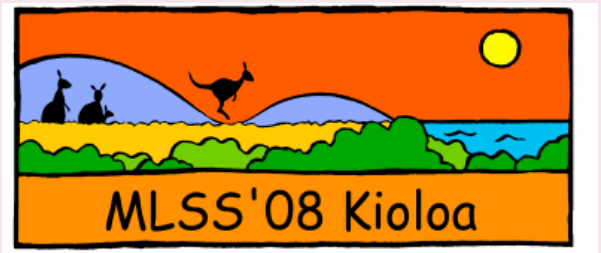
Latent Variable Models for Document Analysis

Wray Buntine
National ICT Australia (NICTA)



Part I

Motivation and Background

What a good Statistical NLP Course Needs

Apart from the usual CS background (algorithms, data structures, coding, *etc.*):

- prerequisites or coverage of information theory, and computational probability theory;
- theory of context free grammars, normal forms, parsing theory,*etc.*;
- programming tools: Python!

None of this is presented here!

Outline

Outline

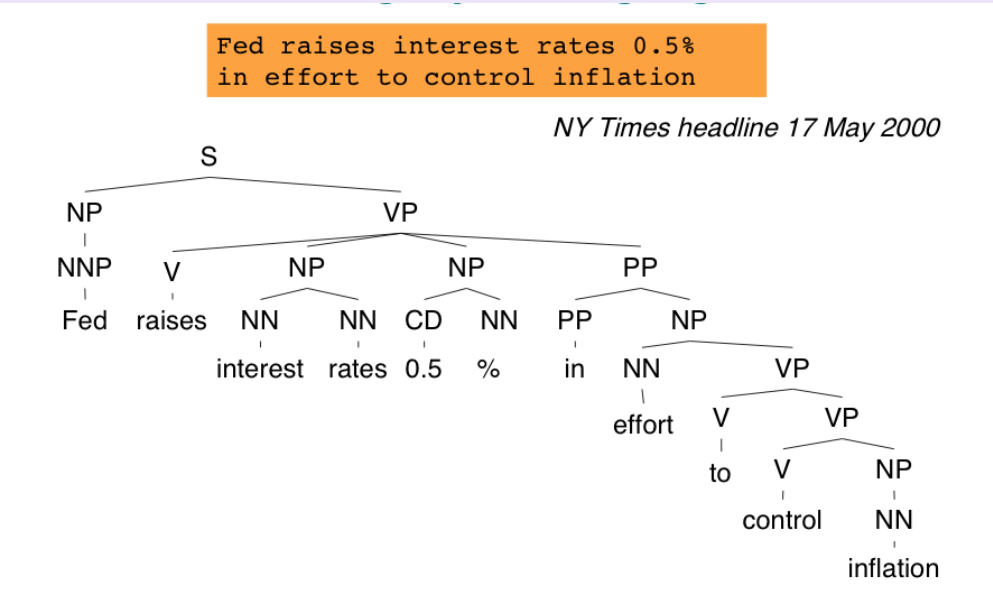
We do a review of the analysis of formal natural language (not a formal analysis of natural language).

What is Formal Natural Language

- Formal language is taught in schools (*e.g.*, grammar schools) with correct grammar, punctuation and spelling.
- Most books, more traditional print media, formal business communication, and newspapers use this.
- But errors exist even in the *The Times* and *The New York Times*.
- In contrast, informal language is found in email, people's web pages, chat groups, and "trendy" print media.

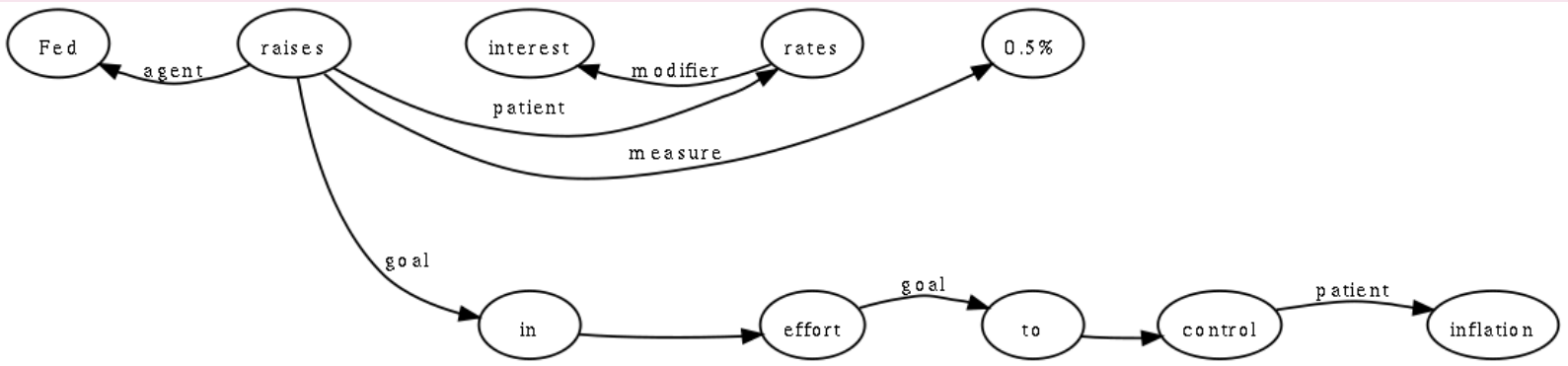
Outline

Analysing Language

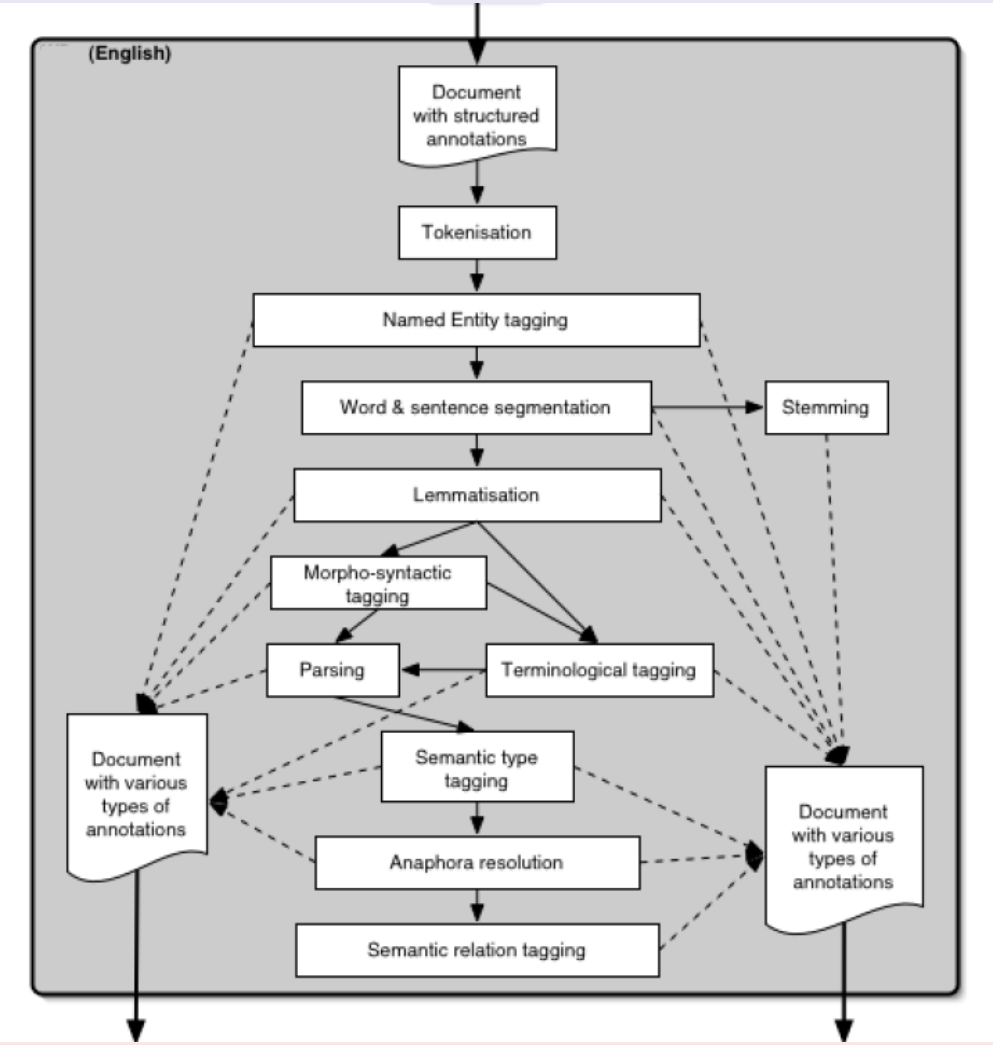


Example from *McCallum's NLP course*

- Left, a traditional parse tree showing constitutent phrases.
- Below, a dependency graph showing *semantic roles*.



Traditional NLP Processing



Full processing pipeline might look like this for English.

- Typical accuracies for various stages might be 90-98%.
- But it can drop down to 60% for the later semantic analysis.

Common Tasks in NLP

Tokenisation: breaking text up into basic tokens such as word, symbol or punctuation.

Chunking: detecting parts in a sentence that correspond to some unit such as "noun phrase" or "named entity".

Part-of-speech tagging: detecting the part-of-speech of words or tokens.

Named entity recognition: detecting proper names.

Parsing: building a tree or graph that fully assigns roles/parts-of-speech to words, and their inter-relationships.

Semantic role labelling: assigning roles such as "actor", "agent", "instrument" to phrases.

NLP in Chinese

Input
A Chinese sentence
我弟弟要买两个足球。
My brother wants to buy two balls.

Output (the word and POS sequence)
我/r (my) 弟弟/n (brother) 要/v (want)
买/v (buy) 两/m (two) 个/q (classifier)
足球/n (football) 。/w (period)

- Tokenisation (segmenting words) is very difficult.
- Easier in Japanese[1] because their foreign words use separate phonetic alphabets.
- Little morphology used.

[1] Japanese writing is based on traditional Chinese.

Translation Difficulties

English: I am in the cafe too.

Finnish: On kahvilassahan.

Finnish, an agglutinating language like Mongolian and Turkish, can express four English words in one!

The translation is: $\text{On}_{\text{I am}}$ $\text{kahvi}_{\text{coffee}}\text{la}_{\text{place}}\text{ssa}_{\text{in}}\text{han}_{\text{emphasis}}$.

This makes statistical machine translation very difficult. For instance, only the base word "kahvila" will be in any dictionary.

Translation Difficulties, cont.

Some languages represent names differently, especially those originating outside of the Latin based alphabets.

Code	Language	Translation
EN	English	Saddam Hussein
LV	Latvian	Sadams Huseins
HU	Hungarian	Szaddám Huszein
ET	Estonian	Saddäm Husayn

Language Ambiguities

An unnamed high-performance commercial parser made the following analysis of a sentence from Reuters Newswire in 1996.

Clothes made of [hemp and smoking paraphernalia]$_{\text{phrase}}$ were on sale.

The correct analysis is:

[Clothes made of hemp]$_{\text{phrase}}$ and [smoking paraphernalia]$_{\text{phrase}}$ were on sale.

This misinterpretation is a common semantic problem with current parsing technology.

Language Ambiguities, cont.

- New$_{\text{adjective}}$ York Tennis Club$_{\text{name}}$ opening today. versus
  New York Tennis Club$_{\text{name}}$ opening today.
- He worked at Yahoo!$_{\text{sentence}}$ Tuesday.$_{\text{sentence}}$ versus
  He worked at Yahoo!$_{\text{name}}$ Tuesday.$_{\text{sentence}}$
- Stolen painting found by tree$_{\text{location}}$. versus
  Stolen painting found by tree$_{\text{actor}}$.
- Iraqi head$_{\text{body part}}$ seeks arms$_{\text{body part}}$. versus
  Iraqi head$_{\text{politician}}$ seeks arms$_{\text{weapons}}$.

Language Ambiguities, cont.

- Ambiguities arise in all processing steps, due to the tokenisation done, the identification of proper names, the part of speech assigned, the parse, or the semantic role assigned.
- All languages have particular versions of the ambiguity problem. *e.g.*, standard Arabic and Hebrew don't represent vowels in their text!

We resolve ambiguity by appeal to *distributional semantics*, that the meaning of a word is given by its distribution with the words surrounding it, its context.

Handling of ambiguity generally requires that intermediate processing carry uncertainty, for instance, by using latent variables in statistical methods.

Outline

Word Classes (dictionary version of part of speech)

Part of speech	Function	Examples
Verb	action or state	(to) be, have, do, like, work, sing, can, must
Noun	thing or person	pen, dog, work, music, town, London, John
Adjective	describes a noun	a/an, 69, some, good, big, red, well, interesting
Adverb	describes a verb, adjective or adverb	quickly, silently, well, badly, very, really
Pronoun	replaces a noun	I, you, he, she, some
Preposition	links a noun to another word	to, at, after, on, but
Conjunction	joins clauses or sentences or words	and, but, when, because
Interjection	short exclamation, can be in sentence	oh!, ouch!, hi!

Word Forms

Morpheme: Is a semantically meaningful part of a word.

Inflection: A version of the word within the one word class by adding a grammatical morpheme. "walk" to "walks", "walking", and "walked".

Lemma: The base word form without inflections, but no change in word class. "walking" lemmatizers back to "walk", but "redness" (N) does not lemmatise to "red" (A).

Derivation: Adding grammatical morphemes to change the word class. "appoint" (V) to "appointee" (N), "clue" (N) to "clueless" (A). Uses "-ation", "-ness", "-ly" *etc.*

Stemming: Primitive version of lemmatization that strips off grammatical morphemes naively, usually in a context free manner.

Open versus Closed: Nouns, verbs, adjectives, adverbs are considered *open* word classes that continually admit new entries.

Parts of Speech (computational version)

Example parts of speech from the Tagging Guidelines for the Penn Treebank.

POS	Function	Examples
CC	coordinating conjunction	and, but, either
CD	cardinal number	three, 27
DT	determiner	a, the, those
IN	preposition or subordinating conjunction	out, of, into, by
JJ	adjective	good, tall
JJS	adjective, superlative	best, tallest
MD	modal	he *can* swim
NN	noun, singular or mass	the *ice* is cold
NNS	noun plural	the *iceblocks* are cold
PDT	predeterminer	*all* the boys
SYM	symbol	$, %
VBD	verb, past tense	swam, walked
...	...	...

Parts of Speech (computational version), cont.

