

Applications of AI

1st hour: Natural Language Processing

2nd hour: Semantic Modelling
Guest lecturer: Dr Daniel Angus

Tutorial 10

- Updated to include the equation and graph for the logistic/sigmoid function

Course evaluation

- SECaT – Student Evaluation of Course and Teacher
- First 10-15 minutes of next week's lecture

Final exam

- Wednesday 9 November 8:00am
- COMP3702
 - A-R 69-110
 - S-Z 67-141
- COMP7702
 - All 67-141

Are you prepared for your examinations?

Do you:

- Have your current student ID card?
- Know where your exam is?
- Know what materials you are permitted to bring to the exam? (check with your course coordinator)
- Have an approved / labelled calculator (where calculators are permitted)

For each exam, ensure you:

- Have rechecked the timetable for exam date, time and venue and checked your emails
- Have your current student ID card on hand, and ready to present on entry to the exam room – should you forget it, you should report to the Student Centre before your exam
- Have spare pencils and pens, as well as any permitted materials
- Arrive at your exam venue 15 minutes before the scheduled start of the exam



Natural language processing

In which we see how to make use of the copious knowledge that is expressed in natural language

Chapter 22

Overview: aims

- Know what n-gram models are
- Understand the application of n-gram character and word models
- Understand the basic principles of Information Retrieval (in relation to unstructured text data)

Overview: topics

- Natural language processing
- Probabilistic methods
- N-gram models
- Information Retrieval

Natural Language

- Spoken: ~100,000 years ago
- Writing: ~7000 years ago
- Unique to humans
 - Many abilities needed for language found in a variety of species, but not the full complexity of human language
- Turing test
- Communication
- Acquisition of Information

Natural language processing

- History:
 - Cryptography, computer translation
 - Turing Test (1950)
 - Machine translation (1950s)
 - Linguistics, meanings, grammars (1960s-70s)
 - ELIZA (Rogerian psychotherapist) (1960s)
 - ...
 - Answering questions – Watson
 - Search – Google
 - Spell check – Microsoft

Natural language processing

- Goals:
 - To create computer applications that can process large amounts of text sensibly
 - To accomplish human-like language processing
- Applications
 - Information retrieval
 - Information extraction
 - Question-answering
 - Summarisation
 - Machine translation
 - Dialogue

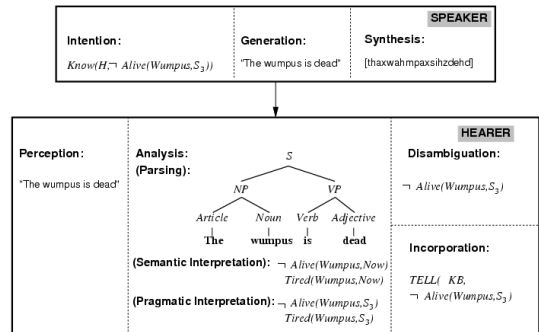
Natural language processing

- Difficulties:
 - Ambiguity
 - Context
 - Uncertainty
 - Reasoning
 - Social interaction

Natural language processing

- Approaches:
 - Complete syntactic and semantic parsing (ch 23), sentences are built from words according to a syntax (grammar), and carry meaning determined from the constituent words, or
 - Corpus-based (ch 22), exploiting volume of data to infer language models, and meaning of constituent words/phrases

Logical language processing (ch 23)



Probabilistic language models

- Given a word sequence, what is the next word?
- Probability Concepts:
 - Prior probability
 - Conditional probability
 - Independence
 - Bayes' rule

Probabilistic language models

- Probability $P(w)$ for a word sequence $w = w_1, w_2, \dots, w_k$
- Using the chain rule:

$$P(w_1, w_2, \dots, w_k) = P(w_1)P(w_2 | w_1)P(w_3 | w_1, w_2) \dots P(w_k | w_1, \dots, w_{k-1})$$
- We want to find: $P(w_n | w_1, \dots, w_{n-1})$

N-gram Language Models

- Assume that each word only depends on a short linear history
- Markov assumption: the probability of a word depends only on the immediately preceding words (independence assumption)
- An n-gram model is a Markov chain of order n-1

