

# LING/C SC/PSYC 438/538

Lecture 21

Sandiway Fong

# Today's Topics

- Last Time
  - Stemming and minimum edit distance
- Reading
  - Chapter 4 of JM: N-grams
- Pre-requisite:
  - An understand of simple probability concepts
    - Sample Space
    - Events
    - Counting
    - Event Probability
    - Entropy/Perplexity (*later*)

In more depth,  
Statistical Natural Language Processing  
course is offered Spring 2011.  
Instructor: Erwin Chan.

# Introduction to Probability

- **some definitions**

- **sample space**

- the set of all possible outcomes of a statistical experiment is called the **sample space** ( $S$ )
    - *finite or infinite (also discrete or continuous)*

- **example**

- *coin toss experiment*
    - possible outcomes: {heads, tails}



- **example**

- *die toss experiment*
    - possible outcomes: {1,2,3,4,5,6}



# Introduction to Probability

- **some definitions**

- **sample space**

- the set of all possible outcomes of a statistical experiment is called the ***sample space*** ( $S$ )
    - *finite or infinite (also discrete or continuous)*

- **example**

- die toss experiment ***for whether the number is even or odd***
    - possible outcomes: {even,odd}
    - *not* {1,2,3,4,5,6}



# Introduction to Probability

- some definitions

- events

- an *event* is a subset of *sample space*
    - *simple* and *compound* events

- example

- *die toss experiment*
    - let A represent the event such that the outcome of the die toss experiment is divisible by 3
    - $A = \{3,6\}$
    - a subset of the sample space  $\{1,2,3,4,5,6\}$

# Introduction to Probability

- **some definitions**

- **events**

- an **event** is a subset of **sample space**
    - **simple** and **compound** events



- **example**

- alternative sample space  $S =$  set of 52 cards
    - A and B would both be compound events



- **example**

- **deck of cards draw experiment**
    - suppose sample space  $S =$  {heart,spade,club,diamond} (*four suits*)

- let A represent the event of drawing a heart
    - let B represent the event of drawing a red card

- $A = \{\text{heart}\}$  (*simple event*)
    - $B = \{\text{heart}\} \cup \{\text{diamond}\} =$  {heart,diamond} (*compound event*)
      - a compound event can be expressed as a set union of simple events

# Introduction to Probability

- **some definitions**

- **events**

- an **event** is a subset of **sample space**
    - **null space**  $\{\}$  (or  $\emptyset$ )
    - **intersection** of two events A and B is the event containing all elements common to A and B
    - **union** of two events A and B is the event containing all elements belonging to A or B or both

- **example**

- **die toss experiment**, sample space  $S = \{1,2,3,4,5,6\}$
    - let A represent the event such that the outcome of the experiment is divisible by 3
    - let B represent the event such that the outcome of the experiment is divisible by 2
    - **intersection** of events A and B is  $\{6\}$  (*simple event*)
    - **union** of events A and B is the compound event  $\{2,3,4,6\}$

# Introduction to Probability

- **some definitions**

- **rule of counting**

- suppose operation  $o_i$  can be performed in  $n_i$  ways,  
a sequence of  $k$  operations  $o_1 o_2 \dots o_k$  can be performed in  $n_1 \times n_2 \times \dots \times n_k$  ways

- **example**

- **die toss experiment**, 6 possible outcomes
    - two dice are thrown at the same time
    - number of sample points in sample space =  $6 \times 6 = 36$





# Introduction to Probability

- some definitions

- **permutations**

- a **permutation** is an arrangement of all or part of a set of objects
    - the **number of permutations** of  $n$  distinct objects is  $n!$
    - ( $n!$  is read as *n factorial*)

- **Definition:**

- $n! = n \times (n-1) \times \dots \times 2 \times 1$
    - $n! = n \times (n-1)!$
    - $1! = 1$
    - $0! = 1$

- **example**

- suppose there are 3 students: adam, bill and carol
    - how many ways are there of lining up the students?
    - **Answer:** 6
    - $3!$  permutations

1st
2nd
3rd

3 ways

2 ways

1 way

# Introduction to Probability

- some definitions

- permutations

- a **permutation** is an arrangement of all or part of a set of objects
    - the **number of permutations** of  $n$  distinct objects **taken  $r$  at a time** is  $n!/(n-r)!$