- For computational analysis, more detail over the 8 word classes is needed in order to capture inflections and variations supporting a parse.
- With just candidate POS for each word, many different parses can exist. McCallum's initial example is shown again below.

	VBZ	VB VBZ	VBZ			
NNP	NNS	NNS	NNS	CD	NN	
Fed	raises	interest	rates	0.5	%	in effort to control inflation

Collocations

Small, usually contiguous, sequence of word that behaves semantically like a single word: "hot dog", "with respect to", "home page", "fourth quarter", "run down",

- Meaning of a collocation is different to the meaning of its parts.
  - The collocation cannot be modified easily without changing the meaning: "kicked the bucket" versus "kicked the tub", "the bucket was kicked".
  - We identify collocations by appeal to distributional semantics.
- Related: multi-word expression/unit, compound, idiom.
- In some languages, collocations replaced by compounds (words are joined with no space or hyphen).
- Important for parsing, dictionaries, terminology extraction, ...

Outline

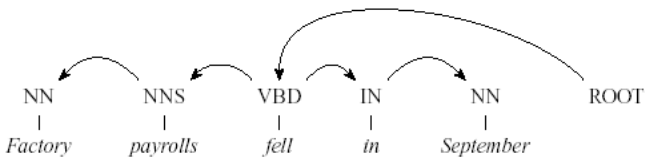
Constituents

A word or a group of words that functions as a single unit within a hierarchical structure.

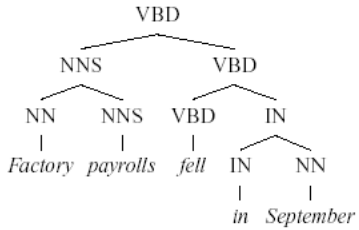
e.g. noun phrase, prepositional phrase, collocation, *etc.*

- Often can be replaced by a single pronoun and the enclosing sentence is still gramatically valid.
- Serve as a valid answer to some question.
  e.g., How did you get to work? By train.
- Admits standard syntactic manipulations.
  e.g., can be joined with another using "and", can be moved elsewhere in the sentence as a unit.
- Building a parse tree involves building the complete set of constituents for a sentence.

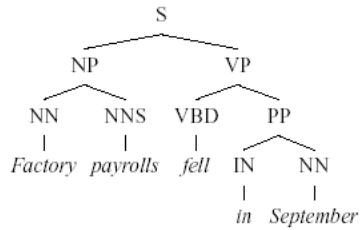
Parsing



(a) Classical Dependency Structure



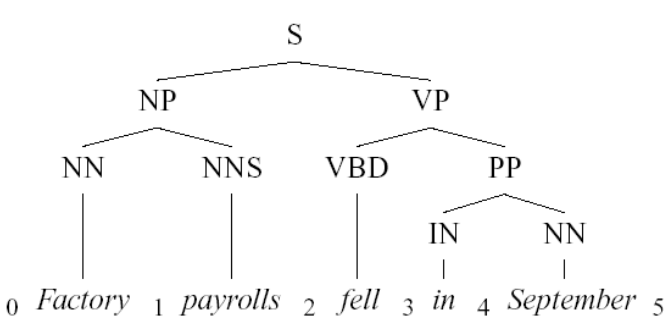
(b) Dependency Structure as CF Tree



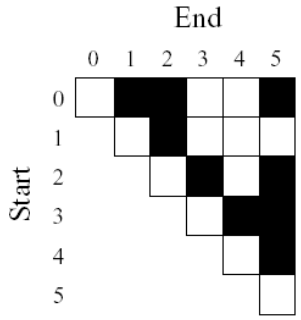
(c) CFG Structure

- Sometimes we want a dependency tree showing syntactic or semantic relationships, as in (a).
  - Usually, we want the relationships labelled.
    e.g. arc from "fell" to "in" labelled with *time*, arc from "fell" to "payrolls" labelled with *patient*.
- Some formal linguistic theory develops a parse tree, in this case a Context Free Grammar (CFG) is used in (c).
- Figure shows a derivation of the parse tree from the dependency tree.

Shallow Parsing



(a)



(b)

Span	Label	Constituent	Context
⟨0,5⟩	S	NN NNS VBD IN NN	◇ – ◇
⟨0,2⟩	NP	NN NNS	◇ – VBD
⟨2,5⟩	VP	VBD IN NN	NNS – ◇
⟨3,5⟩	PP	IN NN	VBD – ◇
⟨0,1⟩	NN	NN	◇ – NNS
⟨1,2⟩	NNS	NNS	NN – VBD
⟨2,3⟩	VBD	VBD	NNS – IN
⟨3,4⟩	IN	IN	VBD – NN
⟨4,5⟩	NN	NNS	IN – ◇

(c)

1: (a) Example parse tree with (b) its associated bracketing and (c) the yields and contexts for each constituent span

- A full parse yields many subtrees or constituents, labelled verb phrase (VP), prepositional phrase (PP), *etc.*
- We can also note the labels of a particular type (*e.g.*, all NPs), and build a classifier that recognises just that type.
- Recognising the start and end of a particular type of constituent is called **shallow parsing** or **chunking**.
- Parsing can also be represented as a structured classification problem, recognising the best coherent set of constituents.

Outline

We look beyond the text content to consider applications of document processing.

1. Formal Natural Language
2. Document Processing
   - Language in the Electronic Age
   - Information Warfare
   - Why Analyse Documents
3. Document Analysis

Processing of Documents

- Documents have a structure with text, links to other documents, citations to publications, images, indexes, and so forth.
- Why do we care about documents?
- What applications can be made?

Outline

Informal Language

Text messages: My smmr hols wr CWOT. B4, we used 2go2 NY 2C my bro, his GF & thr 3 :- kids FTF. ILNY, it's a gr8 plc.

IRC Chat:
`Meta-man:` NLP is a little tricky to do over IRC
`Dan_26:` I see no diff
`galamud:` I'm not pissed! I'm flattered! I mean, er... =)
`Meta-man:` hold that thought ...to your checkbook :]
`JonathanA:` HAH! LOL

Web Page Structure

- Web pages have complicated structures and genre, more so than traditional documents (letters, books, *etc.*).
- Example genres: product page, personal home page, FAQ, news item, blog, corporate data sheet, ...
- Much of the content will be template content shared across many similar pages.
- No standard guidelines, so must determine heuristically.

Linguistic Resources

- A large number of different resources now becoming available, due to the Internet and digitisation.
- Included: gazetteers, dictionaries, tagged text (tagged with POS, name entity types, *etc.*), word sense data, case frame and semantic role data (*i.e.*, for verbs), collocations, aligned translations.
- Tagged and marked up linguistic resources are the hardest to get, but are the ones most needed for supervised statistical NLP.

Availability of linguistic resources is a key determining factor in the success of statistical NLP projects.

Unsupervised (or semi-supervised) approaches to statistical NLP are most needed.

Outline

The Internet Society

- Primary school students have internet component in coursework are given internet search tasks as assignments.
- Internet news and blogs have overtaken newspapers as primary information source, but the business models are unclear.
- E-government, business and consumer e-services booming.
- Search and internet-based multimedia now a significant form of entertainment.

  e.g. 8 year-old boy with keywords “dinosaur”, “meteor”.

The Internet Society, cont.

- Advertising on specialist websites, on particular keyword searches, or on your email based on its content, is well focussed.
- Targeted advertising through the web, for instance Google AdSense, is considered the best value for money for advertising.
- Major industry companies track "green" websites and blogs for potential environmental scandals.

Document analysis has taken on a new life due to the internet. Business, government and consumer ramifications still unfolding.

Information Warfare

Definition: "the use and management of information in pursuit of a competitive advantage over an opponent."

- Email spam, link spam, *etc.* Whole websites are now fabricated with fake content in the effort by spammers.
- "More than half of Americans say US news organizations are politically biased, inaccurate, and don't care ...," *Pew Research Center on "news"* (Aug. 2007)
  - "Poll respondents who use the Internet as their main source of news – roughly one quarter of all Americans – were even harsher with their criticism."
  - 80% of the watchers of FOX news had one or more major misconceptions over Iraq war, compared with only 23% for PBS/NPR, *WorldPublicOpinion.ORG survey* (Oct. 2003)

It's an information war out there on the internet (between consumers, companies, not-for-profits, voters, parties, news publishers, ...).

Outline

Bioinformatics: Medline

- *PubMed* is the most popular database in Biology, and the main database MedLine has over 16 million entries.
  - entries are abstracts and metadata in (*MedLine format*, *XML format*, ...
  - 2,000-4,000 new entries/day from 5000 journals in 37 languages.
- The abstract databases are searchable using free text and contolled vocabularies, such as *MeSH* terms.

Tasks in MedLine

- The MeSH terms are generally entered by users and not thorough. Thus subject-specific searching patchy.
- Named entities (genes, proteins) have many different versions so it is difficult to search for them.
- Same problems apply to many technical information resources, such as patent databases.

European Media Monitor: NewsExplorer

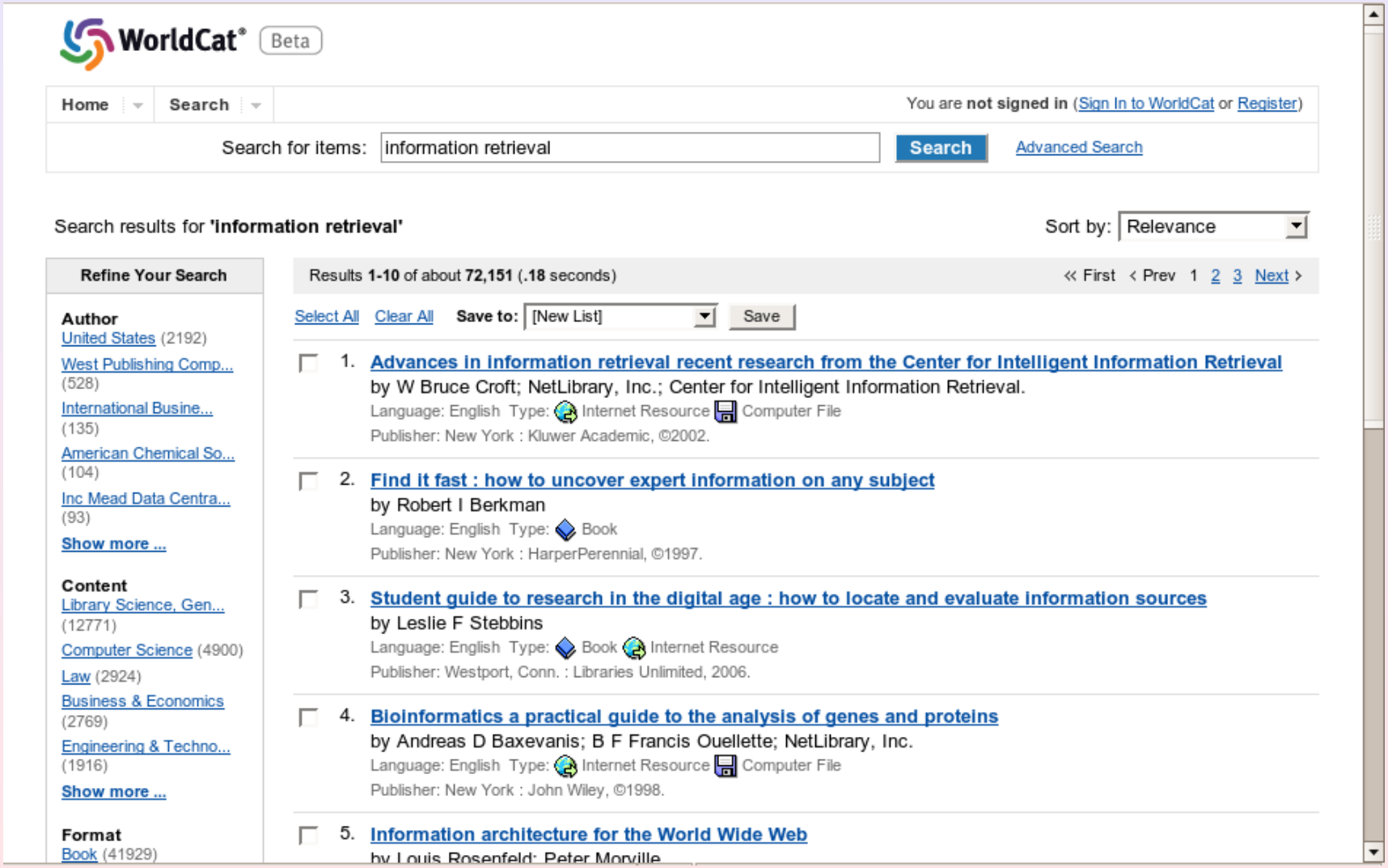
- Developed at the European Commission's Joint Research Center (JRC) in Italy. Online at `http://press.jrc.it/`.
- Completely automated:
  - automatically generate daily news summaries, and provides a *daily briefing*,
  - collect and cluster news events, and *news personalities*,
  - provide geographical, *theme* and time summaries,
  - cross-lingual capabilities.
- Uses relatively simple NLP and SML technology cleverly.
- Widely regarded within the EU Commission and by Google.

Advanced Search Engines

- Clustering output to give a dynamic snapshot of the area, such as *Clusty*.
- Providing a stronger typing of content in terms of area, keyword, genre, document type, such as *Exalead*
- Subject specific areas such as *academic search*, product search and *library catalogue search*.

Advanced Search Engines: Visualisation

World Wide Library



WorldCat® Beta

Home | Search

You are **not signed in** (Sign In to WorldCat or Register)

Search for items: information retrieval [Search] Advanced Search

Search results for **'information retrieval'**

Sort by: Relevance

Refine Your Search

Author
United States (2192)
West Publishing Comp... (528)
International Busine... (135)
American Chemical So... (104)
Inc Mead Data Centra... (93)
Show more ...

Content
Library Science, Gen... (12771)
Computer Science (4900)
Law (2924)
Business & Economics (2769)
Engineering & Techno... (1916)
Show more ...

Format
Book (41929)

Results **1-10** of about **72,151** (**.18** seconds)

« First ‹ Prev 1 2 3 Next ›

Select All Clear All **Save to:** [New List] Save

1. **Advances in information retrieval recent research from the Center for Intelligent Information Retrieval**
by W Bruce Croft; NetLibrary, Inc.; Center for Intelligent Information Retrieval.
Language: English Type: Internet Resource Computer File
Publisher: New York : Kluwer Academic, ©2002.

