I Can Sense It: a comprehensive online system for WSD

Salil Joshi[*] *Mitesh M. Khapra* *Pushpak Bhattacharyya*
(1) IBM Research India, Bangalore, India
(2) IBM Research India, Bangalore, India
(3) CSE Department, IIT Bombay, Mumbai, India
saljoshi@in.ibm.com, mikhapra@in.ibm.com, pb@cse.iitb.ac.in

Abstract

We have developed an online interface for running all the current state-of-the-art algorithms for WSD. This is motivated by the fact that exhaustive comparison of a new Word Sense Disambiguation (WSD) algorithm with existing state-of-the-art algorithms is a tedious task. This impediment is due to one of the following reasons: (1) the source code of the earlier approach is not available and there is a considerable overhead in implementing it or (2) the source code/binary is available but there is some overhead in using it due to system requirements, portability issues, customization issues and software dependencies. A simple tool which has no overhead for the user and has minimal system requirements would greatly benefit the researchers. Our system currently supports 3 languages, *viz.*, English, Hindi and Marathi, and requires only a web-browser to run. To demonstrate the usability of our system, we compare the performance of current state-of-the-art algorithms on 3 publicly available datasets.

Keywords: WSD System.

[*]Parts of the work done during M.Tech thesis completed from CSE Department, IIT Bombay

1 Introduction

Several WSD algorithms have been proposed in the past ranging from knowledge based to unsupervised to supervised methods. These algorithms have their own merits and demerits, and hence it is desirable to compare a new algorithm with all of these to put the results in the right perspective. Even when the implementations of these algorithms are publicly available, running them can be cumbersome as it involves the following tedious steps:

1. Resolving portability issues of operating systems (*e.g.*, linux/windows)
2. Adhering to specific input/output formats
3. Installation issues involving software dependencies
4. Run-time issues pertaining to system requirements

The above process needs to be repeated for every algorithm that the user wants to compare her/his system with. Further, in many cases, there is no publicly available implementation of the algorithm, in which case the user has to bear significant overhead of re-implementing these algorithms.

To circumvent the above problems and to ensure ease of use, we have developed an online system, which allows the user to run several state-of-the-art algorithms. There is no overhead for the user, all (s)he needs is a web browser and the input file which may be sense tagged. Further, we also make provision for the developers of the new algorithms to integrate their algorithm in our system. This can be done by implementing a java interface exposed by us and upload the class file on our web-page.

Some of the important aspects of our system are as follows:

1. **Collection of several approaches** - Users can obtain results for state-of-the-art approaches like IMS (Zhong and Ng, 2010), PPR (Agirre et al., 2009), knowledge based approaches (Patwardhan et al., 2005) etc, for an easy comparison of all approaches on a single dataset.
2. **Parallel execution of several algorithms** - The user can choose to run multiple algorithms in parallel, over the same dataset. The associated overhead of scheduling jobs and managing system resources is handled by the server and the user is exempted of these hassles.
3. **Minimum supervision** - After submitting his/her request, the end user can continue with their work without having to constantly monitor the task. Our interface notifies the user when the results are available. Once the users is notified, (s)he needs to download a single zip file which contains all the output files.
4. **User Friendly** - Currently available systems are mostly without a Graphical User Interface (GUI). Our interface is visually aesthetic, can be viewed on different screen resolutions, and is very easy to use.
5. **Easy interpretability of the output** - Our interface provides an option of viewing the output files online where the disambiguated words are shown along with the gloss of the sense present in the wordnets. Most of the available tools only generate the results with the sense offsets, which are machine readable, but make the manual analysis difficult.
6. **Evaluation using standard metrics** - Our system evaluates all the algorithms selected by the user using standard evaluation metrics (precision, recall and F-score). This allows the user to easily compare the performance of the selected algorithms.
7. **Unifies the input/output formats** - Existing systems use non-standard input/output formats, which results in an additional burden of converting the dataset in the required formats. Our system supports different types of input file formats so that less conversions are required while providing the inputs. Further, the outputs are provided in a format compatible with UKB format, which can be easily parsed.