- example

- a first and a second prize raffle ticket is drawn from a book of 425 tickets
    - Total number of sample points =  $425!/(425-2)!$
    - $= 425!/423!$
    - $= 425 \times 424 = 180,200$  possibilities
    - *instance of sample space calculation*



# Introduction to Probability

- some definitions

- combinations

- the **number of combinations** of  $n$  distinct objects **taken  $r$  at a time** is  $n!/(r!(n-r)!)$
    - *combinations differ from permutations in that in the former case the selection is taken without regard for order*

- example

- given 5 linguists and 4 computer scientists
    - *what is the number of three-person committees that can be formed consisting of two linguists and one computer scientist?*
    - *note: order does not matter here*
    - select 2 from 5:
    - $5!/(2!3!) = (5 \times 4)/2 = 10$
    - select 1 from 4:
    - $4!(1!3!) = 4$
    - answer =  $10 \times 4 = 40$  (rule of counting – i.e. sequencing)

# Introduction to Probability

- some definitions

- probability

- **probability** are **weights** associated with sample points
    - a sample point with relatively **low weight** is unlikely to occur
    - a sample point with relatively **high weight** is likely to occur
    - **weights** are in the range zero to 1
    - **sum** of all the weights in the **sample space** must be 1
    - (see **smoothing**)
    - **probability of an event** is the sum of all the weights for the sample points of the event

- example

- **unbiased coin tossed twice**
    - sample space = {hh, ht, th, tt} (h = heads, t = tails)
    - coin is unbiased => each outcome in the sample space is equally likely
    - weight = 0.25  
( $0.25 \times 4 = 1$ )
    - **What is the probability that at least one head occurs?**
    - sample points/probability for the event: hh 0.25 th 0.25 ht 0.25
    - Answer: 0.75 (*sum of weights*)

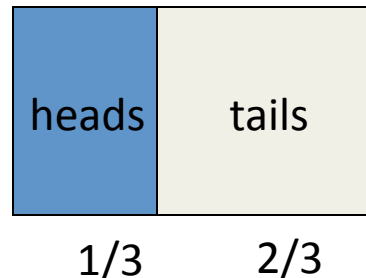
# Introduction to Probability

- some definitions

- probability

- **probability** are **weights** associated with sample points
    - a sample point with relatively **low weight** is unlikely to occur
    - a sample point with relatively **high weight** is likely to occur
    - **weights** are in the range zero to 1
    - **sum** of all the weights in the **sample space** must be 1
    - **probability of an event** is the sum of all the weights for the sample points of the event

> 50% chance  
or < 50%  
chance?



- example

- *a biased coin, twice as likely to come up tails as heads, is tossed twice*
    - *What is the probability that at least one head occurs?*

sample space = {hh, ht, th, tt} (h = heads, t = tails)

sample points/probability for the event:

- ht  $1/3 \times 2/3 = 2/9$
      - hh  $1/3 \times 1/3 = 1/9$
      - th  $2/3 \times 1/3 = 2/9$
      - tt  $2/3 \times 2/3 = 4/9$

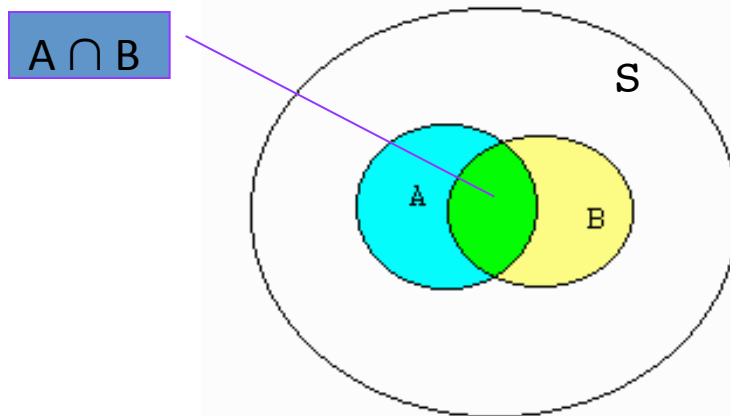
- Answer:  $\approx 0.56$  (sum of weights in **bold**)
    - cf. probability of same event for the unbiased coin = 0.75

