# Personalizing PageRank for Word Sense Disambiguation

**Eneko Agirre and Aitor Soroa**
IXA NLP Group
University of the Basque Country
Donostia, Basque Contry
{e.agirre,a.soroa}@ehu.es

## Abstract

In this paper we propose a new graph-based method that uses the knowledge in a LKB (based on WordNet) in order to perform unsupervised Word Sense Disambiguation. Our algorithm uses the full graph of the LKB efficiently, performing better than previous approaches in English all-words datasets. We also show that the algorithm can be easily ported to other languages with good results, with the only requirement of having a wordnet. In addition, we make an analysis of the performance of the algorithm, showing that it is efficient and that it could be tuned to be faster.

## 1 Introduction

Word Sense Disambiguation (WSD) is a key enabling-technology that automatically chooses the intended sense of a word in context. Supervised WSD systems are the best performing in public evaluations (Palmer et al., 2001; Snyder and Palmer, 2004; Pradhan et al., 2007) but they need large amounts of hand-tagged data, which is typically very expensive to build. Given the relatively small amount of training data available, current state-of-the-art systems only beat the simple most frequent sense (MFS) baseline[1] by a small margin. As an alternative to supervised systems, knowledge-based WSD systems exploit the information present in a lexical knowledge base (LKB) to perform WSD, without using any further corpus evidence.

[1]This baseline consists of tagging all occurrences in the test data with the sense of the word that occurs more often in the training data

Traditional knowledge-based WSD systems assign a sense to an ambiguous word by comparing each of its senses with those of the surrounding context. Typically, some semantic similarity metric is used for calculating the relatedness among senses (Lesk, 1986; McCarthy et al., 2004). One of the major drawbacks of these approaches stems from the fact that senses are compared in a pairwise fashion and thus the number of computations can grow exponentially with the number of words. Although alternatives like simulated annealing (Cowie et al., 1992) and conceptual density (Agirre and Rigau, 1996) were tried, most of past knowledge based WSD was done in a suboptimal word-by-word process, i.e., disambiguating words one at a time.

Recently, graph-based methods for knowledge-based WSD have gained much attention in the NLP community (Sinha and Mihalcea, 2007; Navigli and Lapata, 2007; Mihalcea, 2005; Agirre and Soroa, 2008). These methods use well-known graph-based techniques to find and exploit the structural properties of the graph underlying a particular LKB. Because the graph is analyzed as a whole, these techniques have the remarkable property of being able to find globally optimal solutions, given the relations between entities. Graph-based WSD methods are particularly suited for disambiguating word sequences, and they manage to exploit the interrelations among the senses in the given context. In this sense, they provide a principled solution to the exponential explosion problem, with excellent performance.

Graph-based WSD is performed over a graph composed by senses (nodes) and relations between pairs of senses (edges). The relations may be of several types (lexico-semantic, coocurrence relations, etc.) and may have some weight attached to them. The disambiguation is typically performed by applying a ranking algorithm over the graph, and then assigning the concepts with highest rank to the corresponding words. Given the computational cost of using large graphs like WordNet, many researchers use smaller subgraphs built online for each target context.

In this paper we present a novel graph-based WSD algorithm which uses the full graph of WordNet efficiently, performing significantly better that previously published approaches in English all-words datasets. We also show that the algorithm can be easily ported to other languages with good results, with the only requirement of having a wordnet. The algorithm is publicly available[2] and can be applied easily to sense inventories and knowledge bases different from WordNet. Our analysis shows that our algorithm is efficient compared to previously proposed alternatives, and that a good choice of WordNet versions and relations is fundamental for good performance.

The paper is structured as follows. We first describe the PageRank and Personalized PageRank algorithms. Section 3 introduces the graph based methods used for WSD. Section 4 shows the experimental setting and the main results, and Section 5 compares our methods with related experiments on graph-based WSD systems. Section 6 shows the results of the method when applied to a Spanish dataset. Section 7 analyzes the performance of the algorithm. Finally, we draw some conclusions in Section 8.

## 2 PageRank and Personalized PageRank

The celebrated PageRank algorithm (Brin and Page, 1998) is a method for ranking the vertices in a graph according to their relative structural importance. The main idea of PageRank is that whenever a link from $v_i$ to $v_j$ exists in a graph, a vote from node $i$ to node $j$ is produced, and hence the rank of node $j$ increases. Besides, the strength of the vote from $i$ to $j$ also depends on the rank of node $i$: the more important node $i$ is, the more strength its votes will have. Alternatively, PageRank can also be viewed as the result of a random walk process, where the final rank of node $i$ represents the probability of a random walk over the graph ending on node $i$, at a sufficiently large time.

