UNIBA: JIGSAW algorithm for Word Sense Disambiguation

P. Basile and M. de Gemmis and A.L. Gentile and P. Lops and G. Semeraro
Department of Computer Science - University of Bari - Via E. Orabona, 4 70125 Bari ITALY
{basilepp, degemmis, al.gentile, lops, semeraro}@di.uniba.it

Abstract

Word Sense Disambiguation (WSD) is traditionally considered an AI-hard problem. A breakthrough in this field would have a significant impact on many relevant web-based applications, such as information re-

trivial and information extraction. This paper describes JIGSAW, a knowledge-based WSD system that attempts to disambiguate all words in a text by exploiting WordNet\(^1\) senses. The main assumption is that a specific strategy for each Part-Of-Speech (POS) is better than a single strategy. We evaluated the accuracy of JIGSAW on SemEval-2007 task 1 competition\(^2\). This task is an application-driven one, where the application is a fixed cross-lingual information retrieval system. Participants disambiguate text by assigning WordNet synsets, then the system has to do the expansion to other languages, index the expanded documents and run the retrieval for all the languages in batch. The retrieval results are taken as a measure for the effectiveness of the disam-

biguation.

1 The JIGSAW algorithm

The goal of a WSD algorithm consists in assigning a word \(w_i\) occurring in a document \(d\) with its appropriate meaning or sense \(s_i\), by exploiting the context \(C\) in which \(w_i\) is found. The context \(C\) for \(w_i\) is defined as a set of words that precede and follow \(w_i\). The sense \(s\) is selected from a predefined set of possibilities, usually known as sense inventory. In the proposed algorithm, the sense inventory is obtained from WordNet 1.6, according to SemEval-2007 task 1 instructions. JIGSAW is a WSD algorithm based on the idea of combining three different strategies to disambiguate nouns, verbs, adverbs and adjectives. The main motivation behind our approach is that the effectiveness of a WSD algorithm is strongly influenced by the POS tag of the target word. An adaptation of Lesk dictionary-based WSD algorithm has been used to disambiguate adjectives and ad-

verbs (Banerjee and Pedersen, 2002), an adaptation of the Resnik algorithm has been used to disam-

biguate nouns (Resnik, 1995), while the algorithm we developed for disambiguating verbs exploits the nouns in the context of the verb as well as the nouns both in the glosses and in the phrases that WordNet utilizes to describe the usage of a verb. JIGSAW takes as input a document \(d = \{w_1, w_2, \ldots, w_n\}\) and returns a list of WordNet synsets \(X = \{s_1, s_2, \ldots, s_k\}\) in which each element \(s_i\) is obtained by disam-

biguating the target word \(w_i\) based on the information obtained from WordNet about a few immediately surrounding words. We define the context \(C\) of the target word to be a window of \(n\) words to the left and another \(n\) words to the right, for a total of \(2n\) surrounding words. The algorithm is based on three different procedures for nouns, verbs, adverbs and adjectives, called JIGSAW\(_{\text{noun}}\), JIGSAW\(_{\text{verb}}\), JIGSAW\(_{\text{adverb}}\), JIGSAW\(_{\text{adjective}}\), respectively. More details for each one of the above mentioned procedures follow.

1.1 JIGSAW\(_{\text{noun}}\)

The procedure is obtained by making some variations to the algorithm designed by Resnik (1995) for disambiguating noun groups. Given a set of nouns \(W = \{w_1, w_2, \ldots, w_n\}\), obtained from document \(d\), with each \(w_i\) having an associated sense inven-

tory \(S_i = \{s_{i1}, s_{i2}, \ldots, s_{ik}\}\) of possible senses, the goal is assigning each \(w_i\) with the most appropri-

tate sense \(s_{ih}\) \(\in S_i\), according to the similarity of \(w_i\) with the other words in \(W\) (the context for \(w_i\)). The idea is to define a function \(\varphi(w_i, s_{ij})\), \(w_i \in W\), \(s_{ij} \in S_i\), that computes a value in \([0, 1]\) representing the confidence with which word \(w_i\) can be assigned with sense \(s_{ij}\). The intuition behind this algorithm is essentially the same exploited by Lesk (1986) and other authors: The most plausible assignment of senses to multiple co-occurring words is the one that maximizes relatedness of meanings among the cho-

