Informatics 1 Data & Analysis
Tutorial 6

Week 8, Semester 2, 2011

- Please attempt all questions on this worksheet in advance of the tutorial, and bring with you all work, including printouts of code and other results. Tutorials cannot function properly unless you do the work in advance.
- Data & Analysis tutorial exercises are not assessed, but they are a compulsory and important part of the course. If you do not do the exercises then you are unlikely to pass the exam.
- Attendance at tutorials is obligatory: if you are ill or otherwise unable to attend one week then email your tutor, and if possible attend another tutorial group in the same week.

Introduction

In this tutorial you will be working with a real corpus, and learn how to analyse it with the *Corpus Query Processor* (CQP), a query engine that searches corpora based on user queries over words, parts of speech, or other markup.

Reading

The following exercises are based largely upon a tutorial written by Stefan Evert, the maintainer of the *Corpus Workbench.* Before starting these exercises, you must read Sections 1 and 2 of the CQP Query Language tutorial. Download this as a PDF document from the course web page, which hosts the correct version for this worksheet.

Setting up the working environment

These exercises must be carried out on one of the DICE machines in the Informatics computer labs, as those have access to both the CQP engine and the necessary corpora. In particular, you will be using a corpus comprising the works of Charles Dickens, annotated with part-of-speech tags and syntactic structure.

[If connecting remotely, please log in first to gateway.inf.ed.ac.uk and then student.compute which has CQP installed. A command-line connection through ssh or PuTTy should be sufficient.]

You must first declare the location of a corpus index. Type the following at the command line in a terminal window:

```
export CORPUS_REGISTRY=/group/corpora/public/corpus_workbench/registry
```

Take care to copy this exactly: it is case sensitive and there are two underscores.

To start CQP itself, enter the command `cqp -e`. (The annotation `-e` enables line-editing; try `cqp -eC` for colour.) You should then see the following in your command prompt:

```
[no corpus]>
```

This means that `cqp` is active but no specific *corpus* has been selected yet. In order to select the `DICKENS` corpus type

```
[no corpus]> DICKENS;
```

You will then see the following:

```
DICKENS>
```

Every command line in `cqp` must finish with a semicolon '`;`'. You can now get more specific information about this particular corpus by typing:

```
DICKENS> info;
```

Press the space bar to display the next page and '`q`' to get back to the command prompt. To leave the program, enter the command '`exit;`'.

1 Concordances

1.1 Concordance of a given word

Now that the working environment has been set, we can start performing some queries on the DICKENS corpus. Searching for concordances (i.e. all occurrences of a given word, displayed in context) is a basic form of querying a corpus. To search for a specific word, such as "dear", you should type:

```
DICKENS> [word = "dear"];
```

which can be abbreviated to

```
DICKENS> "dear";
```

As previously, press space for the next page and '`q`' to get back to the command prompt.

We can make any search case insensitive with the '`%c`' modifier:

```
DICKENS> "dear" %c;
```

We can search for more than one word after another by adding more queries:

```
DICKENS> "dear" "friend";
```

1.2 More complex queries

The use of *regular expressions* considerably extends the power of the CQP query language. CQP uses the following format for regular expressions:

`exp1 exp2`: first `exp1` then `exp2` in sequence
`exp*`: zero or more occurrences of `exp`
`exp?`: zero or one occurrences of `exp`
`exp+`: one or more occurrences of `exp`
`exp1|exp2`: either `exp1` or `exp2`

For example, the following query retrieves four variations of the word “regular”: regular, regulars, regularly, regularity.

```
DICKENS> "regular(|s|ly|ity)";
```

A single dot (‘.’) can be used as a wildcard in CQP, matching any single character. For example, this query finds words such as “than”, “then” and “thin”:

```
DICKENS> "th.n";
```

while the following query finds words starting “un. . . ” and ending “. . . ly”, , such as “unusually” and “unsuccessfully”:

```
DICKENS> "un.*ly";
```

The use of boolean logic extends further the power of CQP queries. The operators ‘&’ (and) and ‘|’ (or) can be used to combine two tests, while ‘!=’ tests for inequality. For instance, the following query would find all occurrences of words beginning “un” and ending “ly”, but different from “unlikely”:

```
DICKENS> [(word = "un.*ly") & (word != "unlikely")];
```

Similarly, the following query would find all occurrences of the words “man” and “woman”:

```
DICKENS> [(word = "man") | (word = "woman")];
```

Question 1 - Concordance of a given bigram. Retrieve all occurrences of the bigram “great deal” in the corpus. Is this query affected by making it case insensitive?

Question 2 - Concordance of variations of a given word. Retrieve variations of the word “kiss” in the corpus: “kisses”, “kissing”, ... Is this query affected by making it case insensitive?

