Corpus Linguistics  
(L615)  
Web as Corpus

Markus Dickinson  
Department of Linguistics, Indiana University  
Spring 2013
The web provides new opportunities for gathering data

- Viable source of “disposable corpora”, built ad hoc for specific purposes
- Essential for working with specialized languages

Need to automatically extract web corpus data: manual extraction is time-consuming
Semi-automatic webpage extraction

You can use Perl’s libwww (LWP) module to access webpages
(http://search.cpan.org/~gaas/libwww-perl-6.05/lib/LWP.pm)

You can find other modules, like Web::Query and Webscraper
Fetch a webpage

web.pl

# See: http://www.perl.com/pub/a/2002/08/20/perlandlwp.html

# import the LWP::Simple module
use LWP::Simple;

# define the webpage to access:
$url = "http://cl.indiana.edu/~md7/";

# get the webpage content
$content = get $url;
die "Couldn’t get $url" unless defined $content;

print $content;
You can:

- Write regular expressions to extract data of interest (assuming you know what the HTML looks like)
- Learn more about HTML::Parse and the like

Or: use pre-built software such as BootCaT (discussed later)
Sketch Engine (http://www.sketchengine.co.uk/):
general-purpose web corpus extraction
Downside: you can use it for free for 30 days ... but then you have to pay to use it

Upside: some of the tools they use are available for free:
  ▶ http://www.drni.de/wac-tk/HomePage
  ▶ http://sslmitdev-online.sslmit.unibo.it/wac/wac.php
  ▶ http://wacky.sslmit.unibo.it/doku.php

And some seem to have been modified & incorporated into BootCaT (e.g., boilerplate stripping)
Benefits of a web corpus

This corpus can:

▶ Help address data sparseness issues
▶ Provide more interpersonal material
▶ Check the claims made with other corpora
Crawl seeding & crawling

Crawl the web with a few different seeds, combined into two-word queries:

- single word queries seemed to give too many inappropriate pages
- queries with 3 or more words seemed to give lists of words

Goal was to get both “public sphere” pages (e.g., newspapers) as well as more personal pages (e.g., blogs)

- sampled mid-frequency words from traditional written domain
- used basic vocabulary

Limited search to .de and .at domains; kept only one URL from a given domain
Initial filtering

Filtered documents by size

- Small documents (<5KB) contain very little real text
- Large documents (>200KB) tend to be indices, catalogues, lists, etc.

Removed perfect duplicates

- Actually, they removed both the original & the duplicate: tended to be warning messages & the like
Boilerplate stripping

*boilerplate* = HTML markup, javascript, & other non-linguistic material

- Removing boilerplate information is crucial to obtaining linguistic data only

Heuristic:

- Content-rich sections of a document will have a low html tag density
- Boilerplate sections have a wealth of html

This heuristic is “relatively independent of language and crawling strategy”
Function word & pornography filtering

If a text does not have enough function words, it is likely non-linguistic material (e.g., a list)
  ▶ Require at least 10 function word types & 30 tokens on a page ... which must make up at least 25% of the total words
  ▶ With a function word list from a language, this can also serve as a language identifier

Also remove pages which match a stop list of words likely to appear in pornography
  ▶ Tend to contain randomly-generated keyword lists
  ▶ Needs some work, as it includes otherwise harmless words like girl
Near-duplicate detection

The Sketch Engine removes near & exact duplicates

- Remove function words
- Take “fingerprints” of a fixed number of randomly-selected n-grams
  - e.g., extract 25 5-grams from each document
- Near-duplicates have a high overlap
  - e.g., at least 2 5-grams in common
Post-processing

With this data, the Sketch Engine then prepares it better for searching:

- Run a POS tagger over it (TreeTagger)
- Clean the documents further, using POS tags
  - By noting where the POS tag distribution is unusual, they perform another round of anomalous document finding
  - They look for problematic (erroneous) POS tags and remove those documents
    - Use cues such as number of unrecognized words, proportion of words with upper-case initial letters, ...
General purpose web corpora
Sharoff (2006)

Additional motivation for general-purpose web corpora

- Expensive to build corpora, yet they are needed for under-resourced languages
- Corpora are often restricted in size and/or variety
- News corpora do not represent general language
  - Need a variety of text types

Can search through corpora described here at:
http://corpus.leeds.ac.uk/internet.html
Limitations of web search interfaces

To answer some questions, we could just search the web, but ...