Let $G$ be a graph with $N$ vertices $v_1, \ldots, v_N$ and $d_i$ be the outdegree of node $i$; let $M$ be a $N \times N$ transition probability matrix, where $M_{ji} = \frac{1}{d_i}$ if a link from $i$ to $j$ exists, and zero otherwise. Then, the calculation of the *PageRank vector* $\mathbf{Pr}$ over $G$ is equivalent to resolving Equation (1).

$$\mathbf{Pr} = cM\mathbf{Pr} + (1-c)\mathbf{v} \quad (1)$$

In the equation, $\mathbf{v}$ is a $N \times 1$ vector whose elements are $\frac{1}{N}$ and $c$ is the so called *damping factor*, a scalar value between $0$ and $1$. The first term of the sum on the equation models the voting scheme described in the beginning of the section. The second term represents, loosely speaking, the probability of a surfer randomly jumping to any node, e.g. without following any paths on the graph. The damping factor, usually set in the $[0.85..0.95]$ range, models the way in which these two terms are combined at each step.

The second term on Eq. (1) can also be seen as a smoothing factor that makes any graph fulfill the property of being aperiodic and irreducible, and thus guarantees that PageRank calculation converges to a unique stationary distribution.

In the traditional PageRank formulation the vector $\mathbf{v}$ is a stochastic normalized vector whose element values are all $\frac{1}{N}$, thus assigning equal probabilities to all nodes in the graph in case of random jumps. However, as pointed out by (Haveliwala, 2002), the vector $\mathbf{v}$ can be non-uniform and assign stronger probabilities to certain kinds of nodes, effectively biasing the resulting PageRank vector to prefer these nodes. For example, if we concentrate all the probability mass on a unique node $i$, all random jumps on the walk will return to $i$ and thus its rank will be high; moreover, the high rank of $i$ will make all the nodes in its vicinity also receive a high rank. Thus, the importance of node $i$ given by the initial distribution of $\mathbf{v}$ spreads along the graph on successive iterations of the algorithm.

In this paper, we will use *traditional PageRank* to refer to the case when a uniform $\mathbf{v}$ vector is used in Eq. (1); and whenever a modified $\mathbf{v}$ is used, we will call it *Personalized PageRank*. The next section shows how we define a modified $\mathbf{v}$.

PageRank is actually calculated by applying an iterative algorithm which computes Eq. (1) successively until convergence below a given threshold is achieved, or, more typically, until a fixed number of iterations are executed.

Regarding PageRank implementation details, we chose a damping value of $0.85$ and finish the calculation after $30$ iterations. We did not try other

[2]`http://ixa2.si.ehu.es/ukb`

damping factors. Some preliminary experiments with higher iteration counts showed that although sometimes the node ranks varied, the relative order among particular word synsets remained stable after the initial iterations (cf. Section 7 for further details). Note that, in order to discard the effect of *dangling nodes* (i.e. nodes without outlinks) we slightly modified Eq. (1). For the sake of brevity we omit the details, which the interested reader can check in (Langville and Meyer, 2003).

## 3 Using PageRank for WSD

In this section we present the application of PageRank to WSD. If we were to apply the traditional PageRank over the whole WordNet we would get a context-independent ranking of word senses, which is not what we want. Given an input piece of text (typically one sentence, or a small set of contiguous sentences), we want to disambiguate all open-class words in the input taken the rest as context. In this framework, we need to rank the senses of the target words according to the other words in the context. Theare two main alternatives to achieve this:

- To create a subgraph of WordNet which connects the senses of the words in the input text, and then apply traditional PageRank over the subgraph.
- To use Personalized PageRank, initializing $\mathbf{v}$ with the senses of the words in the input text

The first method has been explored in the literature (cf. Section 5), and we also presented a variant in (Agirre and Soroa, 2008) but the second method is novel in WSD. In both cases, the algorithms return a list of ranked senses for each target word in the context. We will see each of them in turn, but first we will present some notation and a preliminary step.

### 3.1 Preliminary step

A LKB is formed by a set of concepts and relations among them, and a dictionary, i.e., a list of words (typically, word lemmas) each of them linked to at least one concept of the LKB. Given any such LKB, we build an undirected graph $G = (V, E)$ where nodes represent LKB concepts ($v_i$), and each relation between concepts $v_i$ and $v_j$ is represented by an undirected edge $e_{i,j}$.

