Running the Koala word sense disambiguators

Richard Johansson
richard.johansson@gu.se

Version 0.1: June 11, 2016

1 Overview

This document explains how to run the word sense disambiguation (WSD) tools developed during the Koala project.

1.1 Installation

- Download the directory containing the tools from the Koala repository:
  `https://svn.spraakdata.gu.se/repos/koala/wsd`
- Download the following two files and put them into the directory `wsd/models/scouse`:
  - `http://demo.spraakdata.gu.se/richard/scouse/models/ALL_512_128_w10_A2_140403_ctx1.bin`
  - `http://demo.spraakdata.gu.se/richard/scouse/models/lem_cbow0_s512_w10_NEW2_ctx.bin`

1.2 Quick example

Go into the `wsd` directory, and then run

```
curl 'https://spraakbanken.gu.se/ws/korp/annotate?text=Bandet+spelar+rock.' \
 | sh scripts/xml_to_tab.sh | sh scripts/scousewsd.sh
```

2 Input and output formats

The WSD tool assumes that the input has already been segmented. It does not disambiguate between different possible segmentations into multiword units (MWUs), or between compositional and noncompositional readings of a compound.

The input is a text format consisting of tab-separated columns, where one row corresponds to one token (or MWU, or compound segment). Sentences are separated by blank lines. Currently, only columns 5 and 6 are used by the WSD tools, so the first four columns may be replaced by dummy values. (This may change in future versions of the software.)

The columns are

1. *Token index*, corresponding to the `ref` attribute in Korp's XML format. For MWUs, this is a space-separated list.
2. *Token string*. For MWUs, a space-separated list. Note that the same token string may occur in more than one row, which occurs in case of compositional compounds.
3. *Compound part* indicator. This is `(pfx)` for a compound prefix, `(sfx)` for a suffix, and is empty (_) for non-compounds.
4. *Lemgram ids*. A pipe-separated list of the possible lemgram identifiers for this token.
5. *Simplified lemgram ids*. A pipe-separated list of the possible *simplified* lemgrams for this token: that is, the lemgram ids with the paradigm number removed. The simplified lemgram id can also be heuristically constructed from the token and its part-of-speech tag, in case the real lemgram is lacking in the input. For instance, the adjective *brötig* would not be assigned a lemgram since it is not listed in Språkbanken's resources, but its simplified lemgram would still be `brötig..av`.

6. *SALDO ids*. A pipe-separated list of the possible SALDO ids for this token.

The following example shows how the sentence *Nu väntar adventspynt för hela slanten.* could be translated into the input format. (For reasons of space, the simplified lemma of the MWU is not shown here.)

```
1      Nu                _      nu..ab.1                  nu..ab      nu..1
2      väntar            _      vänta..vb.1|vänta..vb.2   vänta..vb   vänta..1|vänta..2
3      adventspynt       (pfx)  advent.nn.1               advent..nn  advent..1
3      adventspynt       (sfx)  pynt..nn.2|pynt..nn.1     pynt..nn    pynt..2|pynt..3|pynt..1
4 5 6  för hela slanten  _      för_hela_slanten..ab.1    (...)       för_hela_slanten..1
7      .                 _      _                         ...mad      _
```

The output format generated by the WSD tool uses the same columns as the input (although some of the input columns may not be printed by every tool), plus one additional seventh column containing the WSD scores as a pipe-separated list. The order of these scores corresponds to the order of the list of SALDO ids in the input.

In the current WSD tools, the scores can be interpreted as probabilities, but in the general case the interpretation of the score does not necessarily have to be probabilistic. A high score will always mean that this sense is preferred by the tool.

The WSD tools will typically not give an output score for every token in the input. In the current implementation, scores are output only if 1) the token has at least 2 SALDO senses listed in column 6, and 2) it is a single word (not a MWU), and 3) it is a content word (not a function word).

The following example shows how the output probabilities generated by a hypothetical WSD tool for the sentence in the example above. In this case, the tool thinks that there is a 60% probability that the SALDO sense of *väntar* is `vänta..1`. (Columns 2–5 are excluded here.)

```
1      (...)  nu..1                      _
2      (...)  vänta..1|vänta..2          0.6|0.4
3      (...)  advent..1                  _
3      (...)  pynt..2|pynt..3|pynt..1    0.75|0.15|0.10
4 5 6  (...)  för_hela_slanten..1        _
7      (...)  _                          _
```

2.1 Running the Korp-XML preprocessor

There is a separate tool that converts from Korp's XML export format into the column-based input format. The conversion tool can be run using the script `scripts/xml_to_tab.sh`. It reads from the standard input and prints to the standard output.

