Lab 1

1 Creating a co-occurence matrix

Note: The following steps should be carried through in the directory containing the script print_long_distance_bigrams.pl and the file general.corpus.en.tgd (this is what will happen anyway if you follow the first set of instructions below).

• Copy archive with materials (note period, meaning: use current directory as destination!), unzip it and move into the corresponding directory (use ls to see what's inside the directory):

```
$ cp /mnt/data/tplab-dissect.zip .
$ unzip tplab-dissect.zip
$ cd tplab-dissect
```

• "Cleaning" the corpus, keeping verbs, nouns and adjectives only, in their lemma form (and removing "<unknown>" lemmas):

```
$ awk '$2~/^VB/{print $3 "-v"} $2~/^NN/{print $3 "-n"}
$2~/^JJ/{print $3 "-a"}' <general.corpus.en.tgd | grep -v
"<unknown>" > filtered.general.corpus.en.tgd
```

• Counting the content word frequencies, and picking only those that occur at least 20 times as row (and column elements) for our co-occurrence matrix (issue the wc command to check how many they are):

```
$ sort filtered.general.corpus.en.tgd | uniq -c |
awk '{print $2 "\t" $1}' | sort -nrk2 > lemma.counts

$ awk '$2>=20{print $1}' lemma.counts > input.rows
```

• We need identical row and column word element lists since they are both required by the DISSECT toolkit as input files (in principle, they could differ):

```
$ cp input.rows input.cols
```

Collect co-occurrences within a window of 10 words:

```
$ print_long_distance_bigrams.pl -m0 -w9 -u input.rows
filtered.general.corpus.en.tgd | sort | uniq -c |
awk '{print $2 "\t" $3 "\t" $1; print $3 "\t" $2 "\t" $1}'
> input.sm
```

2 Creating a semantic space

• Download DISSECT toolkit

If you are using the cluster, you need to set https_proxy first. Also, make sure that you are on compute-x-x and not masterclic4

```
$ export https_proxy=http://proxy.unitn.it:3128

$ git clone https://github.com/composes-toolkit/dissect
```

Add the toolkit directory to PYTHONPATH:

```
$ export PYTHONPATH="<toolkit directory>/src/":$PYTHONPATH
```

• Build a semantic space from the co-occurrence matrix, with PMI weighting scheme and reduce the dimension to 100 with SVD

```
$ export PIPELINES="<toolkit directory>/src/pipelines"
$ /opt/python/bin/python2.7 $PIPELINES/build_core_space.py
-i input --input_format=sm -o out -w ppmi -r svd_100
```

• Find the top 10 neighbors of "cat-n" in this space

```
$ echo "cat-n" > word_list.txt
$ /opt/python/bin/python2.7 $PIPELINES/compute_neighbours.py
-i word_list.txt -n 10 -s out/CORE_SS.input.ppmi.svd_100.pkl
-o out -m cos
$ cat out/NEIGHBOURS.word_list.txt.CORE_SS.input.ppmi.svd_100.cos
```