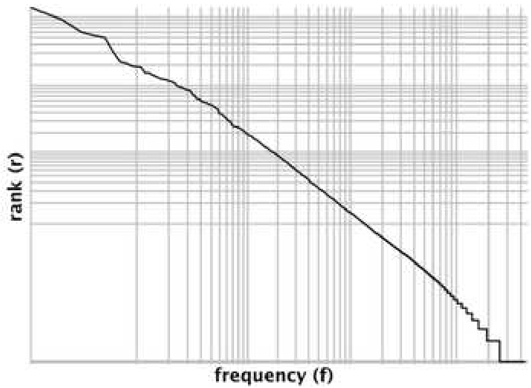
Text Coverage (tokens) with N most frequent items (types)

	German	English
1	3%	5%
10	40%	42%
100	60%	65%
1000	79%	90%
10000	92%	99%
100000	98%	

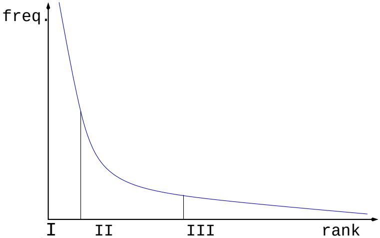
Other collections (allegedly) obeying Zipf's law

- Sizes of settlements
- Frequency of access to web pages
- Income distributions amongst top earning 3% individuals
- Korean family names
- Size of earth quakes
- Word senses per word
- Notes in musical performances
- ...

Plotting a Zipfian distribution on a log-scale



Zipf's law and term importance



- Zone I: High frequency words tend to be function words. Top 135 vocabulary items account for 50% of words in the Brown corpus. These are not important for IR.
- Zone II: Mid-frequency words are the best indicators of what the document is about
- Zone III: Low frequency words tend to be typos or overly specific words (not important, for a different reason) ("Uni7ed", "super-noninteresting", "87-year-old", "0.07685")

TF $\times$ IDF

- Term frequency: (monotonic function of) the number of times a term appears in a document
  - can be thought of as a recall enhancing device
- Inverse document frequency: (monotonic function of) the number of documents in which a term appears
  - can be thought of as a precision enhancing device
- Why not use inverse collection frequency: the total number of occurrences of a term across all documents?
  - a term may have a high collection frequency but still be concentrated in a small set of documents

A Possible TF $\times$ IDF Scheme

- $\text{TF}_{i,j} = 1 + \log(tf_{i,j})$ $(tf_{i,j} > 0)$
  - $tf_{i,j}$ is frequency of ith term in jth document
- $\text{IDF}_i = \log \frac{N}{df_i}$
  - N is number of documents in collection
  - df_i is the number of documents in which ith term appears
- Many variations of TF $\times$ IDF exist
  - Salton and Buckley (1988) give generalisations about effective weighting schemes

Query Representation

- Queries are also vectors in the space, based on the terms in the query
- Query term weights need not be the same as document term weights
- A possible query-term weighting scheme (Salton and Buckley, 1998):
  - $\text{TF} \times \text{IDF} = (0.5 + \frac{0.5 tf}{\max tf}) \log \frac{N}{n}$
    - *tf* is the number of times term appears in query
    - max tf is the highest number of occurrences for any term in the query
    - N is the total number of documents
    - n is number of documents in which query term appears

Similarity Measures

- How similar are the query and document vectors?
- Inner product
  - $D \cdot Q = \sum_i d_i . q_i$
  - documents containing many instances of informative query terms score highly
- Cosine
  - cosine(D,Q) $= \frac{D \cdot Q}{|D||Q|}$
  - length normalised inner product
    - measures angle between vectors
  - prevents longer documents scoring highly simply because of length
- There are alternative similarity measures (cf. lecture 4)

Term Similarity Metrics

- Terms can be compared using a string similarity metric, e.g. edit distance
  - More general way to obtain morphological variants
  - Also deals with other variations, e.g. typos/mis-spellings
  - Variants of *Britney Spears* entered as Google queries over a 3-month period:

    488941 britney spears 40134 brittany spears 36315 brittney spears 24342 britany spears 7331 britny spears 6633 briteny spears 2696 britteny spears 1807 briney spears 1635 brittny spears 1479 brintey spears 1479 britanny spears 1338 britiny spears 1211 britnet spears 1096 britiney spears 991 britaney spears 991 britnay spears 811 brithney spears 811 brtiney spears 664 birtney spears

Phrases, Multi-Word Terms

- Why use multi-word phrases as index terms?
  - search for *New York* may be improved if *New York* is in the index . . .
  - since we don't want documents about York and New Jersey, for example
- How do we determine the multi-word terms?
  - could do it manually, but expensive
  - automatically:
    - observe word combinations in a large corpus of text
    - extract multi-word terms on the basis of some statistic, e.g. frequency
    - accounting for syntax may help, e.g. looking for consecutive nouns in a complex noun phrase
- Why not just use quotes?

Multi-Word Terms from TREC Data

65824 United States	5778 long time
61327 Article Type	5776 Armed Forces
33864 Los Angeles	5636 Santa Ana
17788 North Korea	5527 Bosnia-Herzegovnia
17308 New York	5458 words indistinct
15513 San Diego	5452 international community
15009 Orange County	5443 vice president
12869 prime minister	5247 Security Council
12799 first time	5098 North Korean
12067 Soviet Union	5023 Long Beach
...	...

(Data from James Allan, Umass)

Reading for Today (L3)

- Course textbook, chapter 6

Supplementary:

- Term-Weighting Approaches in Automatic Text Retrieval, Salton and Buckley
- Sparck Jones and Willett, eds., Chapter 6 + Introduction to Chapter 5
- Baeza-Yates & Ribeiro-Neto, Modern Information Retrieval, Chapters 2+7
- Managing Gigabytes, 3.1, 3.2, 4.1, 4.3, 4.6

Post-Lecture Exercise

- Modify your implementation from last time so that your search model is now a VSP – several parameters can be varied
- As last time, instructions and data are on the IR course website

Uses for Clustering

- IR: presentation of results (clustering of documents)
- Summarisation:
  - clustering of similar documents for multi-document summarisation
  - clustering of similar sentences for re-generation of sentences
- Topic Segmentation: clustering of similar paragraphs (adjacent or non-adjacent) for detection of topic structure/importance
- Lexical semantics: clustering of words by cooccurrence patterns

Clustering: definition

- Partition a set of objects into groups with similar properties
- Hard clustering v. soft clustering
  - Hard clustering: every object is member in only one cluster
  - Soft clustering: objects can be members in more than one cluster
- Hierarchical v. non-hierarchical clustering
  - Hierarchical clustering: pairs of most-similar clusters are iteratively linked until all objects are in a clustering relationship
  - Non-hierarchical clustering results in flat clusters of "similar" documents (dissimilar from other clusters)

Difference clustering– text classification

- In clustering, we care about the distance from centroid ("degree of cluster membership")
- Central concepts in clustering: definition of similarity, representation of objects

Building a term–document matrix

- Cf. lecture 3: vector space representation of a document
- Decision 1: What is a term?
  - all words in document; all lemmas in document; all words modulo stoplist
  - all mid-frequency words in document
  - citations or hrefs in document to other documents
  - other features of the document (how many 'negative adjectives' does it contain?)

Building a term–document matrix, ctd

- Decision 2: Term Weighting
  - Binary: presence/absence
  - Term frequency
  - tf/idf weights

	D_1	D_2	D_3	...	D_N
$term_1$	14	6	1	...	0
$term_2$	0	1	3	...	1
...	...	...	...	...	...
$term_n$	4	7	0	...	5

Term – document matrix to document–document matrix

	D_1	D_2	D_3	...	D_N
$term_1$	14	6	1	...	0
$term_2$	0	1	3	...	1
...	...	...	...	...	...
$term_n$	4	7	0	...	5

