Informatics 1B: Data and Analysis

Lecture 9: Semi-structured Data: Statistics and Information Retrieval

Frank Keller

School of Informatics
University of Edinburgh
keller@inf.ed.ac.uk

February 7, 2005

Reading: lecture notes; Manning and Schütze (1999: ch. 15).

Statistical Tests

Last lecture:

- extract collocations from corpora: linguistically meaningful bigrams (e.g., *strong tea*);
- need to filter out noise (e.g., *strong and*);
- use statistical test to distinguish collocations from chance co-occurrences.

This lecture:

- look at one statistical test in detail: χ^2 test;
- discuss an application that relies on semi-structured data: information retrieval.

χ^2 Test

The χ^2 (chi-squared) test:

- compares n *frequency distributions,* each with m values;
- tests the *null hypothesis* that the distributions are the same;
- takes as its input an $n \times m$ *contingency table.*

Example

Compare performance of boys and girls in an exam with marks A, B, C, and D. Data: 4×2 contingency table, with marks on x-axis and distribution on y-axis.

χ^2 Test

Example: exam data

O_{ij}	A	B	C	D	$\sum_i O_{ij}$
Boys	3	23	43	10	79
Girls	6	34	31	4	75
$\sum_j O_{ij}$	9	57	74	14	154

Compute χ^2 statistic by comparing:

- *observed frequencies:* frequencies that have been observed experimentally, and
- *expected frequencies:* frequencies that would be expected if the null hypothesis was true (no difference between the distributions).

χ^2 Test

Equation for χ^2:

(1) $$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

i: ranges over rows of the contingency table; j: ranges over its columns; O_{ij}: observed frequency for cell (i, j); E_{ij}: the expected frequency for cell (i, j)

Equation for *expected frequencies:*

(2) $$E_{ij} = \frac{\sum_j O_{ij} \sum_i O_{ij}}{N}$$

N: overall number of observations; $\sum_j O_{ij}$ and $\sum_i O_{ij}$: *marginals* of contingency table.

Example

Example: exam data

Calculate the expected frequencies for the exam data:

E_{ij}	A	B	C	D
Boys	4.62	29.24	37.96	7.18
Girls	4.38	27.76	36.04	6.82

Now compute χ^2 and compare it against the *critical value*: if it exceeds it, the null hypothesis can be rejected, test is *significant.*

Example: exam data

Plug the expected frequencies into (1): $\chi^2 = 7.55$. This doesn't exceed critical value of 7.82 (get this from a stats book): exam performance of boys and girls not significantly different.

Collocation Filtering

The χ^2 test can be applied to *collocation filtering:* check if a bigrams is a valid collocations. For a bigram $w_1 w_2$, compile a contingency table as follows:

O_{ij}	w_1	$\neg w_1$
w_2	$f(w_1, w_2)$	$f(\neg w_1, w_2)$
$\neg w_2$	$f(w_1, \neg w_2)$	$f(\neg w_1, \neg w_2)$

$f(w_1, w_2)$: word w_1 and word w_2 occur together;
$f(\neg w_1, w_2)$: w_2 occurs preceded by a word other than w_1;
$f(w_1, \neg w_2)$: w_1 occurs followed by a word other than w_2;
$f(\neg w_1, \neg w_2)$: two words other than w_1 and w_2 occur together.

Now apply χ^2 to this table: test the hypothesis that w_1 and w_2 occur together more often than expected by chance.

Example

strong	,	52	powerful	,	5
	and	31		effect	3
	enough	16		sight	3
	.	16		enough	3
	in	15		mind	3
	man	14		for	3
	emphasis	11		and	3
	desire	**10**		with	3
	upon	**10**		enchanter	2
	interest	8		displeasure	2
	a	8		motives	2
	as	8		impulse	2
	inclination	7		struggle	2
	tide	7		grasp	2
	beer	7		friends	2

Collocation Filtering

Example

Use χ^2 to filter the collocations of *strong*, e.g., *strong desire* vs. *strong upon* (both occur 10 times in our corpus).

O_{ij}	*strong*	$\neg$*strong*
desire	10	214
$\neg$*desire*	655	3407085
	$\chi^2(1) = 46684423$	

O_{ij}	*strong*	$\neg$*upon*
upon	10	7107
$\neg$*upon*	655	3407085
	$\chi^2(1) = 6235$	

The χ^2 values are significant for both bigrams, but much higher for *strong desire*. We can therefore filter bigrams by applying a cutoff to their χ^2 values.

From Words to Documents

The course so far:

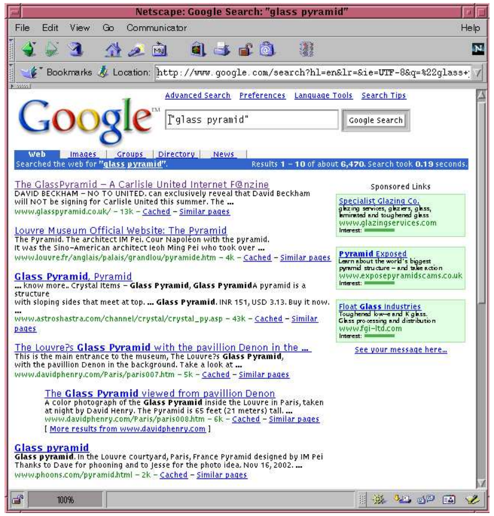
- extract information from semi-structured data;
- use query engines that support regular expression over words;
- use statistical tests to filter the information.