2. **Find it fast : how to uncover expert information on any subject**
by Robert I Berkman
Language: English Type: Book
Publisher: New York : HarperPerennial, ©1997.

3. **Student guide to research in the digital age : how to locate and evaluate information sources**
by Leslie F Stebbins
Language: English Type: Book Internet Resource
Publisher: Westport, Conn. : Libraries Unlimited, 2006.

4. **Bioinformatics a practical guide to the analysis of genes and proteins**
by Andreas D Baxevanis; B F Francis Ouellette; NetLibrary, Inc.
Language: English Type: Internet Resource Computer File
Publisher: New York : John Wiley, ©1998.

5. **Information architecture for the World Wide Web**
by Louis Rosenfeld; Peter Morville

Patent Search: PatentLens

- Started out as a *patent search engine* for Bioinformatics to support patent packaging.
- Software is open source, but largely developed in-house at *Cambia*.
- Many specific facilities to support patents (organisation/company matching, cross-nation support, gene name search ...).
- The patent landscape is changing, see *Open Invention Network*.

Social Bookmarks: Del.icio.us

- *Del.icio.us* is one of the best known social bookmarking sites.
- Uses tagging to provide higher-weighted keywords.
- Uses social bookmarks to get popularity/"authority" for pages.
- Purchased by Yahoo in 2005.

Opinion: their search returns best pages on fairly general topic areas, *e.g.* *information retrieval*, (*i.e.*., but not "home page" or "lost page" search).

Business Applications

Intelligence: information from the web about consumer trends and opinions, and about competitors.

Summaries: executive reports and overviews based on a large collection of documents input.

Intranet support: search and browse, personalisation, categorization, document management.

Administration: eGovernment and electronic document processing.

Advertising: many aspects of advertising now running online.

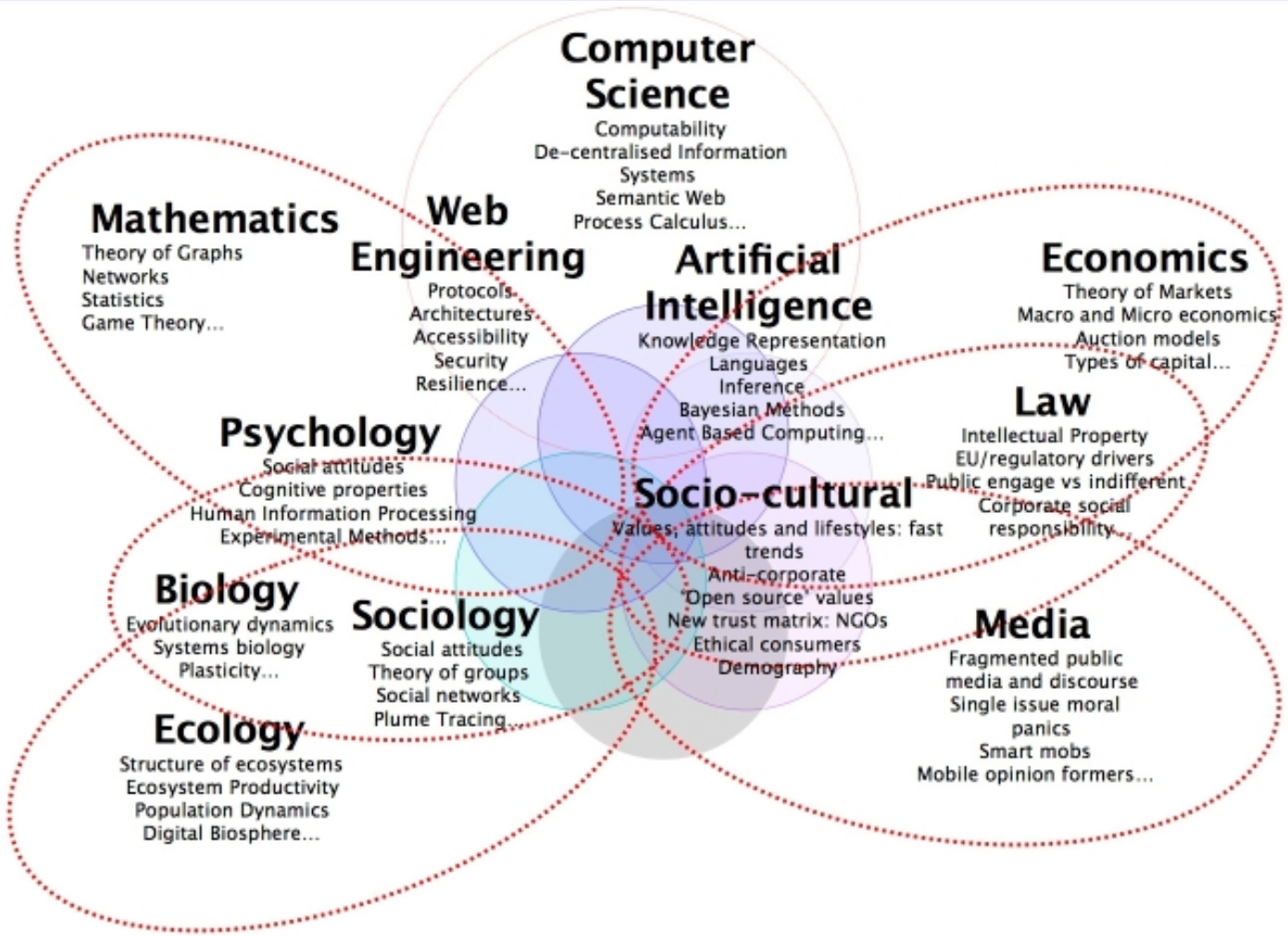
Outline

We sketch out the field of document analysis, with major emphasis on text.

1. Formal Natural Language
2. Document Processing
3. Document Analysis
   - Where is the Science of Document Analysis?
   - Representation
   - Resources
   - Other Areas

Outline

Web Science



From *Web Science*.

Outline

Linguistic Representation

Linguistic aspects:

- basic representations presented previously: morpheme, token, word class, part-of-speech, lemma, collocation, term, named entity, constituent, phrase, parse tree, case frame, semantic role, dependency graph;
- transformations and default processing steps between them;
- differences for different languages;
- sources of ambiguity.

It is important to understand the linguists viewpoints, and their whys and wherefores.

Computational Representation

Computational aspects for the text in documents:

- data formats such as XML and its support tools and representations such as Schema, XQuery, ...;
- data structures and manipulation such as trees, graphs, regular expressions, FSA, ...;
- character processing, UTF8, simplified Chinese, Latin, ...

All of these aspects make a scripting language like Python (or Perl) the best platform for beginning statistical NLP.

Meaning Representation

The layers of processing for the text in documents.

Character level: characters $\longrightarrow$ tokens sentences $\longrightarrow$ paragraphs $\longrightarrow$ documents.

Syntactic level: morphemes $\longrightarrow$ lemmas and parts of speech $\longrightarrow$ collocations, terms and named entities $\longrightarrow$ constituents, phrases $\longrightarrow$ sentences.

Semantic level: case frames and semantic roles, dependencies, topic modelling, genre.

The three levels tend to interact, and the various stages in each level interact as well.

Outline

Part of Speech Data

- Human annotators have taken, say, 20Mb of Wall Street Journal text and carefully assigned POS to tokens.
- There can be some difficulty in assigning POS:
  - "She stepped off/IN the train." `versus` "She pulled off/RP the trick."
  - "We need an armed/JJ guard." *versus* "Armed/VBD with only a knife, ..."
  - "There/EX was a party in progress there/RB."
- POS data laborious to construct, but very useful for statistical methods.

Most parsers don't require POS tagging beforehand. It is generally done as a pre-processing step for information extraction. or shallow parsing.

Computer Dictionary: CELEX

- CELEX is the Dutch Centre for Lexical Information.
- Provides CDROM with lexical information for English, German and Dutch, called *CELEX2*. Available from LDC.
- Contains orthography (spelling), phonology (sound), morphology (internal structure of words), syntax, and frequency for both lemmas and word-forms.
- Provided for 50,000 lemmata.

Headword	Pronunciation	Morphology	Cl	Type	Freq
celebrant	"sE-lI-br@nt	((celebrate),(ant))	N	sing	6
cellarages	"sE-l@-rIdZIs	((cellar),(age),(s))	N	plu	0
cellular	"sEl-jU-l@r*	((cell),(ular))	A	pos	21

Computer Thesaurus: WordNet

- Developed at Princeton University under the direction of psychology professor George A. Miller from 1985 on.
- Contains over 150,000 words or collocations, *e.g.* see *make*, *red*, *text*.
- Words in a network with link types corresponding to:

  hypernym: generalisation,
  hyponym: specialisation,
  holonym: has as a part,
  meronym: is a part of,
  antonym: contrasting or opposite,
  derivationally related: "textual" is for "text",
  word senses: different semantic use cases identified,
  case frames: case frames for verbs.

- Available free (with an "unencumbered license"), and lots of supporting software.

Gazetteers

- Term originally applies to geographic name databases that might contain auxiliary data such as type (mountain, town, river, *etc.*), location, parent state, *etc.*
- Sometimes extended in NLP to apply to other specialised databases of proper names.
- Proper names treated differently in NLP because:
  - they behave as single tokens and don't inflect,
  - generally are marked with first letter uppercase,
  - are the greatest source of new or unknown words in text, and are not usually in dictionaries.

Good gazetteers and dictionaries are critical for performance in any specialised domain.

Linguistic Data Consortium

- *LDC* is an open consortium initially funded by ARPA.
- Wide *variety of data* including speech and transcripts, news and transcripts, language resources, annotated and parsed data.
- Includes the famous Penn Treebank which has POS tagging and parse trees.

Outline

Important Issues

We've looked at applications, representation and linguistic resources, what about:

Software: many open source tools exist of varying quality, though some of the best tools are commercial and expensive.

Evaluation: a myriad of evaluation tracks exist for every aspect, and these generate some important data sets and resources.

Algorithms: space and time complexity, *etc.*

Statistical prerequisites: the field has prodigious users and creators of statistical techniques.

Recognised Problems

Information retrieval (IR): given query words, retrieve relevant parts from a document collection.

Question answering (QA): similar to IR but return an answer.

Document summarisation: taking a small set of documents on a given theme and preparing a short summary or executive brief.

Topic detection and tracking (TDT): tracking topics, and discovering new ones in information streams.

Semantic web annotation: annotating documents with appropriate semantic mark-up.

Classification: categorising documents into topic hierarchies, or creating hierarchies suited for a collection.

Genre identification: predicting the genre type.

Sentiment analysis: predicting the sentiment (negative, satisfied, happy, ...) of a blog or chat participant or commentary.

Recognised Problems, cont.

Document structure analysis: identifying the parts of a web page or document such as title, index, advertising, body, *etc.*

Linguistic resource development: tagging of text with parse structures, POS, semantic roles, name entities, *etc.*, and development of dictionaries, gazetteers, case frames, *etc.*, especially in specialised subjects.

Recommendation: from user characteristics and prior selections, make recommendations, such as collaborative filtering.

Ranking: given candidate responses for a recommendation or retrieval task, do the fine grained ranking.

Cleaning up Wikipedia: the Wikipedia would be an amazing linguistic resource if only, ...

Recognised Problems, cont.

Machine translation (MT): automatically convert text to another language,

Cross language IR (CLIR): from queries in one language probe document collection in another.

Email spam detection: recognising spam email.

Trust and authority: measures of document/author quality in terms authority and trust based on content, links, citation, history, *etc.*

Communities: analysis and identification of online communities.

Video and Image X: most of the above applied to video and images.

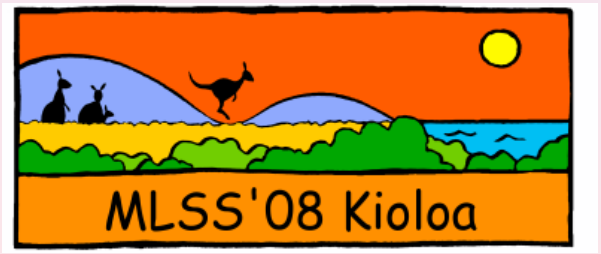
Outline

And so ends Part 1. Next we look at specific problems and algorithms.

1. Formal Natural Language
2. Document Processing
3. Document Analysis

Latent Variable Models for Document Analysis

Wray Buntine
National ICT Australia (NICTA)



Part I

Problems and Methods

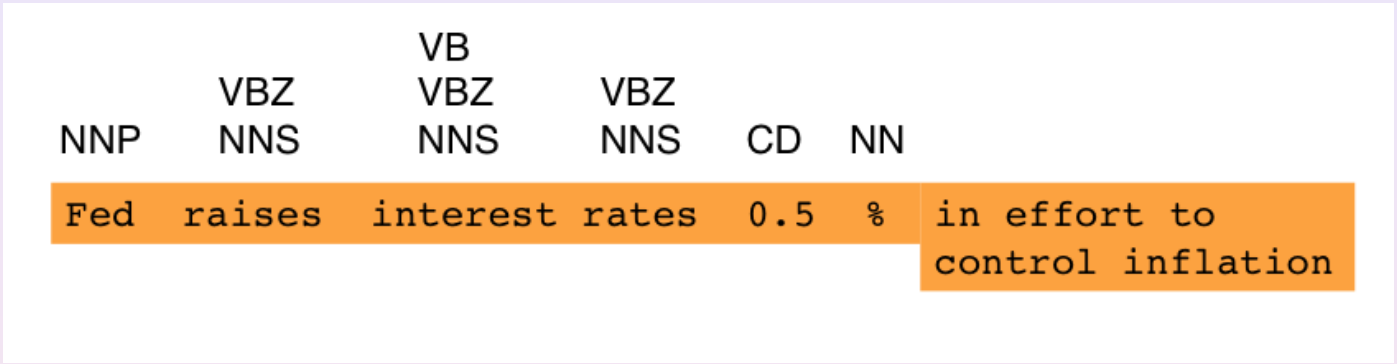
Outline

We review some key problems and key algorithms using latent variables.

Outline

While not the best algorithm for part-of-speech tagging, it is useful to illustrate the methods.
Reference: Manning and Schütze, chaps 9 and 10.