In our experiments we have tried our algorithms using three different LKBs:

- *MCR16 + Xwn*: The Multilingual Central Repository (Atserias et al., 2004b) is a lexical knowledge base built within the MEANING project[3]. This LKB comprises the original WordNet 1.6 synsets and relations, plus some relations from other WordNet versions automatically mapped[4] into version 1.6: WordNet 2.0 relations and eXtended WordNet relations (Mihalcea and Moldovan, 2001) (gold, silver and normal relations). The resulting graph has $99,632$ vertices and $637,290$ relations.
- *WNet17 + Xwn*: WordNet 1.7 synset and relations and eXtended WordNet relations. The graph has $109,359$ vertices and $620,396$ edges
- *WNet30 + gloss*: WordNet 3.0 synset and relations, including manually disambiguated glosses . The graph has $117,522$ vertices and $525,356$ relations.

Given an input text, we extract the list $W_i\ i = 1 \dots m$ of content words (i.e. nouns, verbs, adjectives and adverbs) which have an entry in the dictionary, and thus can be related to LKB concepts. Let $Concepts_i = \{v_1, \dots, v_{i_m}\}$ be the $i_m$ associated concepts of word $W_i$ in the LKB graph. Note that monosemous words will be related to just one concept, whereas polysemous words may be attached to several. As a result of the disambiguation process, every concept in $Concepts_i,\ i = 1, \dots, m$ receives a score. Then, for each target word to be disambiguated, we just choose its associated concept in $G$ with maximal score.

In our experiments we build a context of at least $20$ content words for each sentence to be disambiguated, taking the sentences immediately before and after it in the case that the original sentence was too short.

### 3.2 Traditional PageRank over Subgraph (Spr)

We follow the algorithm presented in (Agirre and Soroa, 2008), which we explain here for completeness. The main idea of the subgraph method is to extract the subgraph of $G_{\text{KB}}$ whose vertices and relations are particularly relevant for a given input

[3]http://nipadio.lsi.upc.es/nlp/meaning

[4]We use the freely available WordNet mappings from http://www.lsi.upc.es/˜nlp/tools/download-map.php

context. Such a subgraph is called a "disambiguation subgraph" $G_{\mathrm{D}}$, and it is built in the following way. For each word $W_i$ in the input context and each concept $v_i \in Concepts_i$, a standard breath-first search (BFS) over $G_{\mathrm{KB}}$ is performed, starting at node $v_i$. Each run of the BFS calculates the minimum distance paths between $v_i$ and the rest of concepts of $G_{\mathrm{KB}}$ . In particular, we are interested in the minimum distance paths between $v_i$ and the concepts associated to the rest of the words in the context, $v_j \in \bigcup_{j \neq i} Concepts_j$. Let $mdp_{v_i}$ be the set of these shortest paths.

This BFS computation is repeated for every concept of every word in the input context, storing $mdp_{v_i}$ accordingly. At the end, we obtain a set of minimum length paths each of them having a different concept as a source. The disambiguation graph $G_D$ is then just the union of the vertices and edges of the shortest paths, $G_D = \bigcup_{i=1}^{m} \{mdp_{v_j} / v_j \in Concepts_i\}$.

The disambiguation graph $G_D$ is thus a subgraph of the original $G_{\mathrm{KB}}$ graph obtained by computing the shortest paths between the concepts of the words co-occurring in the context. Thus, we hypothesize that it captures the most relevant concepts and relations in the knowledge base for the particular input context.

Once the $G_D$ graph is built, we compute the traditional PageRank algorithm over it. The intuition behind this step is that the vertices representing the correct concepts will be more relevant in $G_D$ than the rest of the possible concepts of the context words, which should have less relations on average and be more isolated.

As usual, the disambiguation step is performed by assigning to each word $W_i$ the associated concept in $Concepts_i$ which has maximum rank. In case of ties we assign all the concepts with maximum rank. Note that the standard evaluation script provided in the Senseval competitions treats multiple senses as if one was chosen at random, i.e. for evaluation purposes our method is equivalent to breaking ties at random.