- Search engines only provide limited context
- Search engines do not allow for linguistically complex queries
- Results are organized according to relevance to the topic, not to left/right context
- Search engine counts cannot generally be trusted
Corpus of web URLs

One strategy for releasing a corpus is to organize a list of appropriate URLs

- Need to check that every page has real, connected text
- Need to develop a BNC-style (representative) corpus from the web
BNC-style corpus

Word selection

To get a representative corpus, we need a sufficiently general word list

Issues:

▶ Function words shouldn’t be included: they often occur with incomplete sentences

▶ Polysemous words should be good, in terms of not biasing the corpus towards one particular topic (e.g., word, room)

▶ Lemmatization would be good to use, to handle languages with elaborate morphology
  ▶ high in English and vysokyi in Russian have similar counts/ranks with lemmas, but not full word forms
BNC-style corpus
Word selection (2)

Approach: select 500 frequent word forms from a language

▶ words which start with lower-case letters (to avoid proper nouns)
▶ not specific with respect to a topic

Using more words starts to get into specific topics & increases efforts on developing query list
BNC-style corpus

Query generation

The queries need to get representative content, with a minimum of noisy pages.

Approach: use 4-word queries (i.e., 4-word random combinations of 500 words), totaling 5000 queries:

- Fewer words could lead to unconnected text (e.g., work & room).
- More words returns smaller number of pages: not a random snapshot of the web.
- Four words leads to:
  - Connected prose
  - A variety of domains

To ensure that a particular language is obtained, can add a language-specific function word to the query.
BNC-style corpus

Dowloading

5,000 queries led to 50,000 URLs, with about 3,000-4,000 words per query

▶ Generally enough to get 100 million word corpus
  ▶ Top 25,000 words have at least 100 occurrences each
  ▶ Seems to be sufficient for lexicography

Corpus size potentially limited by tools that can process them
BNC-style corpus

Post-processing

1. Unify page encodings (e.g., all in UTF-8)
2. Convert HTML into plain text (e.g., using `lynx`)
3. Filter out identical/near-identical pages
   ▶ Near-duplicate detection done by looking at shared $n$-grams (shingling algorithm)
## The retrieved corpora

<table>
<thead>
<tr>
<th></th>
<th>I-EN</th>
<th>I-DE</th>
<th>I-RU</th>
</tr>
</thead>
<tbody>
<tr>
<td># of tokens</td>
<td>126,643,151</td>
<td>126,117,984</td>
<td>156,534,391</td>
</tr>
<tr>
<td># of word forms</td>
<td>2,003,056</td>
<td>3,384,491</td>
<td>2,036,503</td>
</tr>
<tr>
<td># of lemmas</td>
<td>1,608,425</td>
<td>3,081,197</td>
<td>791,311</td>
</tr>
<tr>
<td># of URLs</td>
<td>42,133</td>
<td>31,195</td>
<td>33,811</td>
</tr>
<tr>
<td>Avg. doc. length (in words)</td>
<td>3,006</td>
<td>4,043</td>
<td>4,630</td>
</tr>
</tbody>
</table>
Text assessment

Authorship

To determine whether the corpora is balanced like the BNC, Sharoff assesses a variety of factors

Authorship:
- Single
- Multiple
- Corporate: 44% for I-EN, 18% for BNC
- Unknown

Also, female writers are underrepresented: 23%/3% male/female split in I-EN vs. 28%/13% for BNC
Text assessment

Mode

- Written
- Spoken: 0-1% for web corpora, 10% for BNC
- Electronic: 16% for Russian, 13% for English, 9% for German; 0% for BNC
Text assessment

Audience

Test the level of knowledge expected from the audience (size & level are harder to gauge)

- General: 33% in I-EN
- Informed: 45% in I-EN
- Professional: 22% in I-EN

Overall, I-EN seems somewhat balanced w.r.t. this classification (similar to BNC)
Corpus half-life

If a corpus consists of a list of URLs and associated software for extracting them, how stable is such a corpus?