---

\(^1\)http://wordnet.princeton.edu/
\(^2\)http://www.senseval.org/
sen senses. JIGSAW nouns differs from the original algorithm by Resnik (1995) in the similarity measure used to compute relatedness of two senses. We adopted the Leacock-Chodorow measure (Leacock and Chodorow, 1998), which is based on the length of the path between concepts in an IS-A hierarchy. The idea behind this measure is that similarity between two synsets, $s_1$ and $s_2$, is inversely proportional to their distance in the WordNet IS-A hierarchy. The distance is computed by finding the most specific subsumer (MSS) between $s_1$ and $s_2$ in the WordNet hierarchy is a subsumer, the MSS is the one at the lowest level) and counting the number of nodes in the path between $s_1$ and $s_2$ that traverse their MSS. We extended this measure by introducing a parameter $k$ that limits the search for the MSS to $k$ ancestors (i.e. that climbs the WordNet IS-A hierarchy until either it finds the MSS or $k + 1$ ancestors of both $s_1$ and $s_2$ have been explored). This guarantees that “too abstract” (i.e. “less informative”) MSSs will be ignored. In addition to the semantic similarity function, JIGSAW nouns differs from the Resnik algorithm in the use of:

1. a Gaussian factor $G$, which takes into account the distance between the words in the text to be disambiguated;
2. a factor $R$, which gives more importance to the synsets that are more common than others, according to the frequency score in WordNet;
3. a parametrized search for the MSS between two concepts (the search is limited to a certain number of ancestors).

Algorithm 1 describes the complete procedure for the disambiguation of nouns. This algorithm considers the words in $W$ pairwise. For each pair $(w_i, w_j)$, the most specific subsumer $MSS_{ij}$ is identified, by reducing the search to $depth1$ ancestors at most. Then, the similarity $sim(w_i, w_j, depth2)$ between the two words is computed, by reducing the search for the MSS to $depth2$ ancestors at most. $MSS_{ij}$ is considered as supporting evidence for those synsets $s_{ik}$ in $S_i$ and $s_{jh}$ in $S_j$ that are descendants of $MSS_{ij}$. The MSS search is computed choosing the nearest MSS in all pairs of synsets $s_{ik}$,$s_{jh}$. Likewise, the similarity for $(w_i, w_j)$ is the max similarity computed in all pairs of $s_{ik}$,$s_{jh}$ and is weighted by a gaussian factor that takes into account the position of $w_i$ and $w_j$ in $W$ (the shorter is the distance between the words, the higher is the weight). The value $\varphi(i, k)$ assigned to each candidate synset $s_{ik}$ for the word $w_i$ is the sum of two elements. The first one is the proportion of support it received, out of the support possible, computed as $sup_{ik}/norm_i$ in Algorithm 1. The other element that contributes to $\varphi(i, k)$ is a factor $R(k)$ that takes into account the rank of $s_{ik}$ in WordNet, i.e. how common is the sense $s_{ik}$ for the word $w_i$. $R(k)$ is computed as:

$$R(k) = 1 - 0.8 \frac{k}{n-1}$$

where $n$ is the cardinality of the sense inventory $S_i$ for $w_i$, and $k$ is the rank of $s_{ik}$ in $S_i$, starting from 0.

Finally, both elements are weighted by two parameters: $\alpha$, which controls the contribution given...
to $\varphi(i, k)$ by the normalized support, and $\beta$, which controls the contribution given by the rank of $s_{ik}$. We set $\alpha = 0.7$ and $\beta = 0.3$. The synset assigned to each word in $W$ is the one with the highest $\varphi$ value. Notice that we used two different parameters, $depth_1$ and $depth_2$ for setting the maximum depth for the search of the MSS: $depth_1$ limits the search for the MSS computed in the similarity function, while $depth_2$ limits the computation of the MSS used for assigning support to candidate synsets. We set $depth_1 = 6$ and $depth_2 = 3$.

1.2 JIGSAW verbs

Before describing the JIGSAW verbs procedure, the description of a synset must be defined. It is the string obtained by concatenating the gloss and the sentences that WordNet uses to explain the usage of a synset. First, JIGSAW verbs includes, in the context $C$ for the target verb $w_i$, all the nouns in the window of $2n$ words surrounding $w_i$. For each candidate synset $s_{ik}$ of $w_i$, the algorithm computes $nouns(i, k)$, that is the set of nouns in the description for $s_{ik}$.