# Introduction to Probability

- **some definitions**

- **probability**

- let  $p(A)$  and  $p(B)$  be the probability of events  $A$  and  $B$ , respectively.
    - **additive rule:**  $p(A \cup B) = p(A) + p(B) - p(A \cap B)$
    - if  $A$  and  $B$  are **mutually exclusive** events:  $p(A \cup B) = p(A) + p(B)$ 
      - since  $p(A \cap B) = p(\emptyset) = 0$



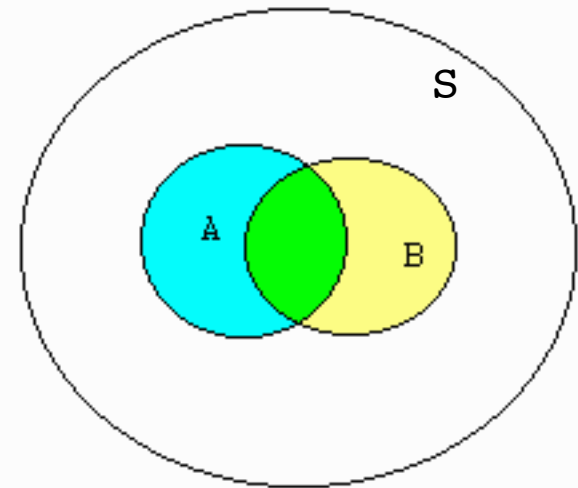
- **example**

- suppose probability of a student getting an A in linguistics is  $2/3$  ( $\approx 0.66$ )
    - suppose probability of a student getting an A in computer science is  $4/9$  ( $\approx 0.44$ )
    - suppose probability of a student getting at least one A is  $4/5$  ( $= 0.8$ )
    - **What is the probability a student will get an A in both?**

- $p(A \cup B) = p(A) + p(B) - p(A \cap B)$
    - $4/5 = 2/3 + 4/9 - p(A \cap B)$
    - $p(A \cap B) = 2/3 + 4/9 - 4/5 = 14/45 \approx \mathbf{0.31}$

# Introduction to Probability

- **some definitions**
  - **conditional probability**
    - let A and B be events
    - $p(B|A)$  = the **probability** of event B **occurring given** event A **occurs**
    - **definition:**  $p(B|A) = p(A \cap B) / p(A)$  provided  $p(A) > 0$
  - ***used an awful lot in language processing***
    - (*context-independent*)
    - probability of a word occurring in a corpus
    - (*context-dependent*)
    - probability of a word occurring given the previous word



# N-grams

- Google Web N-gram corpus  
*Frequency counts for sequences up to N=5*
  - Number of tokens:  
1,024,908,267,229
  - Number of sentences:  
95,119,665,584
  - Number of unigrams:  
13,588,391
  - Number of bigrams:  
314,843,401
  - Number of trigrams:  
977,069,902
  - Number of fourgrams:  
1,313,818,354
  - Number of fivegrams:  
1,176,470,663

## Web 1T 5-gram Version 1

*Item Name:* Web 1T 5-gram Version 1

*Authors:* Thorsten Brants, Alex Franz

*LDC Catalog No.:* LDC2006T13

*ISBN:* 1-58563-397-6

*Release Date:* Sep 19, 2006

*Data Type:* text

*Data Source(s):* web collection

*Application(s):* language modeling, machine learning, natural language processing

*Language(s):* English

*Language ID(s):* eng

*Distribution:* 6 DVD



# N-grams: Unigrams

- **introduction**

- Given a corpus of text, the  $n$ -grams are the sequences of  $n$  consecutive words that are in the corpus

- **example** (*12 word sentence*)

- the cat that sat on the sofa also sat on the mat

- **N=1** (8 *unigrams*)

- the     3
- sat     2
- on      2
- cat     1
- that    1
- sofa    1
- also    1
- mat     1

# N-grams: Bigrams

- **example** (12 word sentence)
  - the cat that sat on the sofa also sat on the mat
- **N=2** (9 *bigrams*)
  - sat on 2
  - on the 2
  - the cat 1
  - cat that 1
  - that sat1
  - the sofa 1
  - sofa also 1
  - also sat1
  - the mat 1

← 2 words →

# N-grams: Trigrams

- **example** (12 word sentence)
  - the cat that sat on the sofa also sat on the mat
- **N=3 (9 trigrams)**
  - *most language models stop here, some stop at quadrigrams*
    - too many n-grams to deal with? Sparse data problem
    - low frequencies