Since MWU segmentation and compound splitting are ambiguous in Korp's data model, this tool uses a number of heuristics in order to convert into the tabbed format, which does not allow ambiguous segmentation. (These heuristics may of course introduce errors.)

Multiword units: If the flag `SPLIT_MWUS` is set to true, all potential MWUs will be represented as separate tokens. If the flag is set to false, MWUs will be included using a left-to-right, longest-match heuristic. For instance, if we see the sequence *en gång till*, the three-word MWU (*en gång till*) will be included in the output, but not the two-word MWU (*en gång*), even if that would be the correct interpretation in that context.

Compounds: If the flag `SPLIT_COMPOUNDS` is set to true, compunds for which no SALDO unit can be found are split into prefix and suffix if available. The SALDO entries corresponding to the lemgrams of the prefix and suffix are looked up automatically. In case a compound is listed as a whole in SALDO, no segmentation is done even if the flag is set to true. For instance, for the word *fotboll*, only the two SALDO senses will be output, not the possible segmentation into *fot+boll*.

3 Running the WSD systems

The package includes a number of WSD tools: three different tools at the time of writing. All the different WSD tools are bundled into the a Java JAR file, which can be executed directly. However, each tool comes with with a separate shell script that defines useful default values, and it is probably easier to use the appropriate script than calling the JAR directly. All the scripts currently assume that they are executed from inside the `wsd` directory, so they will need to be modified if it is necessary to call them from elsewhere.

3.1 The SCOUSE WSD system

The SCOUSE WSD tool[1] was developed at Språkbanken (See Johansson and Nieto Piña (2015b) for a description of its training procedure, and Johansson and Nieto Piña (2015a) for the disambiguation.) It works by mapping a semantic network (in this case SALDO) into a distributional model. This tool is comparatively efficient – it can disambiguates a few thousand instances per second – so this is the tool that is recommended for large-scale use.

To use this tool, run the script `scripts/scousewsd.sh`. In the script, the following options can be modified if necessary:

`SENSE_VECTORS` The file containing the sense vectors.

`CONTEXT_VECTORS` The file containing the context vectors.

`WINDOW` The number of tokens on each side of the target token are considered.

`DECAY` Whether tokens closer to the target are given a higher importance.

`S1_PRIOR` If set to a positive value, the the first sense will be given a preference during disambiguation. If set to zero, all senses are considered equally likely.

3.2 The UKB WSD system

The UKB tool,[2] developed at the University of the Basque Country, is a graph-based WSD system that uses personalized PageRank to disambiguate the ambiguous lemmas (Agirre and Soroa, 2009). In our evaluations, it tends to give comparable or even slightly higher accuracies than SCOUSE WSD, but since it can process only a few instances per second, it is less interesting for processing a large amount of text, although it might be useful in an interactive setting.

The UKB tool is executed by running the script `scripts/ukb.sh`. The options that can be modified in the script are the following:

`EXEC` The location of the UKB executable.

`DICT` The dictionary that defines the lemma-to-sense mapping.

`MODEL` The SALDO graph in UKB's format.

`BATCH_SIZE` The number of sentences to be read from the input before running UKB.

3.3 First-sense WSD

As a dummy baseline, we have also included a system that always selects the sense whose SALDO identifier has the lowest numerical id. For instance, if the SALDO senses `val..2` and `val..3` are available in a context, `val..2` will be selected. It is started by executing `scripts/firstsense.sh`.

[1] http://demo.spraakdata.gu.se/richard/scouse/

[2] `http://ixa2.si.ehu.es/ukb/`; version 2.0 is used here.

4 Known limitations and drawbacks

- no disambiguation of function words (e.g. prepositions)
- no support for multiword units
- no disambiguation between compositional and noncompositional readings of multiwords and compounds
- suboptimal performance in case a lemma is ambiguous

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 33–41, Athens, Greece.

Richard Johansson and Luis Nieto Piña. 2015a. Combining relational and distributional knowledge for word sense disambiguation. In *Proceedings of the 20th Nordic Conference of Computational Linguistics*, pages 69–78, Vilnius, Lithuania. Linköping University Electronic Press, Sweden.

Richard Johansson and Luis Nieto Piña. 2015b. Embedding a semantic network in a word space. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1428–1433, Denver, United States.