$\rightarrow$

$$\begin{array}{c|cccc} D_1 & P_{1,1} & P_{2,1} & \dots & P_{N,1} \\ D_2 & P_{1,2} & P_{2,2} & \dots & P_{N,2} \\ D_3 & P_{1,3} & P_{2,3} & \dots & P_{N,3} \\ & \dots & \dots & \dots & \dots \\ D_N & P_{1,N} & P_{2,N} & \dots & P_{N,N} \\ & D_1 & D_2 & \dots & D_N \end{array}$$

- Decision 3: Proximity measure $P_{ij} = \text{prox}\ (D_i,\ D_j)$
- This creates a document-document matrix, which reports similarities/distances between objects (documents)
- s proximity measures are symmetric, the matrix is a triangle

Term – document matrix to document–document matrix

	D_1	D_2	D_3	...	D_N
$term_1$	14	6	1	...	0
$term_2$	0	1	3	...	1
...	...	...	...	...	...
$term_n$	4	7	0	...	5

$\rightarrow$

$$
\begin{array}{c|cccc}
D_1 & P_{1,1} & P_{2,1} & \dots & P_{N,1} \\
D_2 & P_{1,2} & P_{2,2} & \dots & P_{N,2} \\
D_3 & P_{1,3} & P_{2,3} & \dots & P_{N,3} \\
 & \dots & \dots & \dots & \dots \\
D_N & P_{1,N} & P_{2,N} & \dots & P_{N,N} \\
 & D_1 & D_2 & \dots & D_N
\end{array}
$$

- Decision 3: Proximity measure $P_{ij} = \text{prox}\ (D_i,\ D_j)$
- This creates a document-document matrix, which reports similarities/distances between objects (documents)
- As proximity measures are symmetric, the matrix is a triangle

Term – document matrix to document–document matrix

	D_1	D_2	D_3	...	D_N
$term_1$	14	6	1	...	0
$term_2$	0	1	3	...	1
...	...	...	...	...	...
$term_n$	4	7	0	...	5

$\rightarrow$

$$
\begin{array}{c|cccc}
D_1 & P_{1,1} & P_{2,1} & \dots & P_{N,1} \\
D_2 & P_{1,2} & P_{2,2} & \dots & P_{N,2} \\
D_3 & P_{1,3} & P_{2,3} & \dots & P_{N,3} \\
 & \dots & \dots & \dots & \dots \\
D_N & P_{1,N} & P_{2,N} & \dots & P_{N,N} \\
\multicolumn{1}{c}{} & D_1 & D_2 & \dots & D_N
\end{array}
$$

- Decision 3: Proximity measure $P_{ij} = \text{prox}\ (D_i,\ D_j)$
- This creates a document-document matrix, which reports similarities/distances between objects (documents)
- s proximity measures are symmetric, the matrix is a triangle
- The diagonal is trivial (identity)

Similarity metrics (no weighting)

X: set of terms occurring in document D_X, Y: set of terms occurring in document D_Y. I.e, this concerns only the presence of terms.

- Raw Overlap: $raw\_overlap(X, Y) = |X \cap Y|$
- Dice's coefficient: (normalisation by avg. size of vectors)

$$dice(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|}$$

- Jaccard's coefficient: (normalisation by size of combined vector)

$$jacc(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

Similarity metrics (no weighting)

- Overlap coefficient:

$$overlap\_coeff(X, Y) = \frac{|X \cap Y|}{min(|X|, |Y|)}$$

- Cosine: (normalisation by vector lengths)

$$cosine(X, Y) = \frac{|X \cap Y|}{\sqrt{|X|} \cdot \sqrt{|Y|}}$$

Weighted similarity metrics

Cosine (normalised inner product), used in IR as a measure of document–query similarity in vector space: document i is represented as a vectors of terms or lemmas ($\vec{w_i}$); $w_{i,j}$ is the weight associated with j th term of vector $\vec{w_i}$

$$cos(\vec{w_i}, \vec{w_k}) = \frac{\vec{w_i}\vec{w_k}}{|\vec{w_i}||\vec{w_k}|} = \frac{\sum_{j=1}^{d} w_{i,j} \cdot w_{k,j}}{\sqrt{\sum_{j=1}^{d} w_{i,j}^2} \cdot \sqrt{\sum_{j=1}^{d} w_{k,j}^2}}$$