$$\max_{j,k} = \max_{w_l \in nouns(i, k)} \{ \text{sim}(w_j, w_i, \text{depth}) \}$$

where $\text{sim}(w_j, w_i, \text{depth})$ is defined as in JIGSAW nouns. In other words, $\max_{j,k}$ is the highest similarity value for $w_j$ wrt the nouns related to the $k$-th sense for $w_i$. Finally, an overall similarity score among $s_{ik}$ and the whole context $C$ is computed:

$$\varphi(i, k) = R(k) \cdot \frac{\sum_{w_j \in C} \text{sim}(w_j, w_i, \text{depth}) \cdot \max_{j,k}}{\sum_{w_l \in C} \text{sim}(w_l, w_i, \text{depth})}$$

where $R(k)$ is defined as in Equation 1 with a different constant factor (0.9) and $\text{sim}(w_i, w_j)$ is the same Gaussian factor used in JIGSAW nouns, that gives a higher weight to words closer to the target word. The synset assigned to $w_i$ is the one with the highest $\varphi$ value. Algorithm 2 provides a detailed description of the procedure.

1.3 JIGSAW others

This procedure is based on the WSD algorithm proposed by Banerjee and Pedersen (2002). The idea is to compare the glosses of each candidate sense for the target word to the glosses of all the words in its context. Let $W_i$ be the sense inventory for the target word $w_i$. For each $s_{ik} \in W_i$, JIGSAW others computes the string $\text{targetGloss}_{ik}$ that contains the words in the gloss of $s_{ik}$. Then, the procedure computes the string $\text{contextGloss}_i$, which contains the words in the glosses of all the synsets corre-

Algorithm 2: The procedure for the disambiguation of verbs

1: procedure \text{JIGSAWverbs}($w_i$, $d$, $\text{depth}$) \triangleright finds the proper synset of a polysemous verb $w_i$ in document $d$
2: \quad $C \leftarrow \{w_1, ..., w_n\}$ \triangleright $C$ is the context for $w_i$. For example, $C = \{w_1, w_2, w_3, w_4, w_5\}$, if the sequence of words \{w_1, w_2, w_3, w_4, w_5\} occurs in $d$, $w_3$ being the target verb, $w_j$ being nouns, $j \neq 3$.
3: \quad $S_i \leftarrow \{s_{i1}, ..., s_{im}\}$ \triangleright $S_i$ is the sense inventory for $w_i$, that is the set of all candidate synsets for $w_i$ returned by WordNet.
4: \quad $s \leftarrow \text{null}$ \triangleright $s$ is the synset to be returned.
5: \quad $\text{score} \leftarrow -\text{MAXDOUBLE}$ \triangleright $\text{score}$ is the similarity score assigned to $s$.
6: \quad $p \leftarrow 1$ \triangleright $p$ is the position of the synsets for $w_i$.
7: \quad \textbf{for all } $\text{s}_{ik} \in S_i$ \textbf{do}
8: \quad \quad $\text{max} \leftarrow \{\text{max}_{ik_1}, ..., \text{max}_{ik_n}\}$
9: \quad \quad $\text{nouns}(i, k) \leftarrow \{\text{noun}_1, ..., \text{noun}_n\}$ \triangleright $\text{nouns}(i, k)$ is the set of all nouns in the description of $s_{ik}$.
10: \quad \quad $\text{sumGauss} \leftarrow 0$
11: \quad \quad $\text{sumTot} \leftarrow 0$
12: \quad \quad \textbf{for all } $j \in C$ \textbf{do} \triangleright computation of the similarity between $C$ and $s_{ik}$
13: \quad \quad \quad $\text{max}_{jk} \leftarrow 0$ \triangleright $\text{max}_{jk}$ is the highest similarity value for $w_j$, wrt the nouns related to the $k$-th sense for $w_i$.
14: \quad \quad \quad $\text{sumGauss} \leftarrow \text{G}(\text{pos}(w_i), \text{pos}(w_j))$ \triangleright Gaussian function which takes into account the difference between the positions of the nouns in $d$.
15: \quad \quad \quad \textbf{for all } $\text{noun} \in \text{nouns}(i, k)$ \textbf{do}
16: \quad \quad \quad \quad $\text{sim} \leftarrow \text{sim}(w_j, \text{noun}, \text{depth})$ \triangleright $\text{sim}$ is the similarity between the $j$-th noun in $C$ and $l$-th noun in $\text{nouns}(i, k)$ if $\text{sim} > \text{max}_{jk}$ then
17: \quad \quad \quad \quad \quad $\text{max}_{jk} \leftarrow \text{sim}$
18: \quad \quad \quad \quad \quad \textbf{end if}
19: \quad \quad \quad \quad \textbf{end for}
20: \quad \quad \quad \textbf{end for}
21: \quad \quad \quad $\text{sumTot} \leftarrow \text{sumTot} + \text{G}(\text{pos}(w_i), \text{pos}(w_j)) \cdot \text{max}_{jk}$
22: \quad \quad \textbf{end for}
23: \quad $\varphi(i, k) \leftarrow R(k) \cdot \text{sumTot} / \text{sumGauss}$
24: \quad $\varphi(i, k) \leftarrow R(k) \cdot \text{sim} \cdot \text{depth}$ if $\varphi(i, k) > \text{score}$ then
25: \quad \quad $\text{score} \leftarrow \varphi(i, k)$
26: \quad \quad $p \leftarrow k$ \quad \textbf{end if}
27: \quad \textbf{end for}
28: \quad $s \leftarrow s_p$
29: \quad $\text{return } s$
30: \textbf{end procedure}
sponding to each word in the context for \( w_i \). Finally, the procedure computes the overlap between \( \text{contextGloss}_i \) and \( \text{targetGloss}_{ik} \), and assigns the synset with the highest overlap score to \( w_i \). This score is computed by counting the words that occur both in \( \text{targetGloss}_{ik} \) and in \( \text{contextGloss}_i \). If ties occur, the most common synset in WordNet is chosen.