- sat on the 2
- the cat that 1
- cat that sat 1
- that sat on 1
- on the sofa 1
- the sofa also 1
- sofa also sat 1
- also sat on 1
- on the mat 1

← 3 words →

ceramics collectables collectibles	55
ceramics collectables fine	130
ceramics collected by	52
ceramics collectible pottery	50
ceramics collectibles cooking	45
ceramics collection ,	144
ceramics collection .	247
ceramics collection	120
ceramics collection and	43
ceramics collection at	52
ceramics collection is	68
ceramics collection of	76
ceramics collection	59
ceramics collections ,	66
ceramics collections .	60
ceramics combined with	46
ceramics come from	69
ceramics comes from	660
ceramics community ,	109
ceramics community .	212
ceramics community for	61
ceramics companies .	53
ceramics companies consultants	173

Google

# N-grams: Quadrigrams

- Example: (12 word sentence)
  - the cat that sat on the sofa also sat on the mat

- N=4 (9 quadrigrams)

- the cat that sat 1
- cat that sat on 1
- that sat on the 1
- sat on the sofa 1
- on the sofa also 1
- the sofa also sat 1
- sofa also sat on 1
- also sat on the 1
- sat on the mat 1

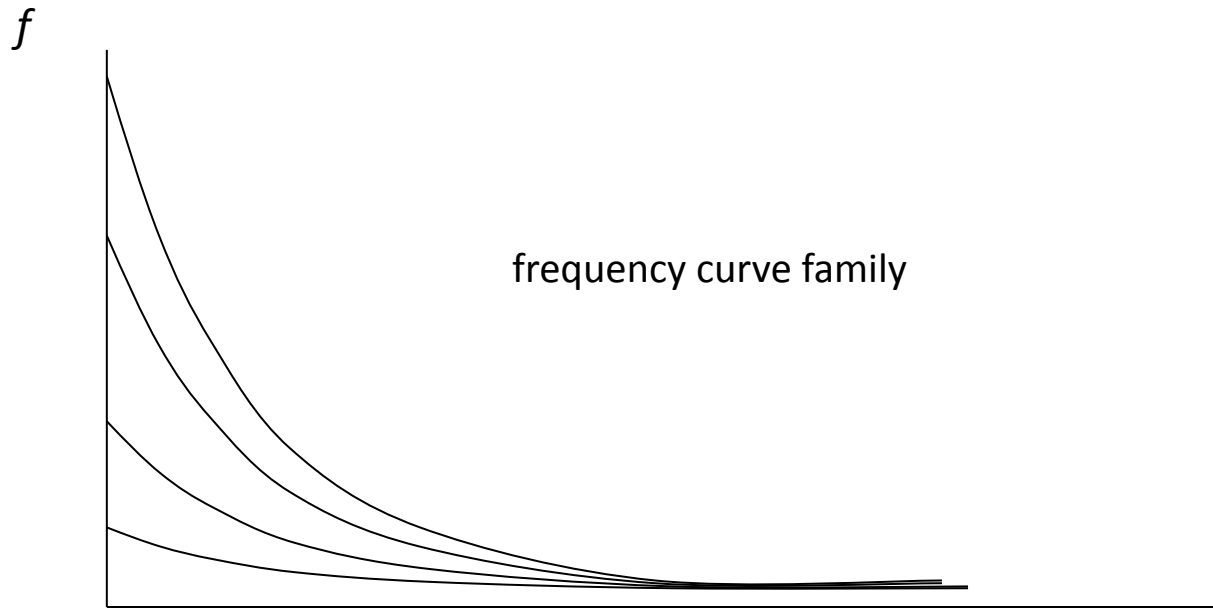
← 4 words →

serve as the incoming	92	
serve as the incubator	99	
serve as the independent		794
serve as the index	223	
serve as the indication	72	
serve as the indicator	120	
serve as the indicators	45	
serve as the indispensable		111
serve as the indispensable		40
serve as the individual	234	
serve as the industrial	52	
serve as the industry	607	
serve as the info	42	
serve as the informal	102	
serve as the information		838
serve as the informational		41
serve as the infrastructure		500
serve as the initial	5331	
serve as the initiating	125	
serve as the initiation	63	