Weighted similarity metrics

Dice's coefficient (normalisation by average size of the two vectors)

$$dice(\vec{w_i}, \vec{w_k}) = \frac{2 \cdot \sum_{j=1}^{d} w_{i,j} \cdot w_{k,j}}{\sqrt{\sum_{j=1}^{d} w_{i,j}^2} + \sqrt{\sum_{j=1}^{d} w_{k,j}^2}}$$

Jaccard's coefficient (normalisation by size of combined vector, penalises small number of shared feature values)

$$jacc(\vec{w_i}, \vec{w_k}) = \frac{\sum_{j=1}^{d} w_{i,j} \cdot w_{k,j}}{\sum_{j=1}^{d} w_{i,j}^2 + \sum_{j=1}^{d} w_{k,j}^2 - \sum_{j=1}^{d} w_{i,j} w_{k,j}}$$

Hierarchical clustering: agglomerative (BottomUp, greedy)

Given: a set $X = x_1, \ldots x_n$ of objects;
Given: a function $sim : \mathcal{P}(X) \times \mathcal{P}(X) \rightarrow \mathcal{R}$

for i:= 1 **to** n **do**
    $c_i := x_i$
C $:= c_1, \ldots c_n$
j := n+1
while C > 1 **do**
    $(c_{n_1}, c_{n_2}) := max_{(c_u, c_v) \in C \times C} sim(c_u, c_v)$
    $c_j := c_{n_1} \cup c_{n_2}$
    C := C $\{ c_{n_1}, c_{n_2} \} \cup c_j$
    j:=j+1
end

Similarity function $sim : \mathcal{P}(X) \times \mathcal{P}(X) \rightarrow \mathcal{R}$ measures similarity between clusters, not objects

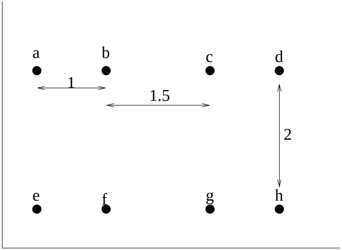
Hierarchical clustering: similarity functions

Similarity between two clusters c_k and c_j (with similarity measure s) can be interpreted in different ways:

- Single Link Function: Similarity of two most similar members
  $$sim(c_u, c_v) = max_{x \in c_u, y \in c_k} s(x, y)$$
- Complete Link Function: Similarity of two least similar members
  $$sim(c_u, c_v) = min_{x \in c_u, y \in c_k} s(x, y)$$
- Group Average Function: Avg. similarity of each pair of group members
  $$sim(c_u, c_v) = avg_{x \in c_u, y \in c_k} s(x, y)$$

Example: hierarchical clustering; similarity functions

Cluster 8 objects a-h; Euclidean distances (2D) shown in diagram



b	1						
c	2.5	1.5					
d	3.5	2.5	1				
e	2	$\sqrt{5}$	$\sqrt{10.25}$	$\sqrt{16.25}$			
f	$\sqrt{5}$	2	$\sqrt{6.25}$	$\sqrt{10.25}$	1		
g	$\sqrt{10.25}$	$\sqrt{6.25}$	2	$\sqrt{5}$	2.5	1.5	
h	$\sqrt{16.25}$	$\sqrt{10.25}$	$\sqrt{5}$	2	3.5	2.5	1
	a	b	c	d	e	f	g

Single Link is $O(n^2)$

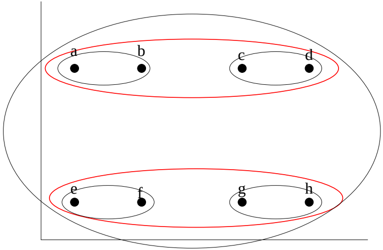
b	1						
c	2.5	1.5					
d	3.5	2.5	1				
e	2	$\sqrt{5}$	$\sqrt{10.25}$	$\sqrt{16.25}$			
f	$\sqrt{5}$	2	$\sqrt{6.25}$	$\sqrt{10.25}$	1		
g	$\sqrt{10.25}$	$\sqrt{6.25}$	2	$\sqrt{5}$	2.5	1.5	
h	$\sqrt{16.25}$	$\sqrt{10.25}$	$\sqrt{5}$	2	3.5	2.5	1
	a	b	c	d	e	f	g