Parts of Speech, Revisited

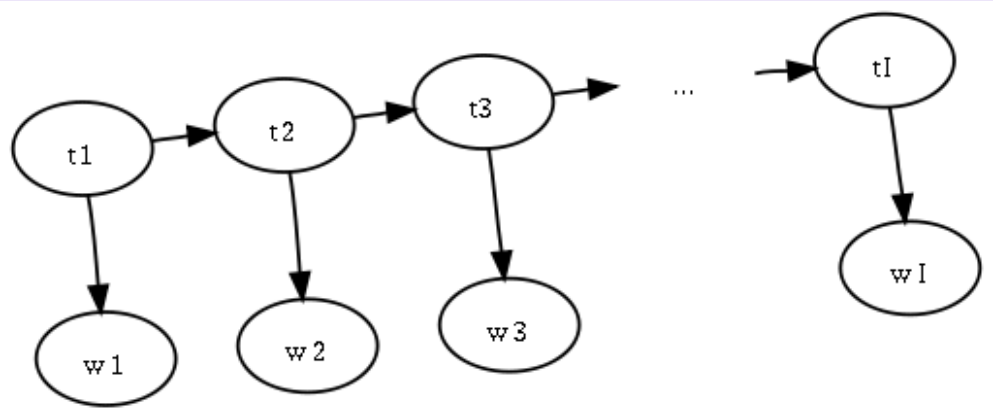


- A set of candidate POS exist for each word. taken from a dictionary or lexicon. Which is the right one in this sentence?
- Lets take some fully tagged data, where the truth is known, and use statistical learning.
- A standard notation for representing tags , in this example, is:

  `Fed/NNP raises/VBZ interest/NNS rates/NNS 0.5/CD %/% ...` *(in effort to control inflation.)*

Outline

Markov Model with Known Tags



- There are I words. $w_i = i$-th word. $t_i =$ tag for i-th word.
- Our 1st-order Markov model in the figure shows which variables depend on which.
- The $(i+1)$-th tag depends on the i-th tag. The i-th word depends on the i-th tag.
- Resultant formula for $p(t_1, t_2, t_3, ..., t_N, w_1, w_2, w_3, ..., w_N)$ is

$$p(t_1) \prod p(t_i|t_{i-1}) \prod p(w_i|t_i)$$

Fiitting Markov Model with Known Tags

- Have $p(t_1, t_2, t_3, ..., t_N, w_1, w_2, w_3, ..., w_N)$ is

$$p(t_1) \prod_{i=2,...,I} p(t_i|t_{i-1}) \prod_{i=1,...,I} p(w_i|t_i)$$

- Have K distinct tags and J distinct words.
- Use $p(t_i|t_{i-1}) = a_{t_{i-1},t_i}$, $p(t_1) = c_{t_1}$, $p(w_i|t_i) = b_{t_i,w_i}$.
- **a** and **b** are probability matrices whose columns sum to one.
- Collecting like terms

$$\prod_k c_k^{S_k} \prod_{k_1,k_2} a_{k_1,k_2}^{T_{k_1,k_2}} \prod_{k,j} b_{k,j}^{W_{k,j}}$$

where T_{k_1,k_2} is count of times tag k_2 follows tag k_1, and
$W_{k,j}$ is count of times tag k assigned to word j, and
S_k is count of times sentence starts with tag k.

Fiitting Markov Model with Known Tags, cont.

- Standard maximum likelihood methods apply, so these parameters **a** and **b** become their observed proportions:
  - a_{k_1,k_2} is proportion of tags of type k_2 when previous was k_1,
  - $b_{k,j}$ is proportion of words of type j when tag was k,
- Thus $a_{k_1,k_2} = \frac{T_{k_1,k_2}}{\sum_{k_2} T_{k_1,k_2}}$, $b_{k,j} = \frac{W_{k,j}}{\sum_j W_{k,j}}$, $c_k = \frac{S_k}{\sum_k S_k}$.
- Note we have many sentences in the training data, and each one has a fresh start, so c_k is estimating from all those initial tags in sentences.
- As is standard when dealing with frequencies, we can smooth these out by adding small amounts to the numerator and denominator to make all quantities non-zero.

Comments

- In practice, the naive estimation of **a** and **b** works poorly because we never have enough data. Most words occur infrequently, so we cannot get good tag statistics for them.
- Kupiec (1992) suggested grouping infrequent words together based on their pattern of candidate POS. This overcomes paucity of data with a reasonable compromise.
  - So "red" and "black" can both be NN or JJ, so they belong to the same *ambiguity class*.
  - Ambiguity classes not used for frequent words.
- Unknown words are also a problem. A first approximation is to assign unknown words with first capitals to NP.

Outline

Estimating Tags for New Text

- We now fix the Markov model parameters **a**, **b** and $\vec{c}$.
- We have a new sentence with I words $w_1, w_2, ..., w_I$. How do we estimate its tag set?
- We ignore the lexical contraints for now (*e.g.*, "interest" is VB, VBZ or NNS), and fold them in later.
- Task so described is:

$$\vec{t} \;=\; \text{argmax}_{\vec{t}}\; p\left(\vec{t}, \vec{w} \,|\, \mathbf{a},\, \mathbf{b}, \vec{c}\right)$$

where the probability is as before.

Estimating Tags for New Text, cont.

Wish to solve

$$\text{argmax}_{\vec{t}}\ p(t_1) \prod_{i=2,\ldots,I} p(t_i|t_{i-1}) \prod_{i=1,\ldots,I} p(w_i|t_i)$$

The task is simplified by the fact that knowing the value for tag t_N splits the problem neatly into parts, so define

$$m(t_N) = \max_{t_1,\ldots,t_{N-1}|t_N}\ p(t_1) \prod_{i=2,\ldots,N} p(t_i|t_{i-1}) \prod_{i=1,\ldots,N-1} p(w_i|t_i)$$

and we get the recursion:

$$\begin{aligned} m(t_1) &= p(t_1) \\ m(t_{N+1}) &= \max_{t_N} \quad p(t_{N+1}|t_N)p(w_N|t_N)m(t_N) \end{aligned}$$

While computing this, we also record, for each t_{N+1} the value of $(t_1, \ldots, t_N)$ that yields a maxima, called the *backtrace*.

Estimating Tags for New Text, cont.

We apply this incrementally, building up a contingent solution from left to right. This is called the **Viterbi algorithm**, first developed in 1967.

1. Initialise $m(t_1)$.
2. For $i = 2, ..., I$, compute $m(t_i)$, then store the backtrace for each t_i.
3. At the end, I, find $\text{argmax}_{t_I} m(t_I)$, and get its backtrace.

This technique is an example of dynamic programming.

Comments

- What about lexical contraints, *e.g.*, "interest" is either VB, VBZ or NNS?
- With the ambiguity classes above, and with the individual words, we just assign zero's to $b_{k,j}$ for j the index of the word.

Outline

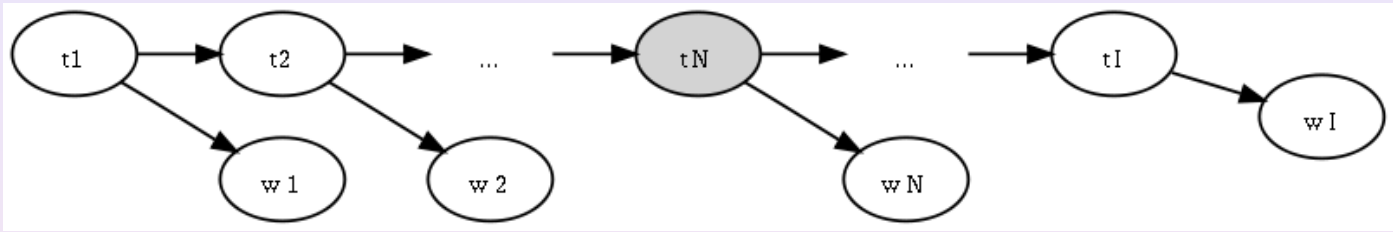
Estimating Tag Probabilities

- We again fix the Markov model parameters **a**, **b** and $\vec{c}$.
- We have a new sentence with I words $w_1, w_2, ..., w_I$. We've got the most likely tag set using the Viterbi algorithm. What's the uncertainty here?
- Task can be described as: find the tag probabilities for t_N.

$$p(t_N|\vec{w}) \propto \sum_{\vec{t}/t_N} p\left(\vec{t}, \vec{w} \,|\, \mathbf{a}, \mathbf{b}, \vec{c}\right)$$

where the probability is as before.

Estimating Tag Probabilities, cont.



Wish to compute

$$p(t_N, \vec{w}) = \sum_{\vec{t}/t_N} p(t_1) \prod_{i=2,\ldots,I} p(t_i|t_{i-1}) \prod_{i=1,\ldots,I} p(w_i|t_i)$$

Note we have:

$$p(t_N|w_1, \ldots, w_{N-1}) = \sum_{t_1,\ldots,t_{N-1}} p(t_1) \prod_{i=2,\ldots,N} p(t_i|t_{i-1}) \prod_{i=1,\ldots,N-1} p(w_i|t_i)$$

$$p(w_{N+1}, \ldots, w_I|t_N) = \sum_{t_{N+1},\ldots,t_I} \prod_{i=N+1,\ldots,I} p(t_i|t_{i-1}) \prod_{i=N+1,\ldots,I} p(w_i|t_i)$$

$$p(t_N, \vec{w}) = p(t_N|w_1, \ldots, w_{N-1}) p(w_{N+1}, \ldots, w_I|t_N) p(w_N|t_N)$$

Estimating Tag Probabilities, cont.

The quantities $p(t_N|w_1,...,w_{N-1})$ and $p(w_{N+1},...,w_I|t_N)$ are traditionally called $\alpha(t_N)$ and $\beta(t_N)$ respectively.

As with the Viterbi, a recursion exists:

$$p(t_N|w_1,...,w_{N-1}) = \sum_{t_{N-1}} p(t_N|t_{N-1})p(w_{N-1}|t_{N-1})p(t_{N-1}|w_1,...,w_{N-2})$$

$$p(w_{N+1},...,w_I|t_N) = \sum_{t_{N+1}} p(t_{N+1}|t_N)p(w_{N+1}|t_{N+1})p(w_{N+2},...,w_I|t_{N+1})$$

Compute the first with a forward pass in N, compute the second with a backward pass in N. Hence computing these probabilities is called the *Forward-Backward* algorithm. Complexity is $O(I\,K^2)$.

Outline

Fitting with Unknown Tags

- We don't always have a large quantity of text tagged with POS. So we would like to try and improve the estimates of the model using untagged or partially tagged data.
- So the problem becomes, estimate $\mathbf{a}$, $\mathbf{b}$ and $\vec{c}$ given the sequence $w_1, w_2, ..., w_l$ but no tags.
- The case with partial tags can be folded in later.
- This problem, where the tags are unknown initially is called a *hidden Markov model* (HMM).

A Little Bit of Magic

- We will use some probability function $q(\vec{t})$ in our solution as a device. This represents *some* valid probability over the tags.
  NB. it can be represented by a large parameter vector.
- For brevity, denote **a**, **b** and $\vec{c}$ by a single parameter vector $\vec{\theta}$.
- Consider the function $Q(\vec{\theta}, q())$ given by

$$\begin{aligned} &= \log p(\vec{w}|\vec{\theta}) - KL\left(q(\vec{t}) \,||\, p(\vec{t}|\vec{w},\vec{\theta})\right) \\ &= E_{q(\vec{t})}\left(\log p(\vec{t},\vec{w}|\vec{\theta})\right) + I(q(\vec{t})) \end{aligned}$$

- A simple expansion of $KL()$ and $I()$ shows the two forms are equal.

A Little Bit of Magic, cont.

- Consider the function $Q(\vec{\theta}, q())$ given by

$$\begin{aligned} &= \log p(\vec{w}|\vec{\theta}) - KL\left(q(\vec{t}) \,||\, p(\vec{t}|\vec{w},\vec{\theta})\right) \\ &= E_{q(\vec{t})}\left(\log p(\vec{t},\vec{w}|\vec{\theta})\right) + I(q(\vec{t})) \end{aligned}$$

- Maximise this w.r.t. $\vec{\theta}$ and $q()$ jointly.
- By the first equation, this holds when $q(\vec{t}) = p(\vec{t}|\vec{w},\vec{\theta})$, and then $Q(\vec{\theta}, q()) = \log p(\vec{w}|\vec{\theta})$.
- By the second equation, this holds if we solve:

$$\text{argmax}_{\vec{\theta}}\ E_{q(\vec{t})}\left(\log p(\vec{t},\vec{w}|\vec{\theta})\right) \ .$$

- Thus, iterating these two steps will achieve the maximum likelihood solution $\text{argmax}_{\vec{\theta}}\ \log p(\vec{w}|\vec{\theta})$.

Fitting with Unknown Tags, cont.

- We wish to evaluate

$$\text{argmax}_{\vec{\theta}}\ E_{q(\vec{t})}\left(\log p(\vec{t},\vec{w}|\vec{\theta})\right)\ .$$

- From before, substituting in for **a**, **b** and $\vec{c}$, wish to maximise w.r.t. $\vec{\theta}$

$$E_{q(\vec{t})}\left(\sum_k \log c_k^{S_k} + \sum_{k_1,k_2} \log a_{k_1,k_2}^{T_{k_1,k_2}} + \sum_{k,j} \log b_{k,j}^{W_{k,j}}\right)\ ,$$

$$= \sum_k E_{q(\vec{t})}(S_k)\log c_k + \sum_{k_1,k_2} E_{q(\vec{t})}(T_{k_1,k_2})\log a_{k_1,k_2} + \sum_{k,j} E_{q(\vec{t})}(W_{k,j})\log b_{k,j}\ .$$

- Thus we don't actually need to evaluate the full distribution $q(\vec{t}) = p(\vec{t}|\vec{w},\vec{\theta})$, we just need $p(t_N|\vec{w},\vec{\theta})$ and $p(t_{N-1},t_N)|\vec{w},\vec{\theta})$.

Baum-Welch Algorithm

Putting it all together.

1. From the current solution for **a**, **b** and $\vec{c}$, perform the Forward-Backward algorithm to compute $p(t_N|w_1, ..., w_{N-1})$ and $p(w_{N+1}, ..., w_I|t_N)$.
2. From these, compute $p(t_N|w_1, ..., w_I)$ and $p(t_{N-1}, t_N|w_1, ..., w_I)$.
3. Hence compute $E_{q(\vec{t})}(S_k)$, $E_{q(\vec{t})}(T_{k_1,k_2})$ and $E_{q(\vec{t})}(W_{k,j})$.
4. Now maximise for **a**, **b** and $\vec{c}$ to get the next iteration.

$$\sum_k E_{q(\vec{t})}(S_k) \log c_k + \sum_{k_1,k_2} E_{q(\vec{t})}(T_{k_1,k_2}) \log a_{k_1,k_2} + \sum_{k,j} E_{q(\vec{t})}(W_{k,j}) \log b_{k,j} \ .$$

Comments

- Unfortunately HMM training doesn't work too well for the POS problem.
- Perhaps this is because we are fitting a joint model $p(\vec{t}, \vec{x}|\vec{\theta})$ rather than a conditional model $p(\vec{t}|\vec{x}, \vec{\theta})$.
- So lets investigate conditional models.