### 3.3 Personalized PageRank (Ppr and Ppr_w2w)

As mentioned before, personalized PageRank allows us to use the full LKB. We first insert the context words into the graph $G$ as nodes, and link them with directed edges to their respective concepts. Then, we compute the personalized PageRank of the graph $G$ by concentrating the initial probability mass uniformly over the newly introduced word nodes. As the words are linked to the concepts by directed edges, they act as source nodes injecting mass into the concepts they are associated with, which thus become relevant nodes, and spread their mass over the LKB graph. Therefore, the resulting personalized PageRank vector can be seen as a measure of the structural relevance of LKB concepts in the presence of the input context.

One problem with Personalized PageRank is that if one of the target words has two senses which are related by semantic relations, those senses reinforce each other, and could thus dampen the effect of the other senses in the context. With this observation in mind we devised a variant (dubbed *Ppr_w2w*), where we build the graph for each target word in the context: for each target word $W_i$, we concentrate the initial probability mass in the senses of the words surrounding $W_i$, but not in the senses of the target word itself, so that context words increase its relative importance in the graph. The main idea of this approach is to avoid biasing the initial score of concepts associated to target word $W_i$, and let the surrounding words decide which concept associated to $W_i$ has more relevance. Contrary to the other two approaches, *Ppr_w2w* does not disambiguate all target words of the context in a single run, which makes it less efficient (cf. Section 7).

## 4 Evaluation framework and results

In this paper we will use two datasets for comparing graph-based WSD methods, namely, the Senseval-2 (S2AW) and Senseval-3 (S3AW) all words datasets (Snyder and Palmer, 2004; Palmer et al., 2001), which are both labeled with WordNet 1.7 tags. We did not use the Semeval dataset, for the sake of comparing our results to related work, none of which used Semeval data. Table 1 shows the results as recall of the graph-based WSD system over these datasets on the different LKBs. We detail overall results, as well as results per PoS, and the confidence interval for the overall results. The interval was computed using bootstrap resampling with 95% confidence.

The table shows that *Ppr_w2w* is consistently the best method in both datasets and for all LKBs. *Ppr* and *Spr* obtain comparable results, which is remarkable, given the simplicity of the *Ppr* algo-

| Senseval-2 All Words dataset | | | | | | | |
|---|---|---|---|---|---|---|---|
| LKB | Method | All | N | V | Adj. | Adv. | Conf. interval |
| MCR16 + Xwn | Ppr | 51.1 | 64.9 | 38.1 | 57.4 | 47.5 | [49.3, 52.6] |
| MCR16 + Xwn | Ppr_w2w | 53.3 | 64.5 | 38.6 | **58.3** | 48.1 | [52.0, 55.0] |
| MCR16 + Xwn | Spr | 52.7 | 64.8 | 35.3 | 56.8 | 50.2 | [51.3, 54.4] |
| WNet17 + Xwn | Ppr | 56.8 | 71.1 | 33.4 | 55.9 | 67.1 | [55.0, 58.7] |
| WNet17 + Xwn | Ppr_w2w | **58.6** | 70.4 | **38.9** | **58.3** | 70.1 | [56.7, 60.3] |
| WNet17 + Xwn | Spr | 56.7 | 66.8 | 37.7 | 57.6 | **70.8** | [55.0, 58.2] |
| WNet30 + gloss | Ppr | 53.5 | 70.0 | 28.6 | 53.9 | 55.1 | [51.8, 55.2] |
| WNet30 + gloss | Ppr_w2w | 55.8 | **71.9** | 34.4 | 53.8 | 57.5 | [54.1, 57.8] |
| WNet30 + gloss | Spr | 54.8 | 68.9 | 35.1 | 55.2 | 56.5 | [53.2, 56.3] |
| MFS | | 60.1 | 71.2 | 39.0 | 61.1 | 75.4 | [58.6, 61.9] |
| SMUaw | | 68.6 | 78.0 | 52.9 | 69.9 | 81.7 | |
| Senseval-3 All Words dataset | | | | | | | |
| LKB | Method | All | N | V | Adj. | Adv. | |
| MCR16 + Xwn | Ppr | 54.3 | 60.9 | 45.4 | 56.5 | **92.9** | [52.3, 56.1] |
| MCR16 + Xwn | Ppr_w2w | 55.8 | 63.2 | 46.2 | 57.5 | **92.9** | [53.7, 57.7] |
| MCR16 + Xwn | Static | 53.7 | 59.5 | 45.0 | 57.8 | **92.9** | [51.8, 55.7] |
| WNet17 + Xwn | Ppr | 56.1 | 62.6 | 46.0 | 60.8 | **92.9** | [54.0, 58.1] |
| WNet17 + Xwn | Ppr_w2w | **57.4** | **64.1** | 46.9 | **62.6** | **92.9** | [55.5, 59.3] |
| WNet17 + Xwn | Spr | 56.20 | 61.6 | **47.3** | 61.8 | **92.9** | [54.8, 58.2] |
| WNet30 + gloss | Ppr | 48.5 | 52.2 | 41.5 | 54.2 | 78.6 | [46.7, 50.6] |
| WNet30 + gloss | Ppr_w2w | 51.6 | 59.0 | 40.2 | 57.2 | 78.6 | [49.9, 53.3] |
| WNet30 + gloss | Spr | 45.4 | 54.1 | 31.4 | 52.5 | 78.6 | [43.7, 47.4] |
| MFS | | 62.3 | 69.3 | 53.6 | 63.7 | 92.9 | [60.2, 64.0] |
| GAMBL | | 65.2 | 70.8 | 59.3 | 65.3 | 100 | |