Building an n-gram model

- N=1 -> Uni-gram model
 - Probability of word = Frequency of words
 - $P(w) = \text{Count of word} / \text{total number of words}$
- N=2 -> Bi-gram model
 - Probability of word given previous word = Frequency of words given one previous other
 - $P(w_i | w_{i-1}) = \text{Count of word pair} / \text{count of previous other}$
- N=3 -> Tri-gram model
 - Probability of word given previous two words = Frequency of words given two previous others
 - $P(w_i | w_{i-2}, w_{i-1}) = \text{Count of word triple} / \text{count of previous word pair}$

N-gram models

- Defines a probability distribution over a set of strings
 - N-gram models assign probabilities to strings, by considering word-co-occurrence statistics – no explicit notion of grammar.
 - Uni-gram, $P(w)$:

$$P('w_1 w_2 w_3 \dots w_n') = \prod_i P(w_i)$$
 - Bi-gram, $P(w_i | w_{i-1})$:

$$P('w_1 w_2 w_3 \dots w_n') = \prod_i P(w_i | w_{i-1})$$
 - Tri-gram, $P(w_i | w_{i-1}, w_{i-2})$:

$$P('w_1 w_2 w_3 \dots w_n') = \prod_i P(w_i | w_{i-2:i-1})$$

Availability of data

- Russell and Norvig's book has 500,000 words
- And a lexicon of 15,000
 - a bi-gram model will need to estimate probabilities of 225 million different word pairs (at most 0.2% will have non-zero counts),
 - a tri-gram model needs probability estimation for 3,000 billion word triplets (at most 10-5% will have non-zero counts)
- Smoothing over zero counts (add-one, linear-interpolation)
- World-wide web is a rich source of text data

Limitations of n-gram models

- Long distance effects – Grammar
- Sparsity – Zipf's law
- Smoothing can help
- For accurate models you need millions of words

N-gram models

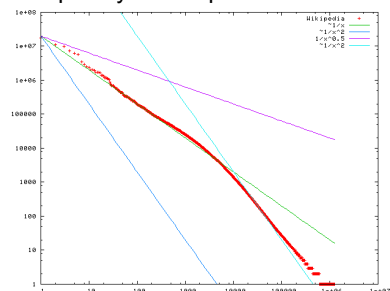
- Let's use Russell and Norvig's book to "train" them...
 - Uni-gram, $P(w)$;
 - $P('logical are as are confusion a may right tries agent goal the was diesel more object then information-gathering search is') = 10^{-59}$
 - Bi-gram, $P(w_i | w_{i-1})$;
 - $P('planning purely diagnostic expert systems are very similar computational approach would be represented compactly using tic tac toe a predicate') = 10^{-23}$
 - Tri-gram, $P(w_i | w_{i-1}, w_{i-2})$;
 - $P('planning and scheduling are integrated the success of naive bayes model is just a possible prior source by that time') = 10^{-10}$

Bag of words

- Another name for the unigram word model
- Word order does not matter
- Zellig Harris (1954): "language is not merely a bag of words but a tool with particular properties."
- Has been adapted for computer vision
 - e.g. for object categorisation, in which 'words' are different visual features

Zipf's law

- Word frequency in Wikipedia from 2006



<http://commons.wikimedia.org/wiki/File:Wikipedia-n-zipf.png>

Quality of n-gram models

- Word error rate
- How well can the model predict the next word
- Entropy
- Perplexity

N-gram character model

- Using characters instead of words
- Can be used for language identification
- The most probable language given the text