Google

# N-grams: frequency curves

- **family of curves sorted by frequency**
  - unigrams, bigrams, trigrams, quad*ri*grams ...
  - *decreasing frequency*



# N-grams: the word as a unit

- **we count words**
- ***but what counts as a word?***
  - **punctuation**
    - useful surface cue
    - also <s> = beginning of a sentence, as a dummy word
    - part-of-speech taggers include punctuation as words (why?)
  - **capitalization**
    - They, they    *same token or not?*
  - **wordform vs. lemma**
    - cats, cats *same token or not?*
  - **disfluencies**
    - part of spoken language
    - *er, um, main- mainly*
    - speech recognition systems have to cope with them

# N-grams: Word

- *what counts as a word?*

- **punctuation**

- useful surface cue
    - also <s> = beginning of a sentence, as a dummy word
    - part-of-speech taggers include punctuation as words (why?)

From the Penn Treebank tagset

Tag	Description	Examples
\$	dollar	\$ -\$ --\$ A\$ C\$ HK\$ M\$ NZ\$ S\$ U.S.\$ US\$
``	opening quotation mark	` ``
"	closing quotation mark	' "
(	opening parenthesis	( [ {
)	closing parenthesis	) ] }
,	comma	,
--	dash	--
.	sentence terminator	. ! ?
:	colon or ellipsis	: ; ...

# Google 1T Word Corpus

- 1.1 Source Data
  - The n-gram counts were generated from approximately 1 trillion word tokens of text from Web pages. We used only publically accessible Web pages. We attempted to use only Web pages with English text, but some text from other languages also found its way into the data.
- Data collection took place in January 2006.
- 2.2 Tokenization
  - Hyphenated word are usually separated, and hyphenated numbers usually form one token.-
  - Sequences of numbers separated by slashes (e.g. in dates) form one token.
  - Sequences that look like urls or email addresses form one token.
- 2.5 Sentence Boundaries
  - Sentence boundaries were automatically detected.
  - The beginning of a sentence was marked with "<S>", the end of a sentence was marked with "</S>".
  - The inserted tokens "<S>" and "</S>" were counted like other words and appear in the n-gram tables.
  - So, for example, the unigram count for "<S>" is equal to the number of sentences into which the training corpus was divided.
- 2.3 Filtering
  - Malformed UTF-8 encoding.
  - Tokens that are too long.
  - Tokens containing any non-Latin scripts (e.g. Chinese ideographs).
  - Tokens containing ASCII control characters.
  - Tokens containing non-ASCII digit, punctuation, or space characters.
  - Tokens containing too many non-ASCII letters (e.g. accented letters).
  - Tokens made up of a combination of letters, punctuation, and/or digits that does not seem useful.
- 2.4 The Token "<UNK>"
  - All filtered tokens, as well as tokens that fell beneath the wordfrequency cutoff (see 3.1 below), were mapped to the special token "<UNK>" (for "unknown word").
- 3. Frequency Cutoffs
- 3.1 Word Frequency Cutoff
  - **All tokens (words, numbers, and punctuation) appearing 200 times or more** (1 in 5 billion) were kept and appear in the n-gram tables
  - Tokens with lower counts were mapped to the special token "<UNK>".
- 3.2 N-gram Frequency Cutoff
  - **N-grams appearing 40 times or more** (1 in 25 billion) were kept, and appear in the n-gram tables.
  - All n-grams with lower counts were discarded.

<http://www ldc.upenn.edu/Catalog/docs/LDC2006T13/readme.txt>



# Google 1T Word Corpus

- Size poses problems
  - 24GB compressed
  - 80GB uncompressed
- How to search it?
  - e.g. perform wildcard searching, when even indexes for the N-gram corpus won't fit in memory

Proceedings of the Australasian Language Technology Workshop 2007, pages 40-48

## Practical Queries of a Massive n-gram Database

Tobias Hawker  
School of  
Information Technologies  
University of Sydney  
NSW 2006, Australia  
toby@it.usyd.edu.au

Mary Gardiner  
Centre for Language Technology  
Macquarie University  
gardiner@ics.mq.edu.au

Andrew Bennetts  
Canonical Ltd.  
andrew@puzzling.org

### Abstract

Large quantities of data are an increasingly essential resource for many Natural Language Processing techniques. The Web 1T corpus, a massive resource containing n-gram frequencies produced from one trillion words drawn from the World Wide Web, is a relatively new corpus whose size will increase performance on many data-hungry applications. In addition, a fixed resource of this kind reduces reliance on using web results as experimental data, increasing replicability of researchers' results.