2 Annotations

2.1 Displaying linguistic annotations

When working with `cqp` you may want to customise how the results of queries are displayed. For example, we can select whether or not to display the annotations on the corpus. By default these are turned off, but they can be enabled with

`DICKENS> show +pos +lemma;` (Query results will *show* annotations)

and disabled with

`DICKENS> show -pos -lemma;` (Query results will *hide* annotations)

Try some of your queries again, and look at the annotations.

2.2 Accessing linguistic annotations

Once we have annotations on text, we can use these make more refined searches. For example, with the *pos* annotations for parts of speech we can search for uses of adjectives, verbs or nouns in a given corpus. Switch on pos tagging with “`show +pos;`” and do some searches to identify the different annotations that used for verbs, adjectives and nouns.

These tags are taken from the Penn Treebank project. You can find out more about them by looking at Sections 1–3 of the following guide:

`http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/Penn-Treebank-Tagset.pdf`

In the same way that a query "`[word = "Dear"]`" will find a particular word in the corpus, a query "`[pos = "NN"]`" will find all (singular) nouns in the corpus. Both "word" and "pos" queries can use regular expressions and logical operations to build more complex queries.

Question 3. What is the common pattern for the pos annotations used for different types of verbs? Use a regular expression to formulate a query that retrieves all occurrences of verbs in the corpus.

Question 4. Write a query to retrieve all tokens which are *not nouns*.

Question 5. Retrieve all occurrences of the word "lock" where it is used as a noun.

Question 6. The "lemma" annotation in `cqp` contains the *headword*: the principal dictionary word associated with a word. Switch on lemma tagging with "`show +lemma;`" and try some queries.

Use this to formulate an alternative (case-insensitive) query for all variations of the word "kiss" in the corpus. Does this return the same results as your answer to Question 2?

3 Frequencies of a given word

Frequency information obtained from corpora can be useful for language research. Here we look at finding the absolute frequency — the number of occurrences — of certain features in the corpus.

To find the total number of tokens in the corpus, type this:

```
DICKENS> Q1 = []; size Q1;
```

Here we have named the result of the query `Q1`, and then retrieved its size. In this case, the query is one that matches every word in the corpus (and, in fact, every punctuation mark too).

A similar query counts the absolute frequency of a specific word:

```
DICKENS> Q2 = [word = "dear"]; size Q2;
```

This query finds the absolute frequency of all bigrams consisting of the word "dear" followed by a noun:

```
DICKENS> Q3 = [word = "dear"] [pos = "N.*"]; size Q3;
```

We can refine this to list the absolute frequency of the bigram grouped by which noun it is that follows "dear":

```
DICKENS> Q3 = [word = "dear"] [pos = "N.*"]; count Q3 by word;
```

A slightly different presentation is:

```
DICKENS> Q3 = [word = "dear"] [pos = "N.*"];
DICKENS> group Q3 matchend word by match word;
```

Here "`matchend word`" refers to the word at the end of the bigram match (the noun) and "`match word`" refers to the word at the start of the bigram (in this case, "dear"). You can also group by pos, or lemma.

A tricky point: Suppose we want to find the most common word in the corpus. A natural query would be:

```
DICKENS> Q4 = []; count Q4 by word;
```

This works, but also includes punctuation marks and white space amongst words. Instead, the following works better:

```
DICKENS> Q4 = [word = "[a-zA-Z].*"]; count Q4 by word;
```

Here the range expression "`[a-zA-Z]`" matches any letter, either lower or upper case.

Question 7 - Absolute Frequency. Find the absolute frequency of the word "girl" in the corpus, ignoring any case differences.

Question 8 - Relative Frequency. Calculate the relative frequency of the word "girl" in the corpus (the number of occurrences as a fraction of the total number of words).

Question 9. Find the most common bigram in the Dickens corpus.

4 Collocations

Question 10. Is the most common bigram a collocation?

Question 11. List the 10 most common adjective-noun pairs. Which of these would you classify as collocations?

Question 12. For a given adjective-noun pair AB (i.e., adjective A followed by noun B) we can create a *contingency table* of frequency observations. Such contingency tables will be used later in the course. The table for AB looks as follows:

	A	$\neg A$
B	$F_{A\,B}$	$F_{\neg A\,B}$
$\neg B$	$F_{A\,\neg B}$	$F_{\neg A\,\neg B}$

where:

$F_{A\,B}$ is the absolute frequency of adjective A followed by noun B
$F_{\neg A\,B}$ is the absolute frequency of any adjective but A followed by noun B
$F_{A\,\neg B}$ is the absolute frequency of adjective A followed by any noun but B
$F_{\neg A\,\neg B}$ is the absolute frequency of any adjective but A followed by any noun but B

Construct the contingency table of case-insensitive frequency observations for the bigram "great deal". What (if anything) does this table tell us about the bigram?

Type "`exit;`" to quit the program.

5 Tutorial Discussion

- Discuss uses of corpora in Informatics.
- Discuss uses of corpora in Linguistics.
- Discuss how the above uses are related.