$$l^* = \arg \max_l P(l | c_{1:N})$$

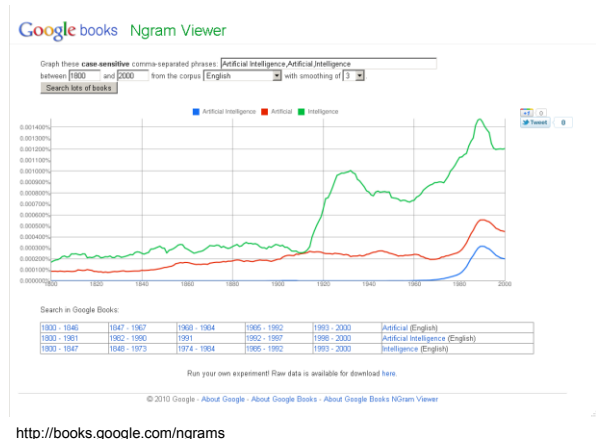
$$= \arg \max_l P(l)P(c_{1:N} | l)$$

$$= \arg \max_l P(l) \prod_{i=1}^N P(c_i | c_{i-2:i-1}, l)$$

Trigram character model

Applications of n-grams

- Language identification
- Text categorisation
- Topic segmentation
- Contextual spelling correction
- Speech recognition
- Language generation
- Machine translation



Information retrieval (1)

- IR is the task of finding documents (or records) relevant to a user
- Examples include internet search engines
- An IR system can be characterised by
 - A document collection (where each document is a web page, a paragraph or a document, ...)
 - A query (phrased in a specific query language)
 - A result set (the subset of documents relevant to the query)
 - A presentation of the result set (e.g. a ranked list of documents or some other visualisation of them)

Information retrieval (2)

- We want to compute $P(R=\text{true} | D, Q)$
 - R – relevance
 - D – document
 - Q – query
- We replace $R=\text{true}$ with r
- Once the query Q is chosen, we are interested in $P(r | D, Q)$ for each document.



Information retrieval (3)

- In practice, we want to find the top few documents with highest relevance given query Q.
- So we want to find the documents D_i which maximise or nearly maximise $P(r | D_i, Q)$.
- Not clear how you would estimate or model this, so we will try to restate this in other forms.



Probability of Relevance for the Document and the Query

$$\begin{aligned}
 &P(r | D, Q) \\
 &= \frac{P(D, Q | r)P(r)}{P(D, Q)} \text{ [Bayes' rule]} \\
 &= \frac{P(Q | D, r)P(D | r)P(r)}{P(D, Q)} \text{ [chain rule]} \\
 &= \frac{P(Q | D, r)P(r | D)P(D)}{P(D, Q)} \text{ [Bayes' rule]}
 \end{aligned}$$

- $P(Q|D,r)$: probability of the query phrase given the document, we might be able to model that.
- $P(r|D)$: relevance of the document irrespective of query – could go by some form of popularity.
- $P(D), P(D,Q)$: a nuisance.

Assumption 1: irrelevant documents and queries

- On irrelevant documents, the query and the document are assumed to be independent.
 - Not quite true, but a good approximation, more accurate when only a tiny fraction of the available documents are relevant to the query.
 - Relevant consequence: $P(Q|D, \neg r) = P(Q | \neg r)$
 - This now doesn't involve D, so we can ignore it in the maximisation - call it α (constant) if needed.

Probability Concepts

Product rule:

$$P(x_1, x_2, \dots, x_n) = P(x_n | x_{n-1}, x_{n-2}, \dots, x_1) P(x_{n-1} | x_{n-2}, \dots, x_1)$$

Chain rule:

$$\begin{aligned}
 P(x_1, x_2, \dots, x_n) &= \\
 P(x_n | x_{n-1}, x_{n-2}, \dots, x_1) &P(x_{n-1} | x_{n-2}, \dots, x_1) \dots P(x_2 | x_1) P(x_1)
 \end{aligned}$$

Bayes' rule

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}$$

Rearranging the term to be maximised

- If we're looking for D to maximise $P(r|D,Q)$, then we can equivalently look for D maximise the odds ratio:

$$\frac{P(r | D, Q)}{P(\neg r | D, Q)}$$
- This works out quite well, because then the $P(D)$ and $P(D,Q)$ terms cancel:

$$\frac{P(r | D, Q)}{P(\neg r | D, Q)} = \frac{P(Q | D, r)P(r | D)}{P(Q | D, \neg r)P(\neg r | D)}$$
- To produce answers from this, need to make some assumptions.