Table 1: Results (as recall) on Senseval-2 and Senseval-3 all words tasks. We also include the MFS baseline and the best results of supervised systems at competition time (SMUaw,GAMBL).

rithm, compared to the more elaborate algorithm to construct the graph. The differences between methods are not statistically significant, which is a common problem on this relatively small datasets (Snyder and Palmer, 2004; Palmer et al., 2001).

Regarding LKBs, the best results are obtained using WordNet 1.7 and eXtended WordNet. Here the differences are in many cases significant. These results are surprising, as we would expect that the manually disambiguated gloss relations from WordNet 3.0 would lead to better results, compared to the automatically disambiguated gloss relations from the eXtended WordNet (linked to version 1.7). The lower performance of WNet30+gloss can be due to the fact that the Senseval all words data set is tagged using WordNet 1.7 synsets. When using a different LKB for WSD, a mapping to WordNet 1.7 is required. Although the mapping is cited as having a correctness on the high 90s (Daude et al., 2000), it could have introduced sufficient noise to counteract the benefits of the hand-disambiguated glosses.

Table 1 also shows the most frequent sense (MFS), as well as the best supervised systems (Snyder and Palmer, 2004; Palmer et al., 2001) that participated in each competition (SMUaw and GAMBL, respectively). The MFS is a baseline for supervised systems, but it is considered a difficult competitor for unsupervised systems, which rarely come close to it. In this case the MFS baseline was computed using previously availabel training data like SemCor. Our best results are close to the MFS in both Senseval-2 and Senseval-3 datasets. The results for the supervised system are given for reference, and we can see that the gap is relatively small, specially for Senseval-3.

## 5 Comparison to Related work

In this section we will briefly describe some graph-based methods for knowledge-based WSD. The methods here presented cope with the problem of sequence-labeling, i.e., they disambiguate all the words coocurring in a sequence (typically, all content words of a sentence). All the methods rely on the information represented on some LKB, which typically is some version of WordNet, sometimes enriched with proprietary relations. The results on our datasets, when available, are shown in Table 2. The table also shows the performance of supervised systems.

The TexRank algorithm (Mihalcea, 2005) for WSD creates a complete weighted graph (e.g. a graph where every pair of distinct vertices is connected by a weighted edge) formed by the synsets of the words in the input context. The weight

| Senseval-2 All Words dataset | | | | | |
|---|---|---|---|---|---|
| System | All | N | V | Adj. | Adv. |
| Mih05 | 54.2 | 57.5 | 36.5 | 56.7 | 70.9 |
| Sihna07 | 56.4 | 65.6 | 32.3 | **61.4** | 60.2 |
| Tsatsa07 | 49.2 | – | – | – | – |
| Spr | 56.6 | 66.7 | 37.5 | 57.6 | **70.8** |
| Ppr | 56.8 | **71.1** | 33.4 | 55.9 | 67.1 |
| Ppr_w2w | **58.6** | 70.4 | **38.9** | 58.3 | 70.1 |
| MFS | 60.1 | 71.2 | 39.0 | 61.1 | 75.4 |

| Senseval-3 All Words dataset | | | | | |
|---|---|---|---|---|---|
| System | All | N | V | Adj. | Adv. |
| Mih05 | 52.2 | - | - | - | - |
| Sihna07 | 52.4 | 60.5 | 40.6 | 54.1 | **100.0** |
| Nav07 | - | 61.9 | 36.1 | **62.8** | - |
| Spr | 56.2 | 61.6 | **47.3** | 61.8 | 92.9 |
| Ppr | 56.1 | 62.6 | 46.0 | 60.8 | 92.9 |
| Ppr_w2w | **57.4** | **64.1** | 46.9 | 62.6 | 92.9 |
| MFS | 62.3 | 69.3 | 53.6 | 63.7 | 92.9 |
| Nav05 | 60.4 | - | - | - | - |