8. **Plug-and-play design** - If an implementation of a new approach is provided, it can be easily plugged into the system. Apart from exposing his/her algorithm to the public, and thereby increasing its visibility/use, this also allows the user to outsource her/his own computational load.

In this system demonstration, we explain the system which powers our interface. The paper is organized as follows: We describe the existing, publicly available systems in section 2. In section 3 we provide technical details about our system. Section 4 summarizes the evaluation results on 3 standard datasets. Section 5 concludes the paper presenting the salient points of our system and some future enhancements for our system.

2 Related Work

There are a few algorithms for which the implementation is publicly available. These include UKB (Agirre et al., 2009), IMS (Zhong and Ng, 2010), SenseLearner (Mihalcea and Csomai, 2005) and SenseRelate (Patwardhan et al., 2005). However, most of these have one or more of the overheads listed above. For example, UKB is currently available only for linux platforms. Further, the user needs to download and install these systems separately and run them on her/his machine which increases the computational cost. SenseLearner has an online interface, but in contrast to our system, it provides only a single algorithm and does not enable the user to compare the performance of different algorithms.

Our system is a one-stop-shop for comparing several algorithms (including UKB, IMS and SenseRelate) with minimum computational and manual overhead for the user. We would like to mention that internally our system uses the implementations provided by UKB, IMS and SenseRelate and hence it would provide the same results as obtained by independently downloading and using these systems. Apart from UKB, IMS and SenseRelate, our system also provides an implementation for McCarthy's approach (Koeling and McCarthy, 2007) and IWSD (Khapra et al., 2010).

3 System Details

Figure 1 shows the main interface of our system. We first provide an overview of the system introducing the inputs which it expects followed by explaining the online output viewer, which is an interesting feature of our system. We also provide details about the mechanism with which new algorithms can be easily added to the system. Kindly refer to the figure while reading this section.

3.1 Interface Design

To support various web browsers on different operating systems, we have designed the web interface using standard open technologies. The interface runs using PHP5[1] on the server side, and for the GUI, we have used a javascript framework *viz.*, ExtJS v4.0[2] which provides a neat and aesthetic display to the user.

3.2 User input

In order to use the system interface, the user needs to provide the following inputs:

1. **Language**: The language of the corpus file(s) for which the WSD algorithm needs to run. As

[1] `http://php.net/downloads.php`
[2] `http://www.sencha.com/products/extjs/`

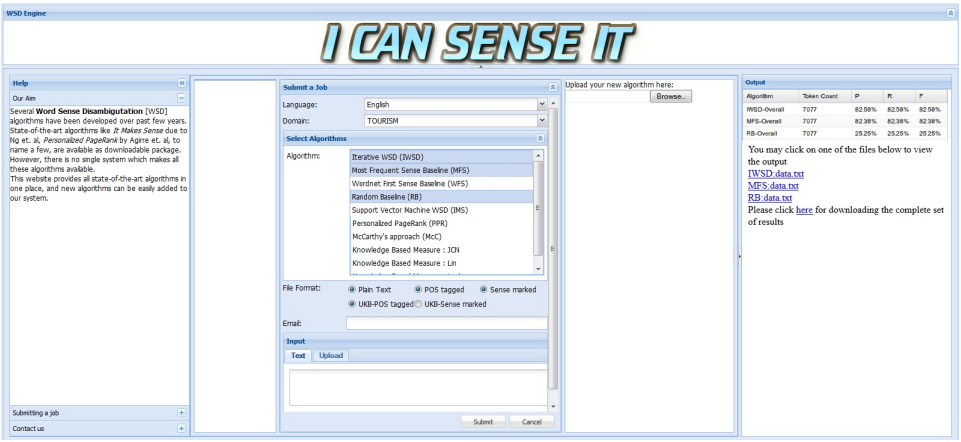

Figure 1: screen-shot of the online interface showing the results of a job along with the links to view the output files in our online output viewer

of now, we provide the user an option of selecting between *English*, *Hindi* and *Marathi* since we have the training datasets and morphological analyzers available for these languages.