Assumption 2: queries vs relevant documents

- We have seen n-gram models for documents. Assume a uni-gram model for now. Also known as "bag of words" – ignores word order.
- So: build up a unigram model of each document D – estimate probabilities of each word within it.
- Then calculate the probability of the given query Q, on the basis of these (unigram model of query also).
- So now have a model which allows calculation of $P(Q|D,r)$.
- Using this model, $P(Q | D, r) = \prod_j P(Q_j | D, r)$
- This is a naive-Bayes type of assumption.

Assumption 3: determine generic relevance through popularity

- $P(r | D)$: relevance of a given document (without specifying a query).
- Some documents are more commonly useful than others.
- Use some measure of popularity to estimate this.
- Example: if the documents are web pages, count the number of links to the document.
- Example 2: if the documents are research papers, could count the number of times cited.
- Then divide by total number of links in relevant web space: answer can be interpreted as $P(r|D)$.
- $P(\neg r | D)$ is just $1 - P(r|D)$.

Maximisation

- So now we just need to choose documents which maximise the following (right-hand side).

$$\frac{P(r | D, Q)}{P(\neg r | D, Q)} = \frac{P(Q | D, r)P(r | D)}{\alpha(1 - P(r | D))}$$

Information Retrieval Example (1)

- Which of two documents is most relevant for the query "artificial intelligence"

	Document 1	Document 2
N	354	523
'artificial'	8	4
'intelligence'	16	23
Number of times relevant out of 1000 queries	3	4

$$\frac{P(r | D, Q)}{P(\neg r | D, Q)} = \frac{P(Q | D, r)P(r | D)}{\alpha(1 - P(r | D))}$$

Information Retrieval Example (2)

Document 1

- $P(Q|D1, r)$
 - = $8/354 * 16/354$
 - = 0.00102142
- $P(r|D1)$
 - = $3/1000$
- $1 - P(r|D1)$
 - = $997/1000$

Document 2

- $P(Q|D2, r)$
 - = $4/523 * 23/523$
 - = 0.00033634
- $P(r|D2)$
 - = $4/1000$
- $1 - P(r|D2)$
 - = $996/1000$

$$\frac{P(r | D, Q)}{P(\neg r | D, Q)} = \frac{P(Q | D, r)P(r | D)}{\alpha(1 - P(r | D))}$$

Information Retrieval Example (3)

Document 1

$$\frac{P(r | D1, Q)}{P(\neg r | D1, Q)} = \frac{0.00102 \times 0.003}{\alpha \times 0.997} = 3.07 \times 10^{-6} / \alpha$$

Document 2

$$\frac{P(r | D2, Q)}{P(\neg r | D2, Q)} = \frac{0.000336 \times 0.004}{\alpha \times 0.996} = 1.35 \times 10^{-6} / \alpha$$

Therefore, for the query "artificial intelligence" and using a uni-gram model, Document 1 is more relevant than Document 2

Evaluating information retrieval (1)

- Precision: proportion of hits that are relevant
 - $tp / (tp + fp)$, e.g. $100 / (100 + 5) \approx 0.95$
- Recall: proportion of relevant documents that are picked up
 - $tp / (tp + fn)$, e.g. $100 / (100 + 530) \approx 0.16$

	In the result set	Not in the result set
Relevant to query	100 (tp)	530 (fn)
Not relevant	5 (fp)	400,000 (tn)

Evaluating information retrieval (2)

- Other options for evaluation:
 - Reciprocal rank of 1st relevant result, e.g. 1/1, 1/3, etc – higher average scores are better.
 - Time to answer: how long an average user takes to find a satisfactory answer.

Refinements to document retrieval

- Stemming
 - Find basic word stems before processing
 - "couches"="couch", "recycling"="recycle"
- Synonyms
 - "sofa"="couch"
 - (but not always: e.g. "she chose to couch her question as a query". Usually sufficiently rare.)
- Spelling
 - Correction prior to processing query
 - "recycling"="recycling"
- Stop lists
 - Ignore extremely common (and extremely uncommon words)
 - "and"=nil

Summary

- Natural language processing
- Probabilistic methods
- N-gram models
- Information Retrieval