Table 2: Comparison with related work. Note that Nav05 uses the MFS.

of the links joining two synsets is calculated by executing Lesk's algorithm (Lesk, 1986) between them, i.e., by calculating the overlap between the words in the glosses of the correspongind senses. Once the complete graph is built, the PageRank algorithm is executed over it and words are assigned to the most relevant synset. In this sense, PageRank is used an alternative to simulated annealing to find the optimal pairwise combinations. The method was evaluated on the Senseval-3 dataset, as shown in row Mih05 on Table 2.

(Sinha and Mihalcea, 2007) extends their previous work by using a collection of semantic similarity measures when assigning a weight to the links across synsets. They also compare different graph-based centrality algorithms to rank the vertices of the complete graph. They use different similarity metrics for different POS types and a voting scheme among the centrality algorithm ranks. Here, the Senseval-3 corpus was used as a development data set, and we can thus see those results as the upper-bound of their method.

We can see in Table 2 that the methods presented in this paper clearly outperform both Mih05 and Sin07. This result suggests that analyzing the LKB structure as a whole is preferable than computing pairwise similarity measures over synsets. The results of various in-house made experiments replicating (Mihalcea, 2005) also confirm this observation. Note also that our methods are simpler than the combination strategy used in (Sinha and Mihalcea, 2007), and that we did not perform any parameter tuning as they did.

In (Navigli and Velardi, 2005) the authors develop a knowledge-based WSD method based on lexical chains called structural semantic interconnections (SSI). Although the system was first designed to find the meaning of the words in WordNet glosses, the authors also apply the method for labeling text sequences. Given a text sequence, SSI first identifies monosemous words and assigns the corresponding synset to them. Then, it iteratively disambiguates the rest of terms by selecting the senses that get the strongest interconnection with the synsets selected so far. The interconnection is calculated by searching for paths on the LKB, constrained by some hand-made rules of possible semantic patterns. The method was evaluated on the Senseval-3 dataset, as shown in row Nav05 on Table 2. Note that the method labels an instance with the most frequent sense of the word if the algorithm produces no output for that instance, which makes comparison to our system unfair, specially given the fact that the MFS performs better than SSI. In fact it is not possible to separate the effect of SSI from that of the MFS. For this reason we place this method close to the MFS baseline in Table 2.

In (Navigli and Lapata, 2007), the authors perform a two-stage process for WSD. Given an input context, the method first explores the whole LKB in order to find a subgraph which is particularly relevant for the words of the context. Then, they study different graph-based centrality algorithms for deciding the relevance of the nodes on the subgraph. As a result, every word of the context is attached to the highest ranking concept among its possible senses. The *Spr* method is very similar to (Navigli and Lapata, 2007), the main difference lying on the initial method for extracting the context subgraph. Whereas (Navigli and Lapata, 2007) apply a depth-first search algorithm over the LKB graph —and restrict the depth of the subtree to a value of 3—, *Spr* relies on shortest paths between word synsets. Navigli and Lapata don't report overall results and therefore, we can't directly compare our results with theirs. However, we can see that on a PoS-basis evaluation our results are consistently better for nouns and verbs (especially the *Ppr_w2w* method) and rather similar for adjectives.

(Tsatsaronis et al., 2007) is another example of a two-stage process, the first one consisting on finding a relevant subgraph by performing a BFS

| Spanish Semeval07 | | |
|---|---|---|
| LKB | Method | Acc. |
| Spanish Wnet + Xnet* | Ppr | 78.4 |
| Spanish Wnet + Xnet* | Ppr_w2w | 79.3 |
| – | MFS | 84.6 |
| – | Supervised | 85.10 |

Table 3: Results (accuracy) on Spanish Semeval07 dataset, including MFS and the best supervised system in the competition.

search over the LKB. The authors apply a spreading activation algorithm over the subgraph for node ranking. Edges of the subgraph are weighted according to its type, following a tf.idf like approach. The results show that our methods clearly outperform Tsatsa07. The fact that the *Spr* method works better suggests that the traditional PageRank algorithm is a superior method for ranking the subgraph nodes.