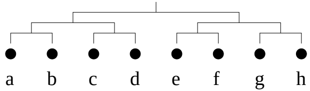
After Step 4 (a–b, c–d, e–f, g–h merged):

c–d	1.5		
e–f	2	$\sqrt{6.25}$	
g–h	$\sqrt{6.25}$	2	1.5
	a–b	c–d	e–f

"min-min" at each step

Clustering Result under Single Link





Complete Link

b	1						
c	2.5	1.5					
d	3.5	2.5	1				
e	2	$\sqrt{5}$	$\sqrt{10.25}$	$\sqrt{16.25}$			
f	$\sqrt{5}$	2	$\sqrt{6.25}$	$\sqrt{10.25}$	1		
g	$\sqrt{10.25}$	$\sqrt{6.25}$	2	$\sqrt{5}$	2.5	1.5	
h	$\sqrt{16.25}$	$\sqrt{10.25}$	$\sqrt{5}$	2	3.5	2.5	1
	a	b	c	d	e	f	g

After step 4 (a–b, c–d, e–f, g–h merged):

c–d	2.5 1.5 $\boxed{3.5}$ 2.5		
e–f	2 $\sqrt{5}$ $\boxed{\sqrt{5}}$ 2	$\sqrt{10.25}$ $\boxed{\sqrt{16.25}}$ $\sqrt{6.25}$ $\sqrt{10.25}$	
g–h	$\sqrt{10.25}$ $\sqrt{6.25}$ $\boxed{\sqrt{16.25}}$ $\sqrt{10.25}$	2 $\boxed{\sqrt{5}}$ $\sqrt{5}$ 2	2.5 1.5 $\boxed{3.5}$ 2.5
	a–b	c–d	e–f

"max-min" at each step

Complete Link

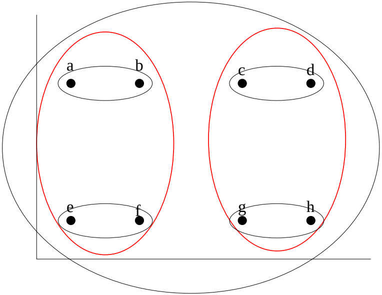
b	1						
c	2.5	1.5					
d	3.5	2.5	1				
e	2	$\sqrt{5}$	$\sqrt{10.25}$	$\sqrt{16.25}$			
f	$\sqrt{5}$	2	$\sqrt{6.25}$	$\sqrt{10.25}$	1		
g	$\sqrt{10.25}$	$\sqrt{6.25}$	2	$\sqrt{5}$	2.5	1.5	
h	$\sqrt{16.25}$	$\sqrt{10.25}$	$\sqrt{5}$	2	3.5	2.5	1
	a	b	c	d	e	f	g

After step 4 (a–b, c–d, e–f, g–h merged):

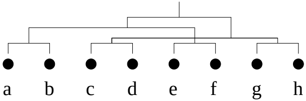
c–d	2.5 1.5 $\boxed{3.5}$ 2.5			
e–f	2 $\sqrt{5}$ $\boxed{\sqrt{5}}$ 2	$\sqrt{10.25}$ $\boxed{\sqrt{16.25}}$ $\sqrt{6.25}$ $\sqrt{10.25}$		
g–h	$\sqrt{10.25}$ $\sqrt{6.25}$ $\boxed{\sqrt{16.25}}$ $\sqrt{10.25}$	2 $\boxed{\sqrt{5}}$ $\sqrt{5}$ 2	2.5 1.5 $\boxed{3.5}$ 2.5	
	a–b	c–d	e–f	