2. **Domain**: Some of the state-of-the-art algorithms are domain specific, and in general, the WSD systems show better performance when the domain of the experiment is known in advance. Apart from an option of *Tourism* and *Health* domains for the languages mentioned above, we also support *News* domain for *Hindi*, and *SemCor* domain for *English*.
3. **Algorithms to be run**: The user can select one or more from the following:
   - IWSD (Iterative WSD) - A supervised WSD algorithm by (Khapra et al., 2010)
   - IMS (It Makes Sense) - An SVM based approach by (Zhong and Ng, 2010)
   - PPR (Personalized Page Rank) - A knowledge based approach by (Agirre et al., 2009)
   - McCarthy's approach - An unsupervised state-of-the-art algorithm by (Koeling and McCarthy, 2007)
   - Knowledge based measures - SenseRelate (Patwardhan et al., 2005) supports several knowledge based measures for WSD. We support 3 measures, *viz.*, Lesk, Lin and JCN out of these.
   - RB (Random Baseline)
   - WFS (Wordnet First Sense Baseline)
   - MFS (Most Frequent Sense Baseline)
4. **Input file format**: In the most basic form, the user can simply upload a plain text file containing one sentence per line. However, most algorithms perform better if the data is POS tagged. Our system does not perform POS tagging. Hence, we allow the user to upload POS tagged data in which each sentence is represented in the following format:
   $word_1\_<pos_1> \; word_2\_<pos_2> \cdots \; word_n\_<pos_n>$
   where $<pos_1>$, $<pos_2>$, *etc.*, are the POS tags of the respective words, and can take one of the 4 values, 1: Noun:, 2: Verb, 3: Adverb, 4: Adjective (since currently available wordnets support only these POS tags). If the user has sense marked gold data and wants to evaluate the performance of different algorithms on this data, then he/she can submit the input in the following format:
   $word_1\_<pos_1><offset_1> \; word_2\_<pos_2><offset_2> \cdots \; word_n\_<pos_n><offset_n>$
   where $<offset_1>$, $<offset_2>$, *etc.*, are the wordnet sense offsets of the respective words.

For processing these formats, our system requires a morphological analyzer or stemmer from the respective language. The gold data sense offsets will be compared against the outcome of the algorithm. Our algorithms use Princeton WordNet v2.1 for English. In addition to these simple formats, we also provide support to the following file format which is compatible with UKB:
$word_1$#<$roots_1$>#<pos_1>#<index>#<$offset_1$> $word_2$#<$roots_2$>#<pos_2>#<index>#<$offset_2$> $\cdots$ $word_n$#<$roots_n$>#<pos_n>#<index>#<$offset_n$>
where <$roots_1$>, <$roots_2$>, *etc.*, represent morphological roots of the respective words. <index> represents the position of the word in the sentence and is stored as w_1, w_2 and so on. Please note that this format requires the input data to be at least POS tagged and optionally sense annotated. In case if the data is not sense annotated, the <$offset$> field will be represented with '1' for the words which are to be disambiguated and 0 otherwise. The output files generated by our system follow this format.

5. **E-mail address**: Depending on the size of the input and the number/type of algorithms chosen, the system will take some time to compute the results. For ease of use, an e-mail is sent to the user, once the computation is done. This email specifies a link from where (s)he will be able to download all the results.
6. **Input (Data)**: The user can either type the text to be disambiguated in the text box provided on the submission form, or (s)he can choose to upload a file containing the text. The uploaded file can be a zipped directory, in which case, it will be extracted on the server side, and all the constituent files will be used as the input dataset for running the algorithm.

3.3 Online output viewer

Our system generates the output files in UKB format as stated earlier. This output can be easily parsed, however, it is not suitable for manual analysis. Our interface provides the users with a facility of viewing the output online, where the sense tags are accompanied with the sense gloss and examples as available in the wordnet. This enables the user to easily comprehend the output. Figure 2 shows a screen-shot of the online output viewer. The interface also provides an *output* pane, where the results of the job are summarized, and a link to the results is provided, so that the user can download them. The same link is also sent to the user to the specified e-mail address.