Outline

Conditional Fitting with Unknown Tags

- So the problem is to estimate a model for $\vec{t}$ given the sequence $w_1, w_2, ..., w_I$ but no tags.
- We no longer have $p(w_i|t_i)$, rather we want a discriminative model, something like $p(t_i|w_i)$, but also $p(t_i|t_{i-1})$,
- One approach, called the conditional random field (CRF) is to fold them in together to get:

$$p(\vec{t}\,|\,\vec{w}, \mathbf{a}, \mathbf{b}, \vec{c}) \;\propto\; \exp\left(\sum_i a_{t_{i-1},t_i} + \sum_i b_{t_i,w_i} + \sum_i c_{t_i}\right)$$

- Compare this with our HMM model:

$$p(\vec{t}, \vec{w}|\mathbf{a}, \mathbf{b}, \vec{c}) \;=\; c_{t_1} \prod_i a_{t_{i-1},t_i} \prod_i b_{t_i,w_i}$$

Conditional Fitting with Known Tags, cont.

- The conditional random field has:

$$p(\vec{t} \mid \vec{w}, \mathbf{a}, \mathbf{b}, \vec{c}) \propto \exp\left(\sum_i a_{t_{i-1},t_i} + \sum_i b_{t_i,w_i} + \sum_i c_{t_i}\right)$$

- We need a normalising constant, Z, a function of $\mathbf{a}$, $\mathbf{b}$ and $\vec{c}$.
- Compute this incrementally, rather like a forward pass of the Forward-Backward algorithm.

$$\begin{aligned} Z_1(t_1) &= 1 \\ Z_N(t_N) &= \sum_{t_{N-1}} Z_{N-1}(t_{N-1}) \exp\left(a_{t_{N-1},t_N} + b_{t_{N-1},w_{N-1}} + c_{t_{N-1}}\right) \\ Z &= \sum_{t_N} Z_N(t_N) \exp\left(b_{t_N,w_N} + c_{t_N}\right) \end{aligned}$$

Conditional Fitting with Known Tags, cont.

We have to use gradient based algorithms to fit this as there are no closed forms.

- Lets look at the likelihood to maximise $\log p(\vec{t}|\vec{w},\vec{\theta})$.

$$\sum_k S_k c_k + \sum_{k_1,k_2} T_{k_1,k_2} a_{k_1,k_2} + \sum_{k,j} W_{k,j} b_{k,j} - \log Z$$

- Note **a**, **b** and $\vec{c}$ are no longer probability matrices and vectors.
- Now it happens that

$$\frac{\partial \log Z}{\partial a_{k_1,k_2}} = E_{p(\vec{t}|\vec{w},a,b,\vec{c})}(T_{k_1,k_2})$$
$$\frac{\partial \log Z}{\partial b_{k,j}} = E_{p(\vec{t}|\vec{w}a,b,\vec{c})}(W_{k,j})$$

  These expected values can be computed by a variant of the forward-backward algorithm, as before.

- Thus we have all the derivatives of the likelihood

Comments

- Conditional training with some unknown tags also works, but is more complicated again.
- In principle. you can now use any features, not just the words $\vec{w}$. People use:
  - capitalisation, all-caps, use of non-alphabetic letters,
  - presence of prefixes and suffixes,
  - properties of surrounding words,
  - match of words to different gazetteers.
- In this case, the performance is very dependent on the choice of features!

Discrete Components Analysis – DCA

Wray Buntine
(Work with help from Kimmo Valtonen, Aleks Jakulin, Sami Perttu, CoSCo, ...)

Complex Systems Computation Group (CoSCo)
University of Helsinki & HIIT
NICTA

March 6, 2008

Overview

- **Background.**
- Quick in Depth Look.
- History, Religion, Interpretations.
- Models and Correspondences.
- Theory of Algorithms.
- Comparative Experiments†.
- Use in Search‡.

† See `http://www.componentanalysis.org`.

‡ See `http://cosco.hiit.fi/search/wikipedia400-1205`

Bag of words as a Sparse Discrete representation for text

A page out of Dr. Zeuss's *The Cat in The Hat*:

> So, as fast as I could, I went after my net. And I said, "With my net I can bet them I bet, I bet, with my net, I can get those Things yet!"

In the *bag of words* representation as *word* (*count*):

> after(1) and(1) as(2) bet(3) can(2) could(1) fast(1) get(1) I(7) my(3) net(3) said(1) so(1) them(1) things(1) those(1) went(1) with(2) yet(1) .

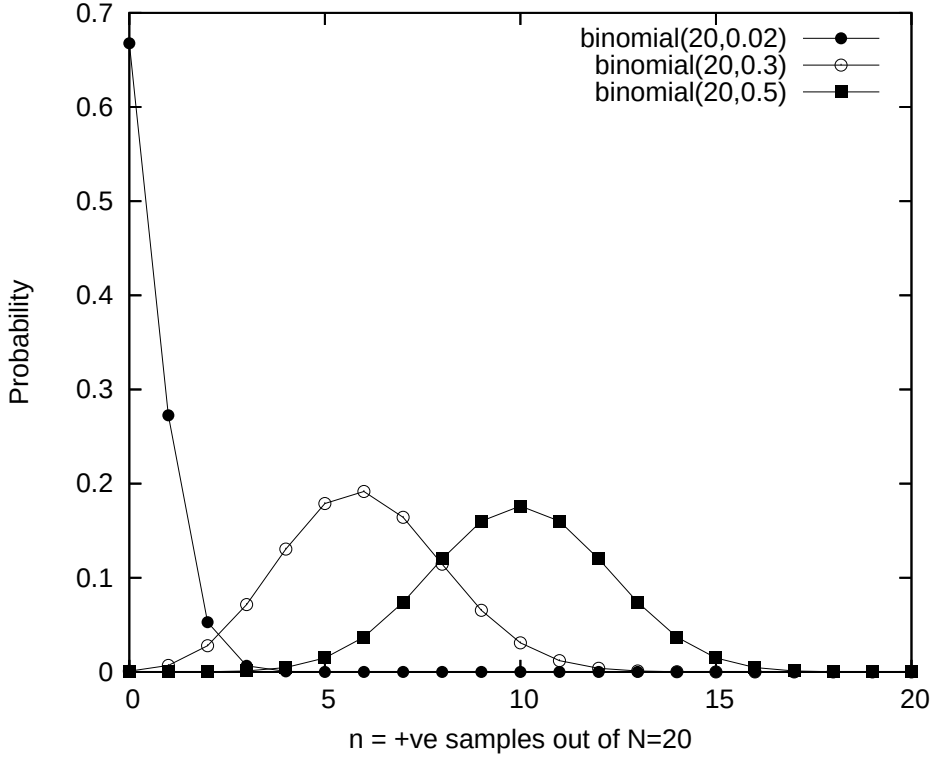
Notes:

- For the Reuters RCV1 collection from 2000: $I \approx 800k$ documents, $J \approx 400k$ different words (excluding those occurring few times), $S \approx 300Mb$ words total.
- Represent as sparse matrix/vector form with integer entries.
- Deleting words occurring less than < 50 times can shrink word dimension J by order of magnitude.

Principal Components Analysis (PCA)

- Invented by Karl Pearson, in 1901.
- Also known as Karhünen-Loève transform or Hotelling transform in image analysis, and latent semantic analysis (LSA) for text.
- Primarily used for *dimensionality reduction* prior to some other statistical processing.
- Has guarantees in terms of minimising least squares error in approximation.
- Also has a Gaussian interpretation using latent variables (Tipping and Bishop, 1999).
- Standard algorithm is to run an SVD and to throw away all but the top K eigenvectors and values, or to use sparse LAPACK style tools to extract just the top K without a full SVD.

Approximating Discrete Data



The plot shows different binomials in both its Gaussian regime and its Poisson regime.

Lesson: discrete data is only Gaussian-like in some contexts. When there are a lot of zeros, it is not Gaussian-like.

PCA: Issues

- PCA (with least squares or alternatively the Gaussian) is known to cause trouble in some contexts:
  - when values occur on a boundary (i.e., Gaussians don't admit boundaries).
  - with discrete and sparse values (i.e., outside the Gaussian regime).
- PCA has no realistic probabilistic interpretation in the discrete case.
  - Can be OK as a dimensionality reduction tool.
  - But no easy way to do probabilistic inference with the model.

NB. The ICA community can supply other issues!

Discretizing PCA

Consider the Gaussian interpretation. For data of dimension J being mapped down to K components:

$$\begin{aligned} \vec{m} &\sim Gaussian(0, \mathbf{I_K}) \ , \\ \vec{x} &\sim Gaussian(\mathbf{\Omega}\vec{m} + \vec{\mu}, \mathbf{I_J}\sigma) \ . \end{aligned}$$

where $\mathbf{\Omega}$ maps from the K dimensional space to the J dimensional space.

Let us contruct a discrete version by mapping the distributions. We could modify this, for instance, to:

$$\begin{aligned} \vec{m} &\sim Dirichlet(\vec{\alpha}) \\ \vec{x} &\sim Multinomial(\mathbf{\Omega}\vec{m}, L) \ , \end{aligned}$$

where L is the total count. Alternatively, a Gamma-Poisson combination is possible.

Independent Components Analysis (ICA)

- Invented by Herault and Jutten in 1986.
- Intended for *blind source separation*, separation of independent signals in some data.
- Used for *dimensionality reduction* as well, based on the observation that PCA can perform poorly.
- Standard algorithm is the FastICA algorithm, developed by Hyvärinen and Oja.
- One interpretation is, using an unknown distribution $U()$:

$$\begin{aligned} m_k &\sim U() && \text{for } k = 1, \ldots K \\ \vec{x} &= \mathbf{\Omega}\vec{m} + \epsilon, \end{aligned}$$

  where ϵ is considered insignificant noise, and ignored.

ICA: Issues

- The key equation with ϵ insignificant

$$\vec{x} \;=\; \mathbf{\Omega}\vec{m} + \epsilon,$$

  can behave poorly in the *sparse discrete case* since $\vec{x}$ is mostly zeroes and ones, so the equation is now discrete.

- In effect, when the *dynamic range* is effectively 2-valued, we want to be carefully measuring the error $(\vec{x} - \mathbf{\Omega}\vec{m})$, thus the noise is significant.

- When the dynamic range of the data is significantly greater 2-valued, this effect becomes less significant.

- Data usually pre-processed beforehand with PCA to remove "noise". Furthermore, discrete data often turned into real values before hand, for instance using `tf*idf` scores.

Discretizing ICA

Consider the interpretation, with an unknown distribution $U()$:

$$\begin{aligned} m_k &\sim U() && \text{for } k = 1, \ldots K \\ \vec{x} &= \mathbf{\Omega}\vec{m} + \epsilon, \end{aligned}$$

where ϵ is considered some insignificant noise.

Lets make this more robust by allowing significant noise. Replace the second equation with:

$$\mathbb{E}_{\vec{x} \sim p(\vec{x}|\vec{m},U)}\left[\vec{x}\right] = \mathbf{\Theta}\vec{m} \ .$$

Requires defining the probability distribution for $\vec{x}$. We can use Poissons or multinomials in the discrete case.

Overview

- Background.
- **Quick in Depth Look.**
- History, Religion, Interpretations.
- Models and Correspondences.
- Theory of Algorithms.
- Comparative Experiments.
- Use in Search.

Example application: Wikipedia, July 2004

- We built a $K = 100$ component model of $I = 290k$ web pages from the English-language Wikipedia* dated July 2004.
- DCA is run on bags of lemmatised words.
- Word lemmas grouped into noun, verb, adjective, adverb and other word classes discarded.
- Thus data vector $\vec{w}$ is a sparse vector of dimension about $J = 300k$, partitioned into four groups, with about $240k$ nouns. But with only about 20-1000 non-zero entries.
- Table on the next page gives one row from Θ for a component. Just the high frequency terms are listed.

*`http://en.wikipedia.org`

NOUNS					
mythology	0.03337	God	0.02048	name	0.014747
goddess	0.012911	spirit	0.012639	legend	0.0087992
myth	0.0070882	demons	0.006807	Sun	0.0060099
Temple	0.0054717	deity	0.0054247	Bull	0.0051629
Dragon	0.0051379	Maya	0.0051243	King	0.00512
Sea	0.0049453	Norse	0.0044707	horse	0.0044592
symbol	0.0042196	animals	0.0040112	fire	0.0039879
hero	0.0038755	Romans	0.0038696	Apollo	0.0037588
Lion	0.0036306	Earth	0.0035993	giant	0.0035076
VERBS					
called	0.034078	said	0.031081	See	0.029521
given	0.0269	associated	0.024591	According	0.021724
represented	0.020964	known	0.018896	could	0.017499
made	0.016952	depicted	0.01524	appeared	0.014662
ADJECTIVES					
Greek	0.091163	ancient	0.055393	great	0.02853
Egyptian	0.028071	Roman	0.025783	sacred	0.020446
ADVERBS					
sometimes	0.058205	often	0.048815	so	0.046594

This component: "Mythology"

Use different methods to gain understanding of the components.

Distinctive phrases: noun phrases that have a high Bayes factor with the component (used part-of-speech system plus noun phrase "patterns" like NN, AN, etc.).

> God; name; spirit; goddess; Greek mythology; Holy Spirit; Greek word; Polynesian mythology; Golden Age; evil spirits;

High ranked document titles: use the component as the topic in topic-specific page rank (Richardson and Domingos, 2001), to get page rankings based on topical content and links.

> Greek mythology; Roman mythology; Celtic mythology; Norse mythology; Underworld; Aztec mythology;

Wikipedia, December 2005

- We built a $K = 400$ component model of $I = 980k$ web pages from the English-language Wikipedia dated December 2005.
- DCA is run on bags of lemmatised words organised by part of speech, and the external URLs at the page.
- Words or URLs occurring less than 10 times in corpus ignored, leaving feature dimension about $J = 1000k$.
- Used Gamma-Poisson model with *sparse* Gamma component priors (i.e., about 90% of component values are zero). Hyper-parameters for the component priors fitted (gradient ascent with trust regions).
- Thus each document is a sparse vector (perhaps 300 entries).
- Fitting used Gibbs with Rao-Blackwellisation, 1000 major cycles, on a dual CPU Opteron (64-bit) with 4Gb memory. About 6 days.
  NB. thus we use Gibbs for approximation, not for estimation!