As stated before, all methods presented here use some LKB for performing WSD. (Mihalcea, 2005) and (Sinha and Mihalcea, 2007) use WordNet relations as a knowledge source, but neither of them specify which particular version did they use. (Tsatsaronis et al., 2007) uses WordNet 1.7 enriched with eXtended WordNet relations, just as we do. Both (Navigli and Velardi, 2005; Navigli and Lapata, 2007) use WordNet 2.0 as the underlying LKB, albeit enriched with several new relations, which are manually created. Unfortunately, those manual relations are not publicly available, so we can't directly compare their results with the rest of the methods. In (Agirre and Soroa, 2008) we experiment with different LKBs formed by combining relations of different MCR versions along with relations extracted from SemCor, which we call supervised and unsupervised relations, respectively. The unsupervised relations that yielded bests results are also used in this paper (c.f Section 3.1).

## 6 Experiments on Spanish

Our WSD algorithm can be applied over non-english texts, provided that a LKB for this particular language exists. We have tested the graph-algorithms proposed in this paper on a Spanish dataset, using the Spanish WordNet as knowledge source (Atserias et al., 2004a).

We used the Semeval-2007 Task 09 dataset as evaluation gold standard (Màrquez et al., 2007). The dataset contains examples of the 150 most frequent nouns in the CESS-ECE corpus, manually annotated with Spanish WordNet synsets. It is split into a train and test part, and has an "all words" shape i.e. input consists on sentences, each one having at least one occurrence of a target noun. We ran the experiment over the test part (792 instances), and used the train part for calculating the MFS baseline. We used the Spanish WordNet as LKB, enriched with eXtended WordNet relations. It contains $105,501$ nodes and $623,316$ relations. The results in Table 3 are consistent with those for English, with our algorithm approaching MFS performance. Note that for this dataset the supervised algorithm could barely improve over the MFS, suggesting that for this particular dataset MFS is particularly strong.

| Method | Time |
|---|---|
| Ppr | 26m46 |
| Spr | 119m7 |
| Ppr_w2w | 164m4 |

Table 4: Elapsed time (in minutes) of the algorithms when applied to the Senseval-2 dataset.

## 7 Performance analysis

Table 4 shows the time spent by the different algorithms when applied to the Senseval-2 all words dataset, using the WNet17 + Xwn as LKB. The dataset consists on 2473 word instances appearing on 476 different sentences. The experiments were done on a computer with four 2.66 Ghz processors and 16 Gb memory. The table shows that the time elapsed by the algorithms varies between 30 minutes for the *Ppr* method (which thus disambiguates circa 82 instances per minute) to almost 3 hours spent by the *Ppr_w2w* method (circa 15 instances per minute). The *Spr* method lies in between, requiring 2 hours for completing the task, but its overall performance is well below the PageRank based *Ppr_w2w* method. Note that the algorithm is coded in C++ for greater efficiency, and uses the Boost Graph Library.

Regarding PageRank calculation, we have tried different numbers of iterations, and analyze the rate of convergence of the algorithm. Figure 1 depicts the performance of the *Ppr_w2w* method for different iterations of the algorithm. As before, the algorithm is applied over the MCR17 + Xwn LKB, and evaluated on the Senseval-2 all words dataset. The algorithm converges very quickly: one sole iteration suffices for achieving a relatively high per-

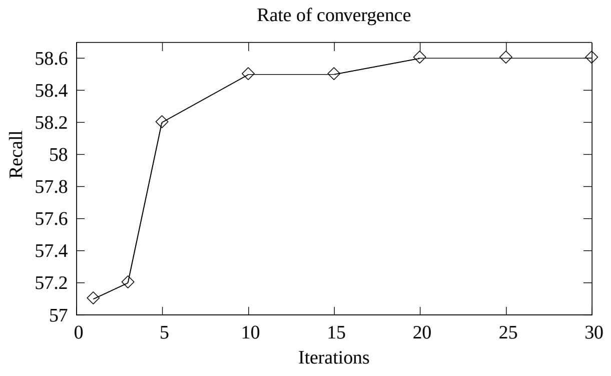

Figure 1: Rate of convergence of PageRank algorithm over the MCR17 + Xwn LKB.

formance, and 20 iterations are enough for achieving convergence. The figure shows that, depending on the LKB complexity, the user can tune the algorithm and lower the number of iterations, thus considerably reducing the time required for disambiguation.