- We can measure a corpus’s *half-life* by seeing how many pages are left after a certain amount of time
- Initial experiments show that some links are gone after a few months
  - February 2005 → August 2005: 934/1000 remaining
  - June 2005 → August 2005: 982/1000 remaining
- Need longer term studies and studies testing different parameters
BootCaT (Bootstrapping Corpora and Terms) works as follows:

1. Automatically search the web using a small set of seed terms
2. Extract new terms from this initial corpus
3. Use these terms to build a new corpus
4. Extract new terms

From this, you can extract a list of multi-word terms

See: http://bootcat.sslmit.unibo.it
Extraction of corpora & unigram terms

Bootstrapping terms starts with a small list of seeds

▶ Typically, only need 5-15 seed terms for a specialized domain
▶ Seed terms are randomly combined & combinations are used as queries
  ▶ Top \( n \) pages are retrieved

From this corpus, new unigram seed terms are extracted

▶ Compare frequencies of terms to frequencies in a reference corpus
▶ Then, the random combination is done again

Process is then repeated
Parameters

User defines the following parameters:

- Number of queries issued for each iteration
- Number of seeds used in a single query
- Number of pages to be retrieved
Extraction of multi-word terms

BootCaT also aims to extract multi-word terms from a particular domain of interest

- Extract one & two-word connectors from the corpus: terms frequently occurring between single-word terms
- Extract a list of stop words (high document frequency)
- Look for multi-word terms
  - contain at least one unigram term
  - do not contain stop words
  - connectors do not appear at the edges
  - have a high enough frequency
  - are not part of longer frequent multi-word terms
  - do not contain shorter frequent multi-word terms
Acquiring BootCaT

Download the toolkit from: http://bootcat.sslmit.unibo.it

▶ The frontend is actually placed over a suite of Perl programs
  ▶ “Each program should do only one thing, but do it well”
    (unix adage)

You will also need to get a Bing API key to be able to run 5000 queries/month

▶ http://docs.sslmit.unibo.it/doku.php?id=bootcat:help:search_engine_key
The BootCaT frontend

This wizard will walk you through the creation of a web corpus using the BootCaT Toolkit.

The steps involved in this simplified corpus creation process are:

- providing a list of seeds, i.e. words (or combination of words) that identify the domain you're investigating, or generic lexical items if you want to create a general-language corpus;
- combining these seeds in random tuples (i.e. sequences of n seeds);
- querying a search engine to find web pages that contain the generated tuples
- downloading the relevant web pages, cleaning up the HTML code and finally creating a corpus.
BootCaT

Using BootCaT

---

Corpus Linguistics
Web as Corpus
Using BootCaT

Insert one seed per line, multi-word seeds go on the same line

- jack tripper
- john ritter
- three's company
- janet
- chrissy
- 1970s
- sitcom
- mr. roper
- farce

I'm done editing seeds
Type seeds and check the relevant box when you're done
BootCaT

Using BootCaT

The tuples that will be used as queries

Tupel length

Generate tuples

Click on 'Generate tuples' to proceed

N. of tuples (max: 84)
BootCaT

Using BootCaT

The tuples that will be used as queries

- "mr. roper" sitcom 1970s
- farce "three's company" "mr. roper"
- "jack tripper" farce "john ritter"
- janet 1970s "jack tripper"
- "mr. roper" farce 1970s
- farce "mr. roper" sitcom
- "three's company" chrissy farce
- "three's company" sitcom 1970s
- 1970s janet "three's company"
- "three's company" chrissy "john ritter"

Tuple length: 3
N. of tuples (max: 84): 10
BootCaT

Using BootCaT

Your Windows Azure Marketplace Account Key

- Remember Account Key on this computer

You are about to query a search engine using the tuples you just created. In order to do this, you must provide a valid Account Key (i.e. a sort of password).

If you do not have an Account Key, click on the button below for instructions on how to obtain one.

Click here if you don't have an Account Key

You must provide a search engine key

< Back  Next >  Quit
BootCaT
Using BootCaT
BootCaT

Using BootCaT
BootCaT

Using BootCaT
BootCaT
Using BootCaT

Click on 'Build corpus' to start building a corpus
Corpus Linguistics
Web as Corpus

BootCaT
Using BootCaT

Corpus saved in /Users/md7/BootCaT Corpora/Jack Tripper
You can find the old Perl code for BootCaT in:
/Applications/BootCaT_front-end.app/Contents/Resources/Java/toolkit/

The process roughly goes as follows:
1. build_random_tuples.pl
2. collect_urls_from_yahoo.pl
3. retrieve_and_clean_pages_from_url_list.pl
4. discard_duplicates.pl
References