Results: `http://cosco.hiit.fi/search/wikipedia400-1205`

DCA Text Applications

Usage:

- Feature discovery for text classification;
- unsupervised synonym and/or ontology discovery;
- language models for text (unsupervised models on relations such as verb-subject) for use in a semantic parser;
- language models for information retrieval (unsupervised models on bag-of-words); and

Major problems in development (e.g., McCallum *et al.*):

Compounds: low order correlations not modelled, e.g., "New York".

Hierarchies: organisation required for use.

Overview

- Background.
- Quick in Depth Look.
- **History, Religion, Interpretations.**
- Models and Correspondences.
- Theory of Algorithms.
- Comparative Experiments.
- Use in Search.

History

- Soft clustering, "grade of membership" (GOM), Woodbury and Manton, 1982.
- Admixture modelling in statistics, (198?).
- Hidden facets in image interpretation, Non-negative Matrix Factorization (NMF), Seung and Lee, 1999.
- Probabilistic Latent Semantic Analysis (PLSI), topics in text, Hofmann, 1999.
- Admixture modelling, fully Bayesian, population structure from genotype data, Pritchard, Stephens and Donnelly, 2000.
- Latent Dirichlet Allocation (LDA) Blei, Ng and Jordan, 2001. Variant of Pritchard *et al.* Introduced mean-field algorithm.
- Multi-aspect modelling: various 2001-2003.
- Gamma-Poisson model (GaP), Canny 2004 (extension of NMF).
- ...

Religious convictions

All manor of statistical beliefs and practices are permitted:

- Maximum likelihood or Kullback-Leibler divergence.
- Exponential family likelihoods or Bregman divergence.
- Regularised maximum likelihood.
- Bayesian and empirical Bayesian.
- ... *did I miss any?*

Catalogue of interpretations

- Approximating a discrete matrix as a product of lower dimension matrices.
- Multinomial version of the Gaussian interpretation of PCA (i.e., Roweis or Bishop style PCA).
- Multi-aspect modelling or soft clustering (documents have proportion/grade of membership).
- Admixture modelling (forming a mixture by mixing means, not distributions).
- Variation of ICA (independent component analysis) suitable for discrete data.
- Hidden topics for the individual words in a document, itself in a collection.

Viewing Components at the Word Level

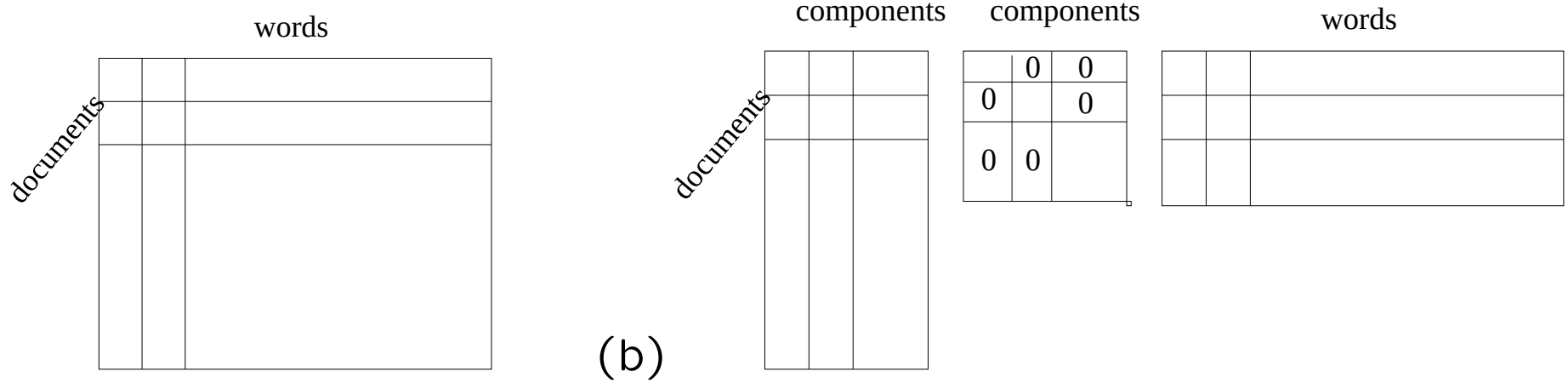
Figure 8 from Blei, Ng, and Jordan, 2003.

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

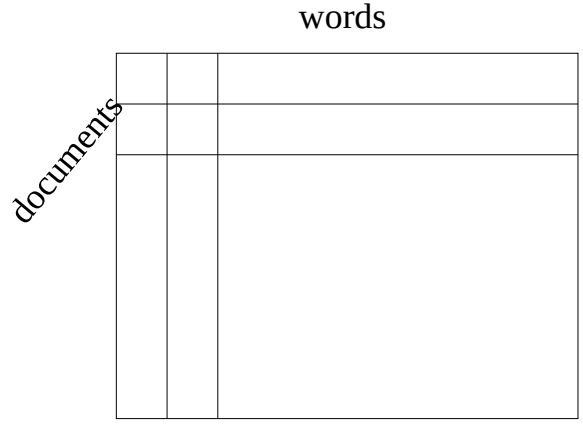
Matrix factorisation

- Standard SVD, decomposes matrix $D = U^T \Lambda V$.
- U and V have rows mutually orthogonal and normalised using L_2 norm.
- Λ is diagonal, and is scaling.
- Principal components analysis (PCA) zeros the smaller values in Λ (as "noise"), making U and V lose rows.
- Want (b) to approximate (a) in a pairwise least squares sense, for fixed K number of components.



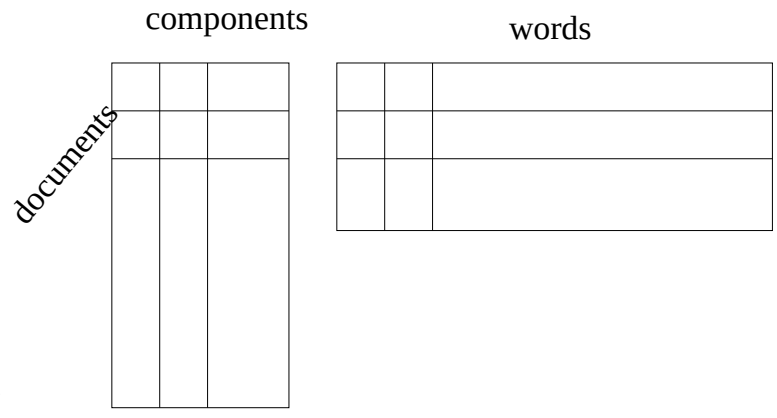
Matrix factorisation: statistical view

- Zero smaller eigenvalues in Λ as noise. U and V lose corresponding rows.
- Let $U^T \leftarrow U^T\Lambda$. It is `documents` $\times$ `components`, projects document words into lower (?) dimensional space.
- Leave V alone, still normalised by row. It is `components` $\times$ `words` of "characteristic documents".
- Thus (a) becomes (c) approximately: distance is L_2, get PCA; divergence is K-L, get DCA.

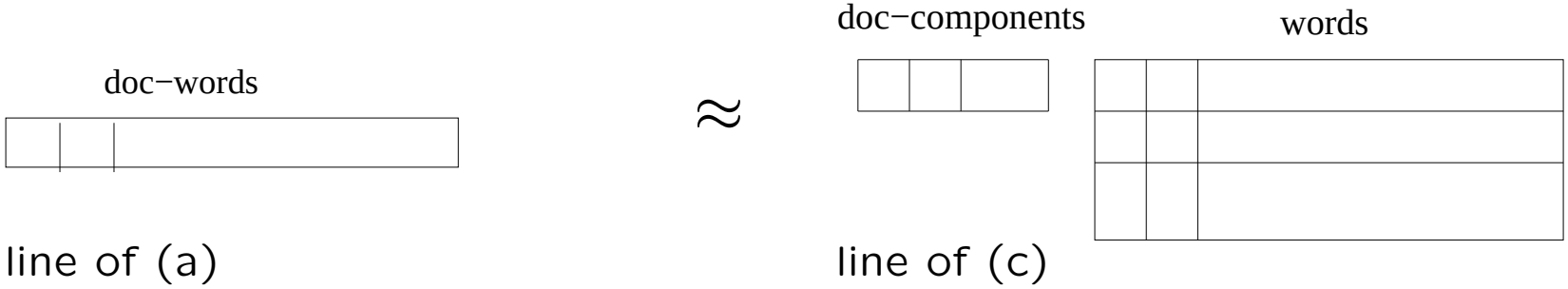


(a)

(c)



Independent components + single document model



line of (a) line of (c)

- `doc-words` is the data vector $\vec{w}$. Given.
- `doc-components` is the hidden variable (a vector) $\vec{l}$. l_k is interpreted as number of words in component k.
- The `components` $\times$ `words` matrix is the *model matrix*, $\mathbf{\Theta}$. We normalise it along rows. Each row is the word rates/probabilities for one component.
- The relationship between LHS (data) and RHS (estimate from hidden data) can be approximated in expectation, i.e., using a general **admixture**.

$$\mathrm{Exp}_{p(\vec{w}\,|\,\mathbf{\Theta},K,\ldots)}(\vec{w}) \;=\; \vec{l}\mathbf{\Theta}$$

- **Compare with ICA**, which uses $\vec{w} = \vec{l}\cdot\mathbf{\Theta}$, impossible with discrete data.

Overview

- Background.
- Quick in Depth Look.
- History, Religion, Interpretations.
- **Models and Correspondences.**
- Theory of Algorithms.
- Comparative Experiments.
- Use in Search.

Gamma-Poisson model

- I is documents, J is words/features, K is components.
- For each document, a hidden variable (a vector) $\vec{l}$. For each component $k = 0, ..., K-1$

$$l_k \sim \text{Gamma}(\alpha_k, \beta_k) \ .$$

- The $K \times J$ *model matrix* is $\mathbf{\Theta}$. Entries $\theta_{k,j}$. We normalise it along rows.
- For each feature/word j

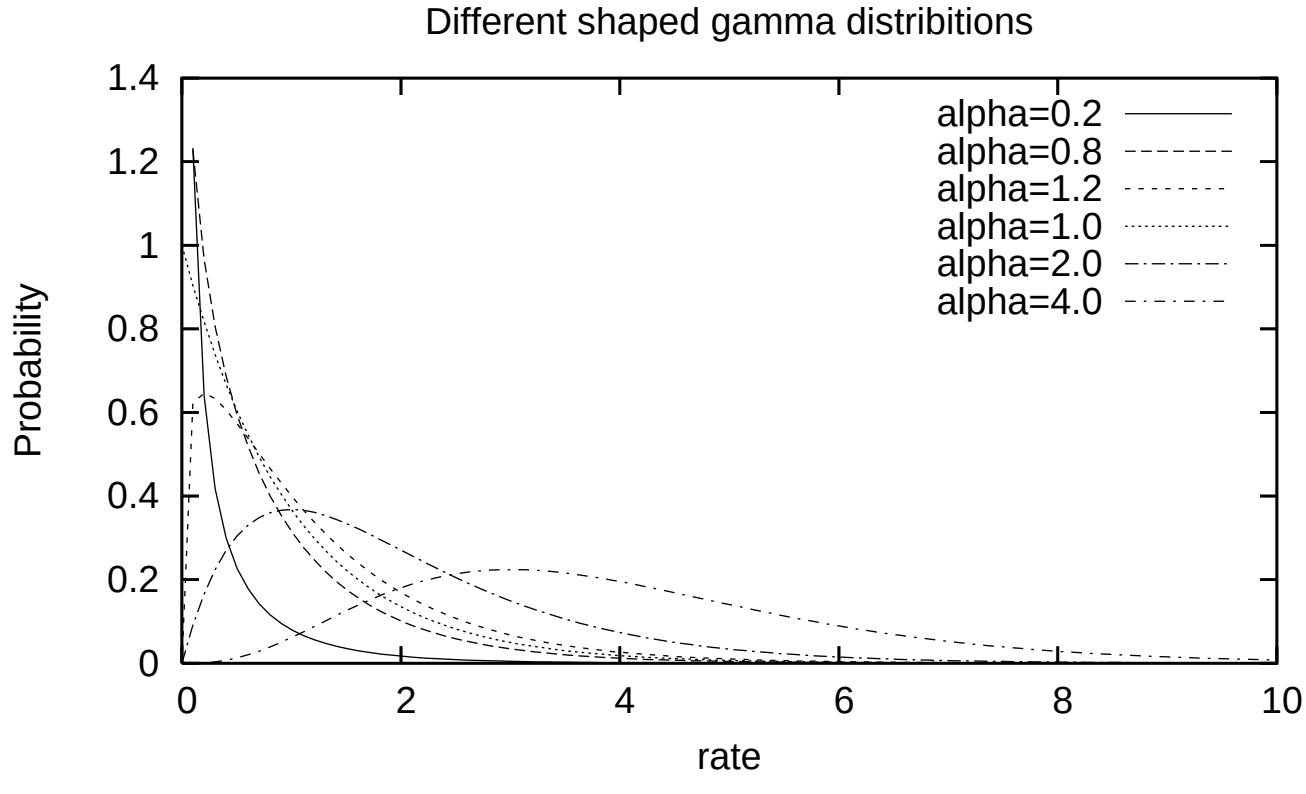
$$w_j \sim \text{Poisson}\left(\sum_k l_k \theta_{k,j}\right)$$

Gamma components

- components

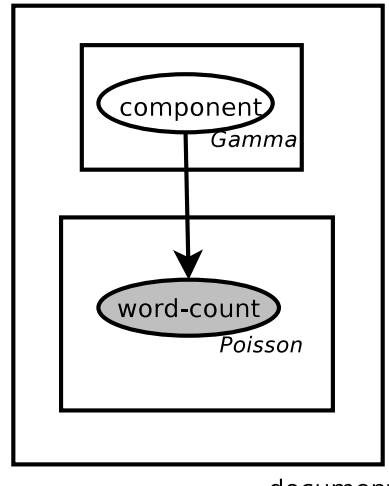
  $l_k \sim \text{Gamma}(\alpha_k, \beta_k)$.