"max-min" at each step → ab/ef and cd/gh merges next

Clustering result under complete link



Complete Link is $O(n^3)$



Example: rat gene clustering

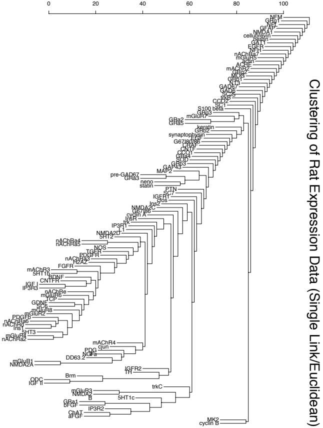
- An example from biology: cluster genes by development of CNS
- Survey 112 rat genes
- Take 9 data points: 5 embryonic (E11, E13, E15, E18, E21), 3 postnatal (P0, P7, P14) and one adult
- Measure expression of gene (how much mRNA in cell?)
- These measures are normalised logs; for our purposes, we can consider them as weights
- Cluster analysis determines which genes operate at the same time

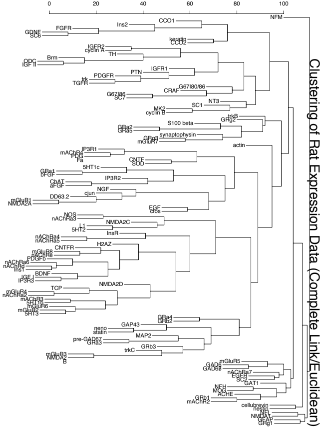
Rat gene data (excerpt)

gene	genbank locus	E11	E13	E15	E18	E21	P0	P7	P14	A
keratin	RNKER19	1.703	0.349	0.523	0.408	0.683	0.461	0.32	0.081	0
cellubrevin	s63830	5.759	4.41	1.195	2.134	2.306	2.539	3.892	3.953	2.72
nestin	RATNESTIN	2.537	3.279	5.202	2.807	1.5	1.12	0.532	0.514	0.443
MAP2	RATMAP2	0.04	0.514	1.553	1.654	1.66	1.491	1.436	1.585	1.894
GAP43	RATGAP43	0.874	1.494	1.677	1.937	2.322	2.296	1.86	1.873	2.396
L1	S55536	0.062	0.162	0.51	0.929	0.966	0.867	0.493	0.401	0.384
NFL	RATNFL	0.485	5.598	6.717	9.843	9.78	13.466	14.921	7.862	4.484
NFM	RATNFM	0.571	3.373	5.155	4.092	4.542	7.03	6.682	13.591	27.692
NFH	RATNFHPEP	0.166	0.141	0.545	1.141	1.553	1.667	1.929	4.058	3.859
synaptophysin	RNSYN	0.205	0.636	1.571	1.476	1.948	2.005	2.381	2.191	1.757
neno	RATENONS	0.27	0.704	1.419	1.469	1.861	1.556	1.639	1.586	1.512
S100 beta	RATS100B	0.052	0.011	0.491	1.303	1.487	1.357	1.438	2.275	2.169
GFAP	RNU03700	0	0	0	0.292	2.705	3.731	8.705	7.453	6.547
MOG	RATMOG	0	0	0	0	0.012	0.385	1.462	2.08	1.816
GAD65	RATGAD65	0.353	1.117	2.539	3.808	3.212	2.792	2.671	2.327	2.351
pre-GAD67	RATGAD67	0.073	0.18	1.171	1.436	1.443	1.383	1.164	1.003	0.985
GAD67	RATGAD67	0.297	0.307	1.066	2.796	3.572	3.182	2.604	2.307	2.079
G67I80/86	RATGAD67	0.767	1.38	2.35	1.88	1.332	1.002	0.668	0.567	0.304
G67I86	RATGAD67	0.071	0.204	0.641	0.764	0.406	0.202	0.052	0.022	0
GAT1	RATGABAT	0.839	1.071	5.687	3.864	4.786	4.701	4.879	4.601	4.679
ChAT	(*)	0	0.022	0.369	0.322	0.663	0.597	0.795	1.015	1.424
ACHE	S50879	0.174	0.425	1.63	2.724	3.279	3.519	4.21	3.885	3.95
ODC	RATODC	1.843	2.003	1.803	1.618	1.569	1.565	1.394	1.314	1.11
TH	RATTOHA	0.633	1.225	1.007	0.801	0.654	0.691	0.23	0.287	0
NOS	RRBNOS	0.051	0.141	0.675	0.63	0.86	0.926	0.792	0.646	0.448
GRa1	(#)	0.454	0.626	0.802	0.972	1.021	1.182	1.297	1.469	1.511