## 8 Conclusions

In this paper we propose a new graph-based method that uses the knowledge in a LKB (based on WordNet) in order to perform unsupervised Word Sense Disambuation. Our algorithm uses the full graph of the LKB efficiently, performing better than previous approaches in English all-words datasets. We also show that the algorithm can be easily ported to other languages with good results, with the only requirement of having a wordnet. Both for Spanish and English the algorithm attains performances close to the MFS.

The algorithm is publicly available[5] and can be applied easily to sense inventories and knowledge bases different from WordNet. Our analysis shows that our algorithm is efficient compared to previously proposed alternatives, and that a good choice of WordNet versions and relations is fundamental for good performance.

## Acknowledgments

This work has been partially funded by the EU Commission (project KYOTO ICT-2007-211423) and Spanish Research Department (project KNOW TIN2006-15049-C03-01).

## References

E. Agirre and G. Rigau. 1996. Word sense disambiguation using conceptual density. In *In Proceedings of the 16th International Conference on Computational Linguistics*, pages 16–22.

E. Agirre and A. Soroa. 2008. Using the multilingual central repository for graph-based word sense disambiguation. In *Proceedings of LREC '08*, Marrakesh, Morocco.

J. Atserias, G. Rigau, and L. Villarejo. 2004a. Spanish wordnet 1.6: Porting the spanish wordnet across princeton versions. In *In Proceedings of LREC '04*.

J. Atserias, L. Villarejo, G. Rigau, E. Agirre, J. Carroll, B. Magnini, and P. Vossen. 2004b. The meaning multilingual central repository. In *In Proceedings of GWC*, Brno, Czech Republic.

S. Brin and L. Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7).

J. Cowie, J. Guthrie, and L. Guthrie. 1992. Lexical disambiguation using simulated annealing. In *HLT '91: Proceedings of the workshop on Speech and Natural Language*, pages 238–242, Morristown, NJ, USA.

J. Daude, L. Padro, and G. Rigau. 2000. Mapping WordNets using structural information. In *Proceedings of ACL'2000*, Hong Kong.

T. H. Haveliwala. 2002. Topic-sensitive pagerank. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 517–526, New York, NY, USA. ACM.

A. N. Langville and C. D. Meyer. 2003. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380.

M. Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, New York, NY, USA. ACM.

L. Màrquez, L. Villarejo, M. A. Martí, and M. Taulé. 2007. Semeval-2007 task 09: Multilevel semantic annotation of catalan and spanish. In *Proceedings of SemEval-2007*, pages 42–47, Prague, Czech Republic, June.

D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant word senses in untagged text. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 279, Morristown, NJ, USA. Association for Computational Linguistics.

R. Mihalcea and D. I. Moldovan. 2001. eXtended WordNet: Progress report. In *in Proceedings of NAACL Workshop on WordNet and Other Lexical Resources*, pages 95–100.

R. Mihalcea. 2005. Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of HLT05*, Morristown, NJ, USA.

[5] `http://ixa2.si.ehu.es/ukb`

R. Navigli and M. Lapata. 2007. Graph connectivity measures for unsupervised word sense disambiguation. In *IJCAI*.

R. Navigli and P. Velardi. 2005. Structural semantic interconnections: A knowledge-based approach to word sense disambiguation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(7):1075–1086.

M. Palmer, C. Fellbaum, S. Cotton, L. Delfs, and H.T. Dang. 2001. English tasks: All-words and verb lexical sample. In *Proc. of SENSEVAL-2: Second International Workshop on Evaluating Word Sense Disambiguation Systems*, Tolouse, France, July.

S. Pradhan, E. Loper, D. Dligach, and M.Palmer. 2007. Semeval-2007 task-17: English lexical sample srl and all words. In *Proceedings of SemEval-2007*, pages 87–92, Prague, Czech Republic, June.

R. Sinha and R. Mihalcea. 2007. Unsupervised graph-based word sense disambiguation using measures of word semantic similarity. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2007)*, Irvine, CA, USA.

B. Snyder and M. Palmer. 2004. The English all-words task. In *ACL 2004 Senseval-3 Workshop*, Barcelona, Spain, July.

G. Tsatsaronis, M. Vazirgiannis, and I. Androutsopoulos. 2007. Word sense disambiguation with spreading activation networks generated from thesauri. In *IJCAI*.