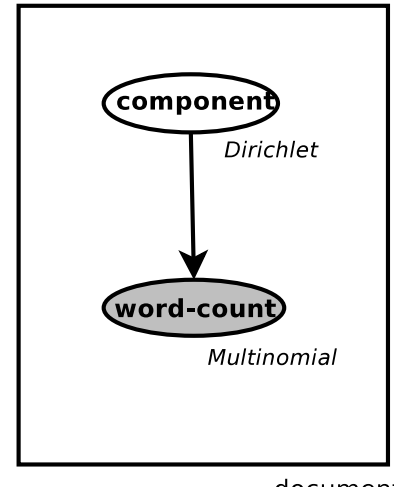
- For $\alpha_k > 4$ approaches Gaussian.



DCA Versions

- On left is GaP model from Canny 2004
- Normalise to get the right, the multinomial PCA (assumes β of gammas is constant).





Gamma-Poisson model, cont.

- Introduced by Canny, SIGIR 2004.
- Originally given as a version of non-negative matrix factorisation (NMF) of Lee and Seung (1999).
- Correspondence with ICA noted, Canny 2004.
- NMF optimization criteria somewhat different.
- Proof that NMF equations arrive at a maximum likelihood solution for the Gamma-Poisson (or equivalently, the Dirichlet-multinomial model given next) in Buntine and Jakulin, 2005, and Goutte and Gaussier, SIGIR 2005.

Dirichlet-multinomial model

Start with the Gamma-Poisson. Assume the β_k are constant, given by β. Total of $\vec{l}$ is $\sum_k l_k$. Define hidden variable (a vector) $\vec{m} = \vec{l} / \left(\sum_k l_k\right)$.

- Components now jointly distributed:

$$\begin{aligned} \sum_k l_k &\sim \text{Gamma}\left(\sum_k \alpha_k, \beta\right) \\ \vec{m} &\sim \text{Dirichlet}_K(\vec{\alpha}) \end{aligned}$$

- Using the word total $L = \sum_j w_j$, get

$$\begin{aligned} L &\sim \text{Poisson}\left(\sum_k l_k\right) \\ \vec{w} &\sim \text{multinomial}\left(\left\{\sum_k m_k \theta_{k,j} \,:\, j\right\}, L\right) \end{aligned}$$

- $(\sum_k l_k, L)$ independent of rest so can be ignored.

Dirichlet-multinomial model, cont.

- Probabilistically completed version of PLSI introduced by Hofmann 1999.
- Admixture modelling of Pritchard *et al.* 2000.
- Latent Dirichlet Allocation, Blei *et al.* 2001, replaces the multinomial on the bag by a sequence of discretes.
- Multinomial PCA by Buntine 2002.

Variations of the Gamma-Poisson

Poisson Admixture	Poisson-Multinomial	Tagged Words
$l_k \sim \text{Gamma}(\alpha_k, \beta_k)$ $w_j \sim \text{Poisson}\left(\sum_k l_k \theta_{k,j}\right)$	$\sum_k l_k \sim \text{Gamma}\left(\sum_k \alpha_k, \beta\right)$ $\vec{m} \sim \text{Dirichlet}_K(\vec{\alpha})$ $L \sim \text{Poisson}\left(\sum_k l_k\right)$ $\vec{w} \sim \text{multinom.}\,(\vec{m}\boldsymbol{\Theta},\, L)$	$l_k \sim \text{Gamma}(\alpha_k, \beta_k)$ $v_{j,k} \sim \text{Poisson}(l_k \theta_{k,j})$ $w_j = \sum_k v_{j,k}$

Variations of the Dirichlet-multinomial

Bagged Admixture	Tagged Words	Sequential Admixture
$\vec{m} \sim \text{Dirichlet}_K(\vec{\alpha})$ $\vec{w} \sim \text{multinom.}(\vec{m}\boldsymbol{\Theta}, L)$	$\vec{m} \sim \text{Dirichlet}_K(\vec{\alpha})$ $\vec{c} \sim \text{multinom.}(\vec{m}, L)$ $\vec{v_{\cdot,k}} \sim \text{multinom.}(\vec{\theta_{k,\cdot}}, c_k)$ $w_j = \sum_k v_{j,k}$	$\vec{m} \sim \text{Dirichlet}_K(\vec{\alpha})$ $k_l \sim \text{discrete}(\vec{m}\boldsymbol{\Theta})$ for $l = 1, ..., L$

where $\vec{\theta_{k,\cdot}}$ denotes the vector $\{\theta_{k,j} : j\}$, and $\vec{v_{\cdot,k}}$ denotes the vector $\{v_{j,k} : j\}$.

DCA interpretations for bag-of-words

For a given document:

- Components l_k are expected count of words in component k.
- Normalised to m_k, are proportion of words in component k. For clustering, $m_k = 0, 1$. Hence method is called *multifaceted clustering*.
- Hidden variables $v_{j,k}$ means tagging each word by a particular component.
- Compare with regular clustering: one hidden topic per document, *versus* one hidden topic per word in the document.
- Word probabilities given by $\sum_k m_k \theta_{k,j}$, hence called *admixtures*.

Other Correspondences

PLSI, Hofmann: regularised max.likelihood version, uses multinomial model for words, but no model for components, just regularisation.

LDA, Blei *et al.*: Dirichlet-multinomial model, made sequential (i.e., sequence of words, not bag of words).

Overview

- Background.
- Quick in Depth Look.
- History, Religion, Interpretations.
- Models and Correspondences.
- **Theory of Algorithms.**
- Comparative Experiments.
- Use in Search.

Likelihoods for Gamma-Poisson

Full likelihood for a document with hidden variables $\vec{l}$ is

$$p(\vec{w}, \vec{l} \,|\, \text{Gamma-Poisson}, K, \vec{\alpha}, \vec{\beta}, \mathbf{\Theta})$$
$$= \prod_k \left(\frac{\beta_k^{\alpha_k}}{\Gamma(\alpha_k)} l_k^{\alpha_k - 1} e^{-\beta_k l_k} \right) \; e^{-\left(\sum_k l_k\right)} \prod_j \frac{1}{w_j!} \left(\sum_k l_k \theta_{k,j} \right)^{w_j}$$

Full likelihood for a document with hidden variables $\vec{l}$ and $\mathbf{v}$ is

$$p(\vec{w}, \mathbf{v}, \vec{l} \,|\, \text{Gamma-Poisson}, K, \vec{\alpha}, \vec{\beta}, \mathbf{\Theta})$$
$$= \prod_k \left(\frac{\beta_k^{\alpha_k}}{\Gamma(\alpha_k)} l_k^{\alpha_k - 1} e^{-\beta_k l_k} \right) \; e^{-\left(\sum_k l_k\right)} \prod_j \prod_k \frac{(l_k \theta_{k,j})^{v_{j,k}}}{v_{j,k}!}$$

- The $(\sum_k l_k ...)^{w_j}$ and $l_k^{v_{j,k}}$ terms in full likelihood means EM algorithm cannot be used.
- However EM can apply to maximise $p(\vec{w}_1, \ldots, \vec{w}_I, \vec{l}_1, \ldots, \vec{l}_I \,|\, \text{DCA}, K, \vec{\alpha}, \vec{\beta}, \mathbf{\Theta})$.
- Gibbs applies directly, but careful, see later.
- Mean field applies too.

Mean Field

General case for exponential family in Ghahramani and Beal, NIPS 2000. Works for Dirichlet-multinomial and Gamma-Poisson.

Approximate posterior on $\vec{l}$ and $\mathbf{v}$ by jointly independent form $q(\vec{l})q(\mathbf{v})$.

$$
\begin{aligned}
q(\vec{l}) &\longleftrightarrow \frac{1}{Z_l} \exp\left(E_{q(\mathbf{v})}\left\{\log p\left(\vec{w}, \vec{l}, \mathbf{v} \mid \mathbf{\Theta}, \vec{\alpha}, \vec{\beta}, K\right)\right\}\right) \\
q(\mathbf{v}) &\longleftrightarrow \frac{1}{Z_v} \exp\left(E_{q(\vec{l})}\left\{\log p\left(\vec{w}, \vec{l}, \mathbf{v} \mid \mathbf{\Theta}, \vec{\alpha}, \vec{\beta}, K\right)\right\}\right) \quad ,
\end{aligned}
$$

Functional form is therefore:

$$
\begin{aligned}
l_k &\sim \text{Gamma}(a_k, b_k) \\
\{v_{j,k} : k = 1, \ldots, K\} &\sim \text{multinomial}(\{n_{j,k} : k = 1, \ldots, K\}, w_j)
\end{aligned}
$$

Likelihood with Components Marginalised

Full likelihood for a document with hidden variables $\mathbf{v}$ (marginalise out $\vec{l}$): is

$$p(\vec{w}, \mathbf{v} \,|\, \mathsf{DCA}, K, \vec{\alpha}, \vec{\beta}, \mathbf{\Theta})$$

$$= \prod_k \frac{\beta_k^{\alpha_k}}{\Gamma(\alpha_k)} \prod_k \frac{\Gamma\left(\alpha_k + \sum_j v_{j,k}\right)}{(1+\beta_k)^{\alpha_k + \sum_j v_{j,k}}} \prod_j \prod_k \frac{\theta_{k,j}^{v_{j,k}}}{v_{j,k}!}$$

- To use, resample each word's component assignment sequentially. (e.g., decrement $v_{j,k_{old}}$ and increment $v_{j,k_{new}}$).

- Marginalising out variables in a problem prior to applying Gibbs sampling is called *Rao-Blackwellisation*.

- We can also marginalise out $\mathbf{\Theta}$ from the full joint likelihood to get a closed formula for $p(\vec{w}_1, \ldots, \vec{w}_I, \mathbf{v}_1, \ldots, \mathbf{v}_I \,|\, \mathsf{DCA}, \vec{\alpha}, \vec{\beta}, K)$.

- Rao-Blackwellisation applies to Gibbs for the Dirichlet-multinomial, Griffiths and Steyvers, 2004, and Gamma-Poisson.

Computational Issues

I	documents
J	lexemes
K	components
S	words in corpus

- Mean field and Gibbs on bag of words are O(JK+SK) in time per cycle and O(IK+JK) in space.
- Mean field DCA at dimension K a few times slower than incremental (i.e., fast) PCA at same dimension.
- Gibbs with Rao-Blackwellisation is O(SK) in time per cycle and O(S+JK) in space. Less then others.
- Gibbs with Rao-Blackwellisation much better time and space for very small "documents", e.g., analysis of sentences or noun-verb pairs, etc.
- Minka's Expectation-Propagation (EP) is an extra order of magnitude so is not practical.

Computational Issues, cont.

- While doing Gibbs sampling, we do *not* really want Gibbs sampling:
  - Components defined symmetrically, thus a true posterior mean would smear out all parameters to a bland uniform.
  - We should really be doing millions of major cycles, we just do 1000's when doing discovery (as opposed to hypothesis testing).

  i.e., we use Gibbs as an algorithm to generate an estimate, not as a method to do sound statistical inference.

- Text data as bags of words and the document intermediate variables (proportions $\vec{m}$, topic assignments $\vec{k}$, etc.) don't need to be kept in main memory and can be streamed over using `mmap()`.

- Memory constraints are thus two copies (current value and sufficient statistics) for $\mathbf{\Theta}$, and a typical desktop could handle $J = 200k$ and $K = 400$.

DCA Component Priors

Hidden component vector $\vec{l}$. For each component $k = 0, ..., K-1$

$$l_k \sim \text{Gamma}(\alpha_k, \beta_k) \ .$$

Component prior parameters are $\vec{\alpha}$ and $\vec{\beta}$.

- Can estimate $\vec{\alpha}$ from the data. Its posterior has a conjugate distribution to a Gamma$(\alpha, 1)$, and is well-behaved. Roughly corresponds to setting α_k to the geometric mean (mean taken in log space) of the l_k's in the data.

- Can estimate $\vec{\beta}$ from the data. This has a gamma distribution, and is well-behaved. Corresponds to setting $\beta_k = \alpha_k / \langle l_k \rangle$.

- In practice, with many features (J large), the data swamps the component prior and everything is pushed rapidly towards a Gamma model with no "prior data", *i.e.*, $\alpha_k \to 0$, and β_k set as above.

- Thus we usually fix $\alpha_k = \frac{2}{K}$, but set a hard minimum of 0.01 or so.

- A Dirichlet Process with parameter α is approximated when $\alpha_k = \frac{\alpha}{K}$.
  NB. the prior expected number of components is approximately α.

Conditional Gamma-Poisson model

Gamma-Poisson model results in many exponentially small components for each document. Seems wasteful. The conditional Gamma-Poisson model zeros these. Usually done as a polishing step without loss in likelihood.

- Hidden variable $\vec{l}$. For each component $k = 0, ..., K-1$

$$\begin{array}{rcll} l_k & = & 0 & \text{with probability } \rho_k \\ l_k & \sim & \text{Gamma}(\alpha_k, \beta_k) & \text{otherwise} \end{array}$$

- Otherwise same as Gamma-Poisson model.

- For each feature/word j

$$w_j \sim \text{Poisson}\left(\sum_k l_k \theta_{k,j}\right)$$

- For text data, using a fixed $\alpha_k = 1$ and letting β_k be fit as per previous, ρ_k often above 0.95. When $\rho_k \to 1.0$, the component can be deleted.

Dirichlet Process models for component priors

- A Dirichlet Process model with parameter α for the component assignments and proportions $\vec{m}$ is approximated by using a K dimensional Dirichlet with parameters all equal to $\frac{\alpha}{K}$.

- Alternatively one can use the truncated Dirichlet Process model (see Blei *et al.*, various).

- As a side effect, one gets an estimate for K by watching the number of unused components during a run.

- While this is clearly more efficient than the previous harmonic mean method, it appears the models so developed may be inferior to others.

- More experimental work needed here.

Overview

- Background.
- Quick in Depth Look.
- History, Religion, Interpretations.
- Models and Correspondences.
- Theory of Algorithms.
- **Comparative Experiments.**
- Use in Search.

Comparative Performance of Component Priors

- Gamma-Poisson with β constant and Dirichlet-multinomial very close in performance, *i.e.*, as per theory!

- In practice, with many features (J large), the data swamps the component prior and everything is pushed rapidly towards a Gamma model with no "prior data", *i.e.*, $\alpha_k \to 0$, and β_k set as above.

- Conditional Gamma components can be done as polishing, or done from the beginning: sample β_k's and ρ_k's dynamically via Gibbs and estimate α_k's via constrained gradient ascent. Generally better likelihoods will be achieved, with many component values zeroed, and some components eliminated.