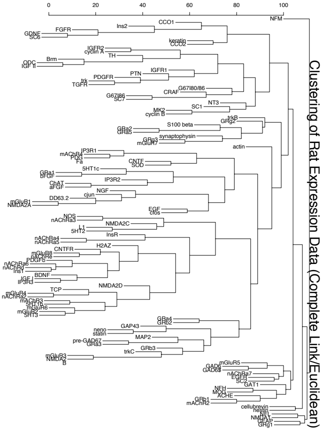
...

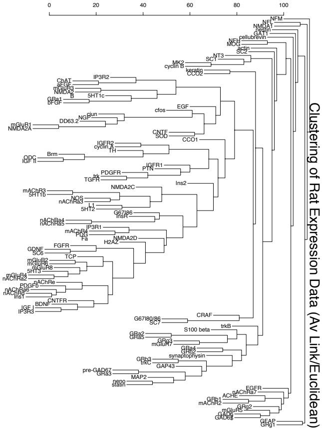
Rat gene clustering – single link vs complete link





Rat gene clustering – complete link vs group average link





Non-hierarchical (partitioning) clustering

- Partitional clustering algorithms produce a set of k non-nested partitions corresponding to k clusters of n objects.
- Advantage: not necessary to compare each object to each other object, just comparisons of objects – cluster centroids necessary
- Centroid c_j of cluster j (with size n_j): average vector of cluster (need not be an actual vector)

$$c_j = \frac{1}{n_j} \sum_{i=1}^{n_j} x_i$$

- Medoid m_j: representative vector closest to centroid

$$m_j = \{\vec{x_i} | \forall_{x_j \in j} : d(\overrightarrow{x_i}, \overrightarrow{c_j}) \leq d(\overrightarrow{x_j}, \overrightarrow{c_j})\}$$

Non-hierarchical (partitioning) clustering

- Measure of cluster quality: mean square distance from each data point to its nearest center should be minimal (within-clusters sum of squares)

$$\sum_{j=1}^{k} \sum_{i \in c_j} d(\vec{x_i}, \vec{f_j})^2$$

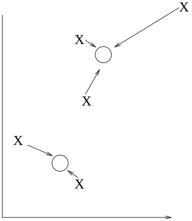
- Partitions can be obtained with hierarchical clustering with similarity threshold or at predefined dendrogram cut-off level
- But: complexity issues; use iterative partitional clustering methods with lower complexity such as K-means, Scatter/Gather
- Optimal partitioning clustering algorithms are $O(kn)$

K-means

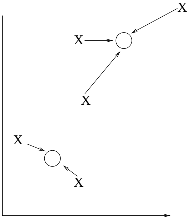
Given: a set $X = \vec{x_1}, \ldots \vec{x_n} \subseteq \mathcal{R}^m$
Given: a distance measure $d : \mathcal{R}^m \times \mathcal{R}^m \rightarrow \mathcal{R}$
Given: a function for computing the mean $\mu : \mathcal{P}(\mathcal{R}) \rightarrow \mathcal{R}^m$

Select k initial centers $\vec{f_1}, \ldots \vec{f_k}$
while stopping criterion not true:
    $\sum_{j=1}^{k} \sum_{i \in c_j} d(\vec{x_i}, \vec{f_j})^2 < \epsilon$ (stopping criterion)
    do
    for all clusters c_j **do**
        $c_j := \{\vec{x_i} | \forall \vec{f_l} : d(\vec{x_i}, \vec{f_j}) \leq d(\vec{x_i}, \vec{f_l})\}$
    end
    for all means $\vec{f_j}$ **do**
        $\vec{f_j} := \mu(c_j)$
    end
end

