More is not always better: balancing sense distributions for all-words Word Sense Disambiguation

This package contains all the data and scripts used to run the experiments presented in the paper and obtain the figures shown in the tables. It is designed in a way that allows **automatic and full reproducibility of the results.** There are only 3 steps required:

1. Download this folder to your computer (Linux/OsX)
2. Run the installation script
3. Run the script for reproducing all the experiments

Table of Content

1) Content of this package

This section explains the meaning of each folder or script contained in this package.

1.1) Scripts

- **install.sh**
    - installation script that will download and install all the required material:
        - IMS WSD system
          http://www.comp.nus.edu.sg/~nlp/software.html
        - WordNet 3.0
          https://wordnet.princeton.edu/wordnet/download
        - SemEval2013 task 12 test dataset
          https://www.cs.york.ac.uk/semeval-2013/task12

- **run_all_experiments.sh**
    - Automatically runs and reproduces all the experiments contained in this paper

- **train_ims_paralel.sh**
    - trains the IMS models for a specific experiment, allowing to split the task into a certain number of subprocesses for efficiency

- **train_ims.sh**
    - trains the IMS models for a set of lemmas

- **evaluate_semeval2013.sh**
    - runs the classification for SemEval2013 using a set of trained models and runs the evaluation using the official scorer

- **evaluate_mfs_lfs.py**
    - From an instance output file, and by calling to the official scorer, computes the figures for the test instances where Most Frequent Sense applies and for the rest

- **semeval2013_to_allwords_format.py**
    - Converts the XML format of the original SemEval2013 dataset to the all-words format expected by the IMS system

1.2) Data

- **evaluation**
    - Folder with the output of the evaluation. For each experiment there are two files (EXP_ID corresponds to the experiment ID, which are explained later in this document):
        - EXP_ID.out → contains the instance output for the experiment (format used by the official SemEval scorer)
        - EXO_ID.figures.txt → contains the evaluation figures obtained by the scripts evaluate_semeval2013.sh and evaluate_mfs_lfs.py

- **experiments_data.tgz**
    - There is a folder for each experiment, with the name of the experiment id, and contains all the training data for the experiment. The format is the same used by the IMS system, where basically there are two files for each lemma.pos, one file contains all the instances with the corresponding contexts, and the other contains the sense keys for the instances

- **models.tgz**
    - Contains the IMS trained models for all the experiments of the paper

- **ims_amended**
    - Amended IMS files (see section below)

- **prop_wn30.original.xml**
    - Original property file for IMS. It will be automatically modified to point to the location of WordNet3.0, once downloaded.
- **sem2013.lemma_pos.list**
    - List of lemmas (and pos) for the semeval2013 dataset. It is used to split the training in several batches, as the training for each word expert is independent of the rest

2) Requirements

These are the requirements before running the installation script:

1) Python3 → the main scripts developed by us are written in Python (version 3 required)
2) NLTK 3.0 library for python: you can install it very easily: http://www.nltk.org/install.html
3) Java8 → This is a requiremenf of the IMS system that we use in our experiment

3) Installation

For installing all the required libraries and resources just run:

$. install.sh

The script will:

- Unzip (you need tgz) the data and models folders
- Download IMS, copy the amended files in the correct place and recompile IMS to generate an updated jar file.
- Download WordNet 3.0 and SemEval2013 dataset
- Convert the SemEval2013 dataset to the proper IMS format

4) Identifiers used for the experiments

It is important to note that the convention used to name the experiments in the paper and in this package differ. In the paper we used numbers from 1 to 11 to simplify the naming of these experiments, while for the folders and files related to these experiments in this package we used more representative names. We will include here a table table that established the correspondence between the identifiers used for the experiments in this package and in the paper (mainly in the results section and in Table 1 that contains all the results).

Paper experiment identifier	Package experiment identifier	Overall Accuracy
1	Bs	65.60
2	Bps	66.80
3	Bpsw	68.90
4	Bsw	69.30
5	BsAp+lfs	63.20
6	BsApw+lfs	62.00
7	BsAw+lfs	67.50
8	Bpsw+gold+1	85.40
9	Bpsw+gold+5	80.40
10	Bpsw+gold+1+lowest_freq	86.80
11	Bpsw+gold+5+lowest_freq.figures.txt	82.00

5) Reproducing all the experiments

You need to make sure that the java version that you are using is 1.8, otherwise the IMS system might not work:

```
$ java -version
java version "1.8.0_73"
Java(TM) SE Runtime Environment (build 1.8.0_73-b02)
Java HotSpot(TM) 64-Bit Server VM (build 25.73-b02, mixed mode)
```

You will need to see the path to your java executable script in 2 scripts:

- Train_ims.sh
  - In the line 4 → JAVA=”path/to/java1.8/java”
- Evaluate_semeval2013.sh
  - Line 16 → JAVA=”path/to/java1.8/java”

Once that you have set the proper thats, you just need to call to the script run_all_experiments.sh, for instance:

```
$ . run_all_experiments.sh > log.out 2> log.err &
```

To make it faster, you can split the training for each experiment in several subprocess. Change the value of the variable *NUM_PROC (by default set to 15)* in the run_all_experiments.sh script. You can set it to 1 if you do not want any parallelisation

The script will call to the training and evaluation for each experiment (the data used for each experiment is in the folder “experiments_data”. It will save the trained models for the folder models/EXP_ID and the results of the evaluation to the files evaluation/EXP_ID.out and evaluation/EXP_ID.figures.txt.

You can also call the script “train_ims_paralel.sh” to train the classifiers for a single experiment:

```
$ train_ims_paralel.sh  PATH_TO_DATA   PATH_TO_FOLDER   NUM_PROC
```

You can also call to the classification and evaluation for a single experiment by using the script evaluate_semeval2013.sh, by providing the path to the model you want to evaluate, and the path to the .out and .figures.txt files where you want to store the results.

```
$ evaluate_semeval2013.sh   PATH_TO_TRAINED_MODEL   file.out > file.figures.txt
```

6) Changes to the IMS system

Under some circumstances and with extended training set, IMS raises an exception. For avoiding this we had to include a couple of minor changes into the IMS code. The corrected files can be found in the folder “ims_amended_files”. The installation script (install.sh) will take care of copying them to the proper place the recompile IMS. The changes are:

- CLibLinearLexeltWriter.java
    - Line 145:
        - Original → if (values.get(i).equals(value))
        - Modified → if (values.get(i) != null && values.get(i).equals(value))
- CStatistic.java
    - Line 481
        - Original → values[i.getValue()] = i.getKey();
        - Modification → surrounded by try/catch block