Comparative Performance: Algorithms

- Use a Dirichlet-discrete model with $\beta = 1$ and $\alpha_k = 0.5$.
- A Fisher Dirichlet prior is used for documents.
- Perplexity results are measured on a hold-out set.
- Algorithms:

  Mean field with inner loop convergence: As done in Minka's mean field implementation available in Matlab, reported by some. Inner loop of the mean field algorithm repeated to convergence on first cycle.

  Mean field: As described.

  Direct Gibbs: As described.

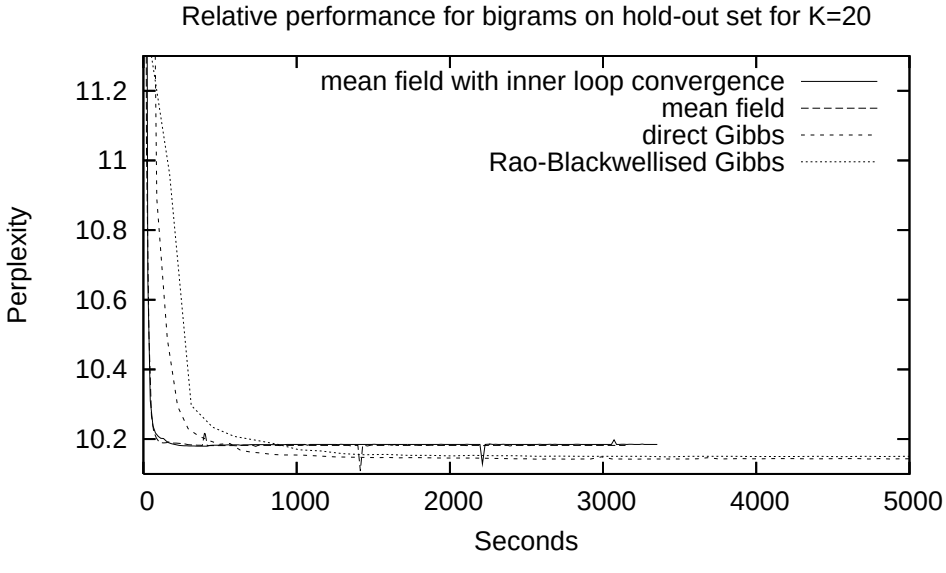
  Rao-Blackwellised Gibbs: With adjustments to ensure unbiases estimates of the log-probabilities. This adds about 50% time penalty, thus time reported below discounts this.

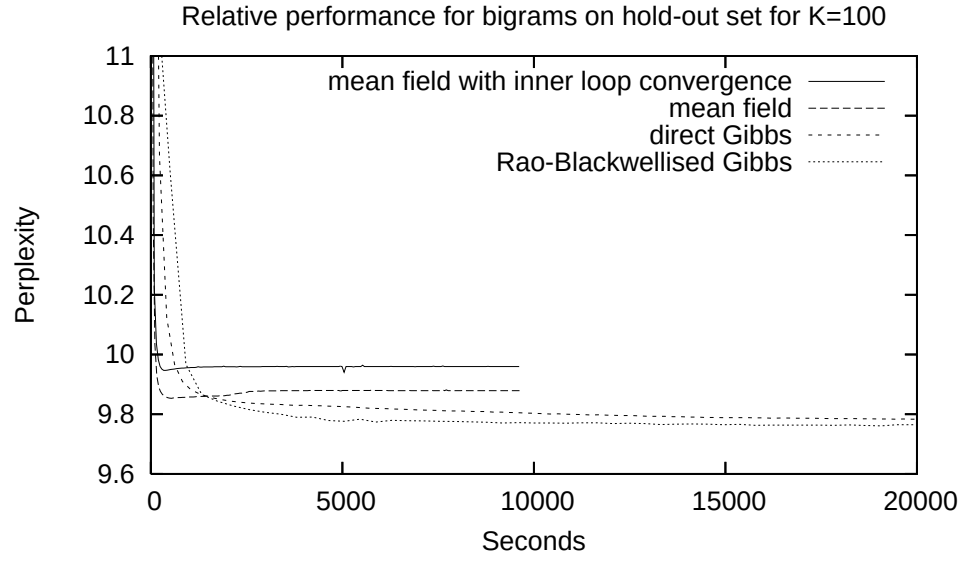
Comparative Performance: Data

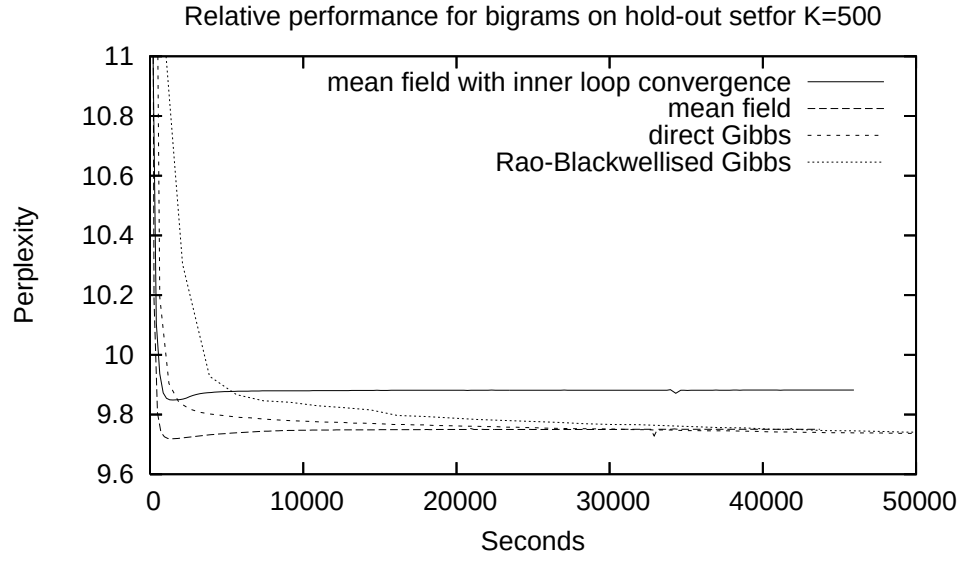
Data from the Reuters RCV1 corpus (Volume 1: English Language, 1996-08-20 to 1997-08-19).

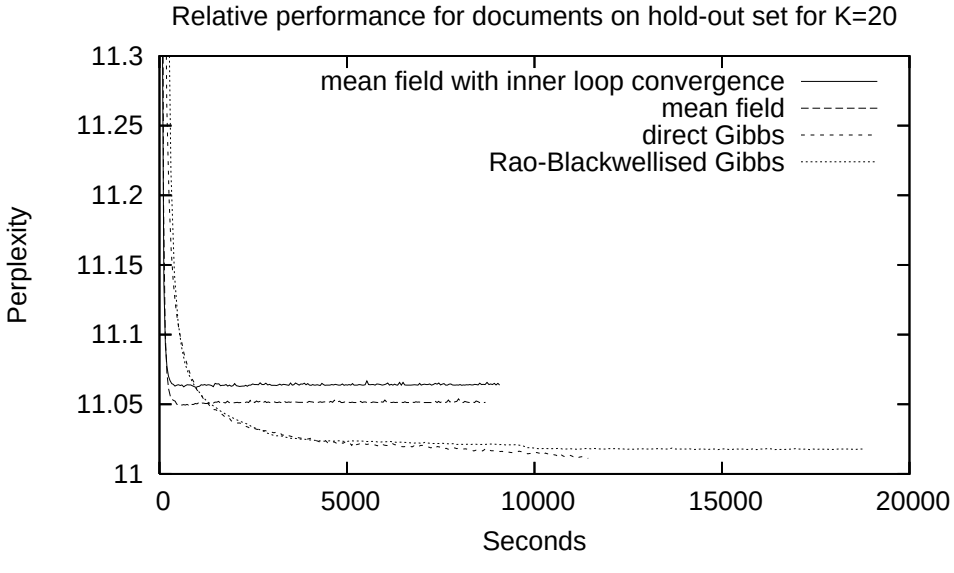
Bigrams: verb-noun bigrams in sentences, extracted from a lemmatized version of the full corpus. Single "document" has all noun lemma used in the vicinity of a given verb lemma. training $I = 30000$, $J = 20000$, and $K = 20$, 100, 500, $S/I \approx 700$. Testing is on random selection of $20,000$ held out.

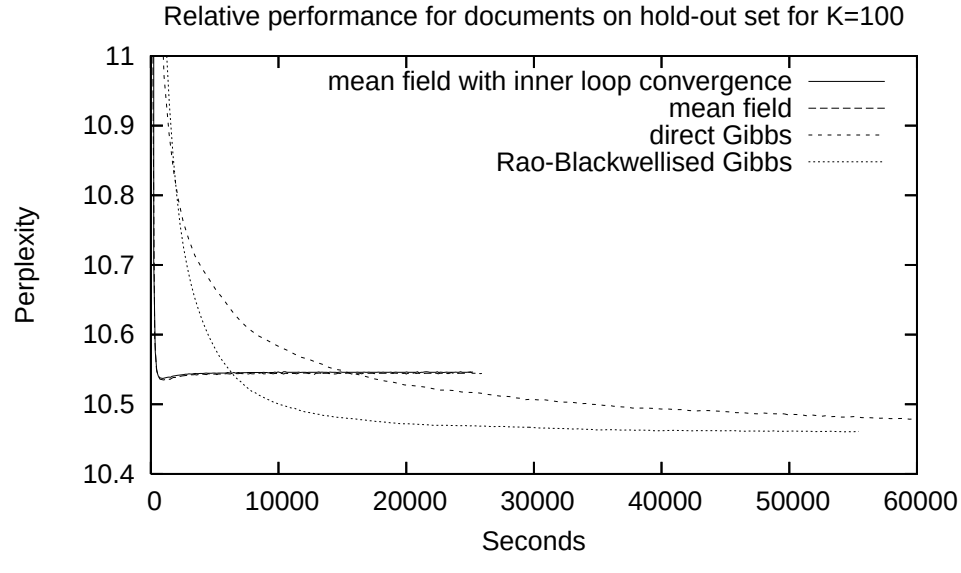
Documents: uses the first $150,000$ documents in the corpus, but only records the most frequent $50,000$ lemmatised nouns extracted from a lemmatized version of the corpus. training $I = 100000$, $J = 50000$, and $K = 20$, 100, $S/I \approx 100$. Testing on last 50000.


Relative performance for bigrams on hold-out set for K=20
mean field with inner loop convergence
mean field
direct Gibbs
Rao-Blackwellised Gibbs
Perplexity
Seconds
11.2
11
10.8
10.6
10.4
10.2
0
1000
2000
3000
4000
5000


Relative performance for bigrams on hold-out set for K=100
mean field with inner loop convergence
mean field
direct Gibbs
Rao-Blackwellised Gibbs
Perplexity
Seconds
11
10.8
10.6
10.4
10.2
10
9.8
9.6
0
5000
10000
15000
20000


Relative performance for bigrams on hold-out setfor K=500
Perplexity
Seconds
11
10.8
10.6
10.4
10.2
10
9.8
9.6
0
10000
20000
30000
40000
50000
mean field with inner loop convergence
mean field
direct Gibbs
Rao-Blackwellised Gibbs


Relative performance for documents on hold-out set for K=20
Perplexity
Seconds
mean field with inner loop convergence
mean field
direct Gibbs
Rao-Blackwellised Gibbs
11.3
11.25
11.2
11.15
11.1
11.05
11
0
5000
10000
15000
20000


Relative performance for documents on hold-out set for K=100
Perplexity
Seconds
mean field with inner loop convergence
mean field
direct Gibbs
Rao-Blackwellised Gibbs
11
10.9
10.8
10.7
10.6
10.5
10.4
0
10000
20000
30000
40000
50000
60000

Results summary

- Rao-Blackwellised Gibbs version has the fastest convergence (unless unbiased estimates of log-probabilities are required).
- Doing inner convergence of the loop in the mean field algorithm can damage performance.
- Mean field converges faster, and sometimes much faster. In one case, mean field also equals the others in perplexity.
- The direct Gibbs version sometimes catches up with the Rao-Blackwellised Gibbs version over time.
- The mean field version appears to start over-fitting quite quickly.

Overview

- Background.
- Quick in Depth Look.
- History, Religion, Interpretations.
- Models and Correspondences.
- Theory of Algorithms.
- Comparative Experiments.
- **Use in Search.**

Use in ALVIS

- Predefined topics (e.g., MESH, ODP, Dewey Decimal) best when they apply.
- Otherwise, use DCA to develop topics. (Hierarchical version TBD top-down).
- Use topics to allow mixed topic-browse and search.
- Allow users to enter block of text to indicate topical-preference, then combine this as the topical aspect of a fused language model.

Search engine topics: good coverage of a domain

Ads and banners; Advertising agency's appointments; Advertising and marketing; Affiliate program; America Online (AOL); Ask Jeeves and Google Answers; Blogs; Book search; Customer service; Dates; Desktop search; Domain name registration; DoubleClick, Inc.; Earnings of companies; E-mail spam; Film industry; Forums and discussions; Games; Google co-founders; Interactive media and advertising; Internet; Legal; Local search; Maps; Microsoft and Google; Mobile communication; Music search; News; Online multimedia; Organizations and standards; Paid inclusions and search; People; Privacy issue; "Acceptable" SEO; Regions; Research; Search Engine Marketing; Search engine optimization; Search engine optimization and marketing; Search marketing; Security issues; Shopping Search; Stock market; User interfaces; Users, people, communities; Web advertising; Wikipedia/Deja;

Overview of techniques

- Extracting relevant named-entities using an IR system: extension of a relevance language model:

$$p(name|query) = \sum_{d \in docs} p(d|query)p(name|d, query)$$

  $p(d|query)$ got from the IR score for a document. $p(name|d, query)$ is *ad hoc* based on location arguments.

- Extracting relevant topics using an IR system.

$$p(topic|query) = \sum_{d \in docs} p(d|query)p(topic|d)$$

  Efficient to compute (similar to IR retrieval) since topic probabilities ($p(topic|d)$) are sparse.

Overview

- Quick in Depth Look.
- History, Religion, Interpretations.
- Models and Correspondences.
- Theory of Algorithms.
- Comparative Experiments†.
- Use in Search‡.
- **The End.**

† See `http://www.componentanalysis.org` and `http://wikipedia.hiit.fi`

‡ See `http://cosco.hiit.fi/search/wikipedia400-1205`