Figure 2: screen-shot of the online interface showing the online output viewer

3.4 Integration of new algorithms

As mentioned earlier, our system can integrate new algorithms on-the-fly. The developers who have a new algorithm, and wish to make it a part of our system, can submit their algorithms online, and such algorithms automatically get added to our system when uploaded.

Currently, our system can not automatically handle the software dependencies exhibited by the new algorithms, if any. In such cases, the new algorithm will not be useful to end users. To prevent this, the developers of the new algorithm can contact us and get the dependencies fixed. Our system runs on a linux based server, and the dependencies must be in the form of publicly available software packages compatible with linux systems.

The interaction between our system and the new algorithm will be in form a shell script, the details of which are provided on our web interface[3]. This shell script can, in turn, call its own resources, binaries and other shell scripts to read the input text files provided in specific format, and produce the output text files in specific format. The detailed instructions for integration, along with the sample text files, can also be accessed from our web interface.

4 Empirical evaluation

To demonstrate the use of our system, we have evaluated the performance of all the algorithms on 3 standard datasets[4]. The results are summarized in table 1. There are several knowledge based measures which SenseRelate supports. We show the results for a representative measure, *viz.*, Lesk (KB-Lesk).

Algorithm	**Tourism**			**Health**			**SemCor**		
	P%	R%	F%	P%	R%	F%	P%	R%	F%
IWSD	77.00	76.66	76.83	78.78	78.42	78.60	67.42	66.27	66.82
PPR	53.10	53.10	53.10	51.10	51.10	51.10	50.42	50.42	50.42
IMS	78.82	78.76	78.79	79.64	79.59	79.61	68.38	67.82	68.02
McCarthy's approach	51.85	49.32	50.55	N/A	N/A	N/A	N/A	N/A	N/A
RB	25.50	25.50	25.50	24.61	24.61	24.61	21.08	21.08	21.08
WFS	62.15	62.15	62.15	64.67	64.67	64.67	63.49	63.49	63.49
MFS	77.60	75.2	76.38	79.43	76.98	78.19	67.75	65.87	66.57
KB-Lesk	50.86	50.84	50.85	51.80	51.78	51.79	39.59	39.24	39.41

Table 1: Precision, Recall and F-scores for various algorithms supported by our system

5 Conclusion

In this paper, we have described a system which allows for easy comparison of several state-of-the-art WSD systems. Since our system is an online system, it minimizes the overhead on the end user by eliminating the installation issues. Our system only depends on a morphological analyzer for the input data. Further, since all the computation takes place on our server, it drastically reduces the system requirements and computational efforts for the end user. The interface to the system is extremely user friendly, aesthetic and supports multiple input file formats. New algorithms can be integrated easily with our system with minimal additional efforts on part of the developer. The system also provides an online results viewer which is useful for manual analysis as it provides the sense gloss and examples for each disambiguated word.

In the future, we would like our system to support more and more languages.

[3] http://www.cfilt.iitb.ac.in/wsd-demo

[4] http://www.cfilt.iitb.ac.in/wsd/annotated_corpus

References

Agirre, E., Lacalle, O. L. D., and Soroa, A. (2009). Knowledge-based wsd on specific domains: Performing better than generic supervised wsd. In *In Proceedings of IJCAI*.

Khapra, M., Shah, S., Kedia, P., and Bhattacharyya, P. (2010). Domain-specific word sense disambiguation combining corpus basedand wordnet based parameters. In *Proc. of GWC*, volume 10.

Koeling, R. and McCarthy, D. (2007). Sussx: Wsd using automatically acquired predominant senses. In *Proceedings of ACL/SIGLEX SemEval*, pages 314–317.

Mihalcea, R. and Csomai, A. (2005). Senselearner: Word sense disambiguation for all words in unrestricted text. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 53–56. Association for Computational Linguistics.

Patwardhan, S., Banerjee, S., and Pedersen, T. (2005). Senserelate:: Targetword: a generalized framework for word sense disambiguation. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 73–76. Association for Computational Linguistics.

Zhong, Z. and Ng, H. (2010). It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, pages 78–83. Association for Computational Linguistics.