The rest of the course:

- retrieve whole documents rather than words (or *n*-grams) from semi-structured data;
- specifically: information retrieval (IR), increasingly relevant application (large document collections, web);

Probably the most well-know IR applications are search engines.

Information Retrieval Application



Information Retrieval Application

Define the IR task: *ad hoc retrieval problem:* given a query, find the documents that are relevant to it. Assumptions:

- search large, static document collection;
- user has an information need, formulated in terms of a query (typically keywords);
- task: find all and only the documents relevant to the query.

Example: search engine

Document collection to be searched: web pages. Information need: find pages on a particular topic. Query: user specifies keywords. Search engine returns a ranked list of relevant web pages.

Other examples: bibliographic information system; electronic newspaper archives.

Scientific Questions in IR

We will now introduce some core IR techniques. Scientific problems to be addressed:

- *Query type:* How to formulate queries to an IR system?
- *Indexing:* Best way of representing the documents searched by the system?
- *Retrieval model:* How to find the best-matching document? How to do it efficiently?
- *Output presentation:* Best way of presented the results of the search?
- *Evaluation:* How to measure the performance of the system? How to test that the system does what it is supposed to?

Indexing

Indexing: represent the document collection to the searched in a way that facilitates retrieval:

- find *terms:* words or phrases that describe the documents and can be matched against a query for retrieval;
- *manual indexing:* human annotators choose terms; typically employs large vocabularies (thousand of terms);
- *advantages:* works well for closed document collections (e.g., books in a library); achieves high precision;
- *disadvantages:* annotators need to be trained to achieve consistency; doesn't work well for dynamic document collections (e.g., web).

Manual Indexing

Example: manual index

ACM Computing Classification System (1998)	
B	Hardware
B.3	Memory structures
B.3.0	General
B.3.1	Semiconductor Memories (NEW) (was B.7.1)
	Dynamic memory (DRAM) (NEW)
	Read-only memory (ROM) (NEW)
	Static memory (SRAM) (NEW)
B.3.2	Design Styles (was D.4.2)
	Associative memories
	Cache memories
	Interleaved memories
	Mass storage (e.g., magnetic, optical, RAID)
	Primary memory
	Sequential-access memory

Automatic Indexing

IR systems can perform *automatic indexing:* automatically extract relevant terms from documents:

- no predefined set of index terms;
- instead use the words in the documents as terms; vocabulary changes dynamically with document collection.

Inverted index:

- data structure that contains all words in the document collection;
- given an inverted index and a query, we can retrieve all documents containing the words in the query;
- faster than searching through whole document collection.

Creation of an Inverted Index

Document	Text
1	Pease porridge hot, pease porridge cold
2	Pease porridge in the pot
3	Nine days old
4	Some like it hot, some like it cold
5	Some like it in the pot
6	Nine days old

$\Longrightarrow$

Number	Text	Documents
1	cold	1, 4
2	days	3, 6
3	hot	1, 4
4	in	2, 5
5	it	4, 5
6	like	4, 5
7	nine	3, 6
8	old	3, 6
9	pease	1, 2
10	porridge	1, 2
11	pot	2, 5
12	some	4, 5
13	the	2, 5

More on Indexing

Important concepts for indexing:

- *positional information:* the index tells you where in the document a give word occurs;
- *term weighting:* assume some terms more important than others (e.g., based on frequency);
- *term manipulation:* map certain words or phrases on the same term (e.g., capitalization, plural);
- *stop words removal:* remove words that have little information value and occur in most documents (e.g., *the*, *of*, *our*).

Creation of an Inverted Index with Position Information

Document	Text
1	Pease porridge hot, pease porridge cold
2	Pease porridge in the pot
3	Nine days old
4	Some like it hot, some like it cold
5	Some like it in the pot
6	Nine days old

$\Longrightarrow$

Number	Text	(Document; Word)
1	cold	(1; 6), (4; 8)
2	days	(3; 2), (6; 2)
3	hot	(1; 3), (4; 4)
4	in	(2; 3), (5; 4)
5	it	(4; 3, 7), (5; 3)
6	like	(4; 2, 6), (5; 2)
7	nine	(3; 1), (6; 1)
8	old	(3; 3), (6; 3)
9	pease	(1; 1, 4), (2; 1)
10	porridge	(1; 2, 5), (2; 2)
11	pot	(2; 5), (5; 6)
12	some	(4; 1, 5), (5; 1)
13	the	(2; 4), (5; 5)

Summary

- χ^2 test compares two frequency distributions; null hypothesis: distributions are the same;
- uses contingency tables as data representation;
- χ^2 can be used to filter collocation: test if a two words occur together more often than chance;
- information retrieval: retrieve documents from a collection based on a user query (e.g., search engine);
- indexing: represent the documents in the collection as a set of index terms;
- can be performed automatically using inverted index; allows dynamic, efficient retrieval.

References

Manning, Christopher D. and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.