However, effectively utilising a resource of this size presents significant challenges. We discuss the challenges of using a data source of this magnitude, and describe strategies for overcoming these, including efficient extraction of queries including wildcards, and specialised data compression. We present a software suite, "Get 1T", implementing these techniques, released as free software for use by the natural language research community, and others.

### 1 Introduction

The size and quality of data used for statistical Natural Language Processing can have a major impact on the results a system achieves (Banko and Brill, 2001). The World Wide Web gives researchers access to an unprecedented quantity of machine-readable natural language text. However, even for techniques that can take advantage of unannotated

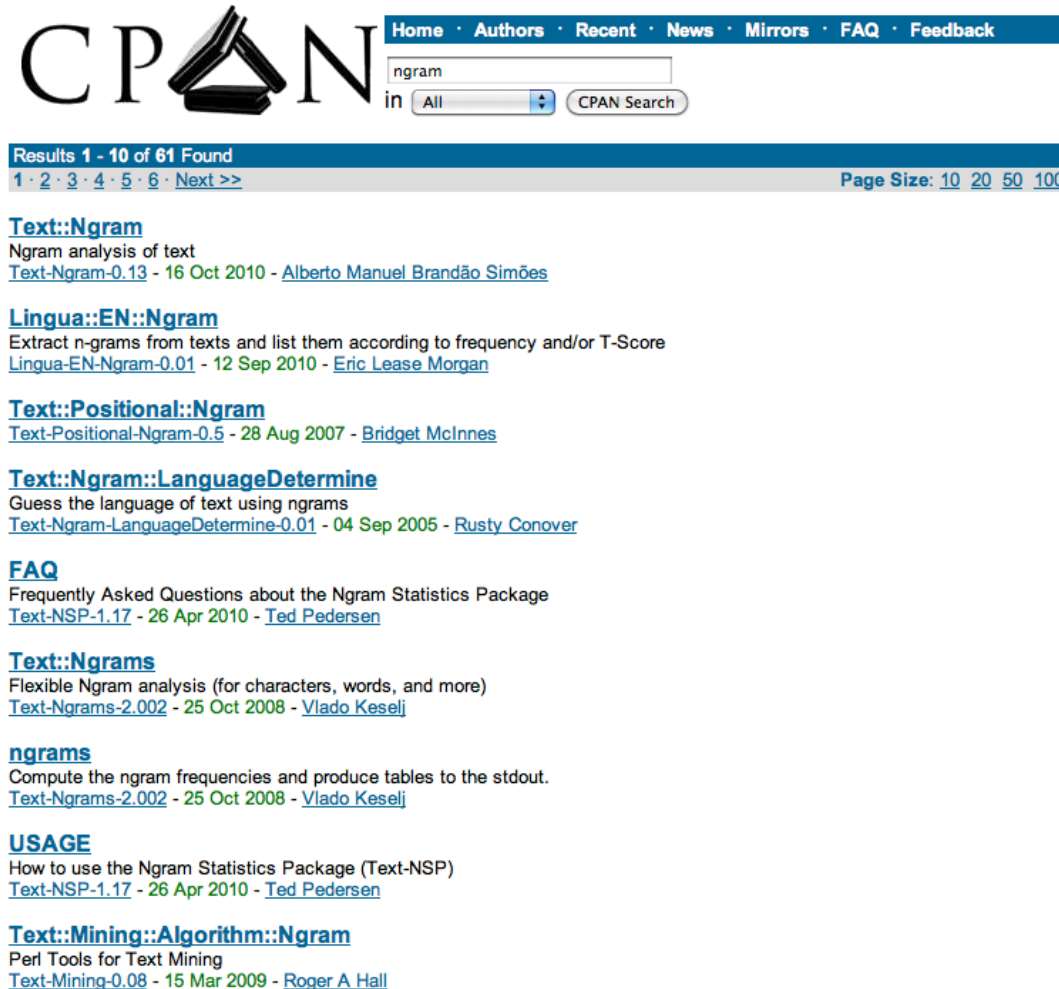
text, there is much more web data available than can normally be used, because this same size overwhelms the processing resources available to most researchers. One way of overcoming this practical problem is to use the processing resources of commercial search engines, by using the number of hits they report for frequency estimation — see for example Grefenstette (1999), Turney (2001), Keller and Lapata (2003) and Nakov and Hearst (2005). There are however several drawbacks to this approach, recently highlighted by Kilgarriff (2007), particularly replicability of experiments.