2 Experiment

We performed the experiment following the instructions for SemEval-2007 task 1 (Agirre et al., 2007). JIGSAW is implemented in JAVA, by using JWNL library\(^3\) in order to access WordNet 1.6 dictionary. We ran the experiment on a Linux-based PC with Intel Pentium D processor having a speed of 3 GHz and 2 GB of RAM. The dataset consists of 29,681 documents, including 300 topics. Results are reported in Table 1. Only two systems (PART-A and PART-B) participated to the competition, thus the organizers decided to add a third system (ORGANIZERS) developed by themselves. The systems were scored according to standard IR/CLIR measures as implemented in the TREC evaluation package\(^4\). Our system is labelled as PART-A.

| system     | precision | recall | attempted |
|------------|-----------|--------|-----------|
| ORGANIZERS | 0.584     | 0.577  | 93.61%    |
| PART-A     | 0.498     | 0.375  | 75.39%    |
| PART-B     | 0.388     | 0.240  | 61.92%    |

Table 2: WSD results on all-words task

Table 1: SemEval-2007 task 1 Results

All systems show similar results in IR tasks, while their behaviour is extremely different on CLIR task. WSD results are reported in Table 2. These results are encouraging as regard precision, considering that our system exploits only WordNet as knowledge-base, while ORGANIZERS uses a supervised method that exploits SemCor to train a kNN classifier.

3 Conclusions

In this paper we have presented a WSD algorithm that exploits WordNet as knowledge-base and uses three different methods for each part-of-speech. The algorithm has been evaluated by SemEval-2007 task 1. The system shows a good performance in all tasks, but low precision in CLIR evaluation. Probably, the negative result in CLIR task depends on complex interaction of WSD, expansion and indexing. Contrarily to other tasks, organizers do not plan to provide a ranking of systems on SemEval-2007 task 1. As a consequence, the goal of this task - what is the best WSD system in the context of a CLIR system? - is still open. This is why the organizers stressed in the call that this was "a first try".

References

E. Agirre, B. Magnini, o. Lopez de Lacalle, A. Otegi, G. Rigau, and Vossen. 2007. Semeval-2007 task 1: Evaluating wsd on cross-language information retrieval. In Proceedings of SemEval-2007. Association for Computational Linguistics.

S. Banerjee and T. Pedersen. 2002. An adapted lesk algorithm for word sense disambiguation using wordnet. In CICLIng’02: Proc. 3rd Int’l Conf. on Computational Linguistics and Intelligent Text Processing, pages 136–145, London, UK. Springer-Verlag.

C. Leacock and M. Chodorow. 1998. Combining local context and wordnet similarity for word sense identification. In C. Fellbaum (Ed.), WordNet: An Electronic Lexical Database, pages 305–332. MIT Press.

M. Leśk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In Proceedings of the 1986 SIGDOC Conference, pages 20–29. ACM Press.

P. Resnik. 1995. Disambiguating noun groupings with respect to WordNet senses. In Proceedings of the Third Workshop on Very Large Corpora, pages 54–68. Association for Computational Linguistics.

\(^3\)http://sourceforge.net/projects/jwordnet
\(^4\)http://trec.nist.gov/