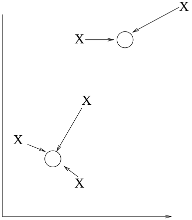
Non-hierarchical clustering: K-means



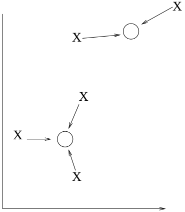
Step 1: Assignment to cluster



Step 2: Reassignment of centroid



Step 3: Reassignment to clusters



Step 4: Reassignment of centroids

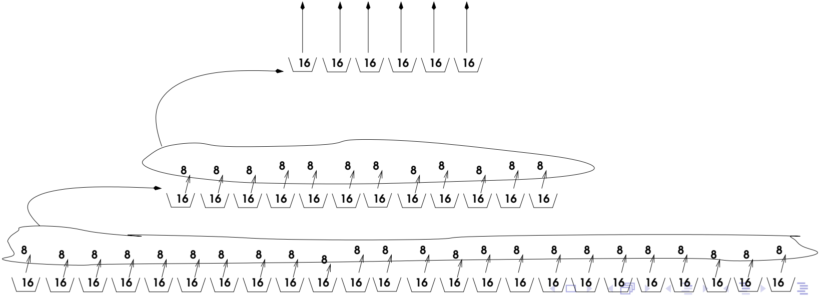
Complexity: $O(n)$

Center-finding algorithms

- Buckshot ($O(kn)$)
  1. Randomly choose $\sqrt{kn}$ object vectors $\rightarrow$ set Y (example: $n = 162$, $k = 8 \rightarrow$ choose $\sqrt{162 \cdot 8} = 36$ random objects
  2. Apply hierarchical clustering to Y, until k clusters are formed. This has $O(kn)$ complexity
  3. Return k centroids of these clusters; (assign remaining documents to these)
- Fractionation ($O(mn)$); has parameters ρ and m
  1. Split document collection into $\frac{n}{m}$ buckets B
     - $B = \{\Theta_1, \Theta_2, \ldots \Theta_{\frac{n}{m}}\}$
     - $\Theta_i = \{x_{m(i-1)+1}, x_{m(i-1)+2}, \ldots x_{mi}\}$; for $1 \geq i \geq \frac{n}{m}$
  2. Apply hierarchical cluster to each B with a threshold at $\rho \cdot m$ clusters
  3. Treat the groups formed in step 2 as individual documents and repeat 1-2 until only k buckets remain
  4. Return the centroids of the k buckets

Fractionation example

- Documents to cluster: $n = 384$; $k = 6$
- Bucket size m=16; Reduction factor $\rho = 0.5$
- First step: 24 buckets, 2nd step: 12 buckets, 3rd step: 6 buckets ($= k$)
- Best centroids found with high reduction (close to 1) and large bucket size



Summary of today

- Two kinds of matrices are in use in clustering:
  - Document-feature matrix
  - Document-document matrix
- Hierarchical clustering on document-document matrix
  - Best algorithms $O(n^2)$ complexity
  - Single-link vs. complete-link vs. group-average
- Partitional clustering
  - Provides less information but is more efficient (best: $O(kn)$)
  - K-means
  - Centroid-finding algorithms
- Hierarchical and non-hierarchical clustering fulfills different needs

Reading for Today (L4)

- Textbook: Chapters 16 and 17.
- alternative is chapter 14 in Manning and Schuetze, Foundations of Statistical Natural Language Processing, 1999, MIT press.
- Paper: M. Hearst and Pedersen, Reexamining the cluster hypothesis: Scatter/Gather on Retrieval Results; ACM/SIGIR-1996

Post-Lecture Exercise

- Download and install the open-source clustering toolkit Cludo
- Download rat gene data from IR course website
- See if you can replicate the clusterings from the lecture (they were not created using Cludo, but with another toolkit)