The recent release of the Web 1T corpus (Brants and Franz, 2006)<sup>1</sup> presents an opportunity to access web-scale data for n-gram frequency estimation without the drawbacks of using a search engine. The corpus provides frequency counts for n-grams up to five tokens long, drawn from approximately 1 trillion words of web data. Promising results have already been achieved using this resource for Word Sense Disambiguation and Lexical Substitution (Hawker, 2007; Yuret, 2007) and for Noun-Phrase Bracketing (Vadas and Curran, 2007).

However, the scale of even this distilled collection of web data presents significant processing challenges. Naïve methods for extracting the desired information from the corpus, such as linear search or keyword indexing, are hopelessly impractical, and even algorithms with good asymptotic performance such as binary search are limited in their applicability. Approaches that attempt to overcome the scaling problems by using indices for the set of discrete to-

<sup>1</sup><http://www ldc.upenn.edu/Catalog/docs/LDC2006T13/readme.txt>

# N-gram Software



The screenshot shows the CPAN (Comprehensive Perl Archive Network) search results for the term 'ngram'. The CPAN logo is at the top left. A navigation bar includes links for Home, Authors, Recent, News, Mirrors, FAQ, and Feedback. A search bar contains the text 'ngram' and a dropdown menu is set to 'All'. Below the search bar, a blue banner indicates 'Results 1 - 10 of 61 Found'. A pagination bar shows links for 1, 2, 3, 4, 5, 6, and a 'Next >>' link. On the right side of the banner, 'Page Size' options are listed: 10, 20, 50, and 100. The search results list several packages:

- Text::Ngram**  
Ngram analysis of text  
[Text-Ngram-0.13](#) - 16 Oct 2010 - [Alberto Manuel Brandão Simões](#)
- Lingua::EN::Ngram**  
Extract n-grams from texts and list them according to frequency and/or T-Score  
[Lingua-EN-Ngram-0.01](#) - 12 Sep 2010 - [Eric Lease Morgan](#)
- Text::Positional::Ngram**  
[Text-Positional-Ngram-0.5](#) - 28 Aug 2007 - [Bridget McInnes](#)
- Text::Ngram::LanguageDetermine**  
Guess the language of text using ngrams  
[Text-Ngram-LanguageDetermine-0.01](#) - 04 Sep 2005 - [Rusty Conover](#)
- FAQ**  
Frequently Asked Questions about the Ngram Statistics Package  
[Text-NSP-1.17](#) - 26 Apr 2010 - [Ted Pedersen](#)
- Text::Ngrams**  
Flexible Ngram analysis (for characters, words, and more)  
[Text-Ngrams-2.002](#) - 25 Oct 2008 - [Vlado Keselj](#)
- ngrams**  
Compute the ngram frequencies and produce tables to the stdout.  
[Text-Ngrams-2.002](#) - 25 Oct 2008 - [Vlado Keselj](#)
- USAGE**  
How to use the Ngram Statistics Package (Text-NSP)  
[Text-NSP-1.17](#) - 26 Apr 2010 - [Ted Pedersen](#)
- Text::Mining::Algorithm::Ngram**  
Perl Tools for Text Mining  
[Text-Mining-0.08](#) - 15 Mar 2009 - [Roger A Hall](#)

- Lots of packages available
  - *no need to reinvent the wheel*
  - [www.cpan.org](http://www.cpan.org)

# Text-NSP

- Standalone (command-line) Perl package from Ted Petersen  
(*Install and try it with a text file of your choice!*)
  - Ngram Statistics Package (Text-NSP)
  - Perl code
  - Allows you get n-grams from text and compute a variety of statistics
  - **Homepage** <http://www.d.umn.edu/~tpederse/nsp.html>

## DESCRIPTION

### 1. Introduction

The Ngram Statistics Package (NSP) is a suite of programs that aids in analyzing Ngrams in text files. We define an Ngram as a sequence of 'n' tokens that occur within a window of at least 'n' tokens in the text; what constitutes a "token" can be defined by the user.

In earlier versions (v0.1, v0.3, v0.4) this package was known as the Bigram Statistics Package (BSP). The name change reflects the widening scope of the package in moving beyond Bigrams to Ngrams.

NSP consists of two core programs and three utilities:

Program [count.pl](#) takes flat text files as input and generates a list of all the Ngrams that occur in those files. The Ngrams, along with their frequencies, are output in descending order of their frequency.

Program [statistic.pl](#) takes as input a list of Ngrams with their frequencies (in the format output by count.pl) and runs a user-selected statistical measure of association to compute a "score" for each Ngram. The Ngrams, along with their scores, are output in descending order of this score. The statistical score computed for each Ngram can be used to decide whether or not there is enough evidence to reject the null hypothesis (that the Ngram is not a collocation) for that Ngram.

# Language Models and N-grams

- Remember what Jerry Ball said in his guest lecture about on-line processing...
  - *humans beings don't wait until the of the sentence to begin to assign structure and meaning*
  - we're expectation-driven...
- All about prediction...

# Language Models and N-grams

- **Brown corpus** (1million words):
  - word  $w$      $f(w)$      $p(w)$
  - *the*            69,971    0.070
  - *rabbit* 11            0.000011
- **given a word sequence**
  - $w_1 w_2 w_3 \dots w_n$
  - probability of seeing  $w_i$  depends on what we seen before
    - *recall conditional probability defined earlier*
- **example (from older version of textbook section 6.2)**
  - Just then, the white **rabbit**
  - **the**
  - expectation is  $p(\text{rabbit} | \text{white}) > p(\text{the} | \text{white})$
  - but  $p(\text{the}) > p(\text{rabbit})$

# Language Models and N-grams

- **given a word sequence**
  - $w_1 w_2 w_3 \dots w_n$
- **chain rule**
  - *how to compute the probability of a sequence of words*
  - $p(w_1 w_2) = p(w_1) p(w_2 | w_1)$
  - $p(w_1 w_2 w_3) = p(w_1) p(w_2 | w_1) p(w_3 | w_1 w_2)$
  - ...
  - $p(w_1 w_2 w_3 \dots w_n) = p(w_1) p(w_2 | w_1) p(w_3 | w_1 w_2) \dots p(w_n | w_1 \dots w_{n-2} w_{n-1})$
- **note**
  - It's not easy to collect (meaningful) statistics on  $p(w_n | w_{n-1} w_{n-2} \dots w_1)$  for all possible word sequences

# Language Models and N-grams

- **Given a word sequence**

- $w_1 w_2 w_3 \dots w_n$

- **Bigram approximation**

- *just look at the previous word only (not all the proceedings words)*

- **Markov Assumption: finite length history**

- 1st order Markov Model

- $p(w_1 w_2 w_3 \dots w_n) = p(w_1) p(w_2 | w_1) p(w_3 | w_1 w_2) \dots p(w_n | w_1 \dots w_{n-3} w_{n-2} w_{n-1})$

- $p(w_1 w_2 w_3 \dots w_n) \approx p(w_1) p(w_2 | w_1) p(w_3 | w_2) \dots p(w_n | w_{n-1})$

- **note**

- $p(w_n | w_{n-1})$  is a lot easier to collect data for (and thus estimate well) than  $p(w_n | w_1 \dots w_{n-2} w_{n-1})$

# Language Models and N-grams

- **Trigram approximation**

- 2nd order Markov Model
- *just look at the preceding two words only*
- $p(w_1 w_2 w_3 w_4 \dots w_n) = p(w_1) p(w_2 | w_1) p(w_3 | w_1 w_2) p(w_4 | w_1 w_2 w_3) \dots p(w_n | w_1 \dots w_{n-3} w_{n-2} w_{n-1})$
- $p(w_1 w_2 w_3 \dots w_n) \approx p(w_1) p(w_2 | w_1) p(w_3 | w_1 w_2) p(w_4 | w_2 w_3) \dots p(w_n | w_{n-2} w_{n-1})$

- **note**

- $p(w_n | w_{n-2} w_{n-1})$  is a lot easier to estimate well than  $p(w_n | w_1 \dots w_{n-2} w_{n-1})$  but harder than  $p(w_n | w_{n-1})$