An Iterative 'Sudoku Style' Approach to Subgraph-based Word Sense Disambiguation

Steve L. Manion
University of Canterbury
Christchurch, New Zealand
`steve.manion`
`@pg.canterbury.ac.nz`

Raazesh Sainudiin
University of Canterbury
Christchurch, New Zealand
`r.sainudiin`
`@math.canterbury.ac.nz`

Abstract

We introduce an *iterative* approach to subgraph-based Word Sense Disambiguation (WSD). Inspired by the Sudoku puzzle, it significantly improves the *precision* and *recall* of disambiguation. We describe how *conventional* subgraph-based WSD treats the two steps of (1) subgraph construction and (2) disambiguation via graph centrality measures as ordered and atomic. Consequently, researchers tend to focus on improving either of these two steps individually, overlooking the fact that these steps can complement each other if they are allowed to interact in an iterative manner. We tested our iterative approach against the conventional approach for a range of well-known graph centrality measures and subgraph types, at the sentence and document level. The results demonstrated that an average performing WSD system which embraces the iterative approach, can easily compete with state-of-the-art. This alone warrants further investigation.

1 Introduction

Explicit WSD is a two-step process of analysing a word's contextual use then deducing its intended sense. When Kilgarriff (1998) established SENSEVAL, the collaborative framework and forum to evaluate WSD, unsupervised systems performed poorly in comparison to their supervised counterparts (Palmer et al., 2001; Snyder and Palmer, 2004). A review of the literature shows there

has been a healthy *rivalry* between the two, in which proponents of unsupervised WSD have long sought to vindicate its potential since two decades ago (Yarowsky, 1995) to even more recent times (Ponzetto and Navigli, 2010).

As Pedersen (2007) rightly states, supervised systems are bound by their training data, and therefore are limited in portability and flexibility in the face of new domains, changing applications, or different languages. This *knowledge acquisition bottleneck*, coined by Gale et al. (1992), can be alleviated by unsupervised systems that exploit the portability and flexibility of Lexical Knowledge Bases (LKBs). As of 2007, SENSEVAL became SEMEVAL, offering a more diverse range of semantic tasks. Unsupervised knowledge-based WSD has since had its performance evaluated in terms of *granularity* (Navigli et al., 2007), *domain* (Agirre et al., 2010), and *cross/multi-linguality* (Lefever and Hoste, 2010; Lefever and Hoste, 2013; Navigli et al., 2013). Results from these tasks have demonstrated unsupervised systems are now a competitive and robust alternative to supervised systems, especially given the ever changing task-orientated settings WSD is evaluated in.

One such class of unsupervised knowledge-based WSD systems that we seek to improve in this paper constructs semantic subgraphs from LKBs, and then runs graph-based centrality measures such as PageRank (Brin and Page, 1998) over them to finally select the senses (as nodes) ranked as the most relevant. This class is known as *subgraph-based* WSD, characterised over the last decade by performing the two key steps of (1) subgraph construction and (2) disambiguation via graph centrality measures, in an ordered atomic sequence. We refer to this characteristic as the *conventional* approach to subgraph-based WSD. We propose an *iterative* approach to subgraph-based WSD that allows for interaction between the two major steps in an incremental manner

and demonstrate its effectiveness across a range of graph-based centrality measures and subgraph construction methods at the sentence and document levels of disambiguation.

2 The Conventional Subgraph Approach

The *conventional* approach to subgraph WSD firstly benefits from some preprocessing, in which words in a sequence $\mathcal{W}$, are mapped to their lemmatisations[1] in a set $\mathcal{L}$, such that $(w_1, ..., w_m) \mapsto \{\ell_1, ..., \ell_m\}$. This facilitates better lexical alignment with the LKB to be exploited. Let this LKB be a large semantic graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, such that $\mathcal{S}$ is a set of vertices representing all known word senses, and $\mathcal{E}$ be a set of edges defining semantic relationships that exist between senses. Now given we wish to disambiguate $\ell_i \in \mathcal{L}$, let $R(\ell_i)$ be a function that *Retrieves* from $\mathcal{G}$, all the senses, $\{s_{i,1}, s_{i,2}, ..., s_{i,k}\}$, that ℓ_i could refer to, noting that i is an anchor to the original word w_i.

2.1 Step 1: Subgraph Construction

For unsupervised subgraph-based WSD, the key publications that have advanced the field broadly construct subgraph, $\mathcal{G}_\mathcal{L}$, as either a union of *subtree paths*, *shortest paths*, or *local edges*[2]. First we initialise $\mathcal{G}_\mathcal{L}$, by setting $\mathcal{S}_\mathcal{L} := \bigcup_{i=1}^n R(\ell_i)$ and $\mathcal{E}_\mathcal{L} := \emptyset$. Next we add edges to $\mathcal{E}_\mathcal{L}$, depending on the desired subgraph type, by adding either the:

(a) *Subtree paths* of up to length L, via a Depth-First Search (DFS) of $\mathcal{G}$. In brief, **for each** sense $s_a \in \mathcal{S}_\mathcal{L}$, **if** a new sense $s_b \in \mathcal{S}_\mathcal{L}$, i.e. $s_b \neq s_a$, is encountered along a path $P_{a \to b} = \{\{s_a, s\}, ..., \{s', s_b\}\}$ with path-length $|P_{a \to b}| \leq L$, **then** add $P_{a \to b}$ to $\mathcal{G}_\mathcal{L}$. [cf. Navigli and Velardi (2005), Navigli and Lapata (2007), or Navigli and Lapata (2010)]

(b) *Shortest paths*, via a Breadth-First Search (BFS) of $\mathcal{G}$. In brief, **for each** sense pair $s_a, s_b \in \mathcal{S}_\mathcal{L}$, find the shortest path $P_{a \to b} = \{\{s_a, s\}, ..., \{s', s_b\}\}$; **if** such a path $P_{a \to b}$ exists and (optionally) $|P_{a \to b}| \leq L$, **then** add $P_{a \to b}$ to $\mathcal{G}_\mathcal{L}$ [cf. Agirre and Soroa (2008), Agirre and Soroa (2009), or Gutiérrez et al. (2013)]

(c) *Local edges* up to a local distance D. In brief, **for each** sense pair $s_a, s_b \in \mathcal{S}_\mathcal{L}$, **if** the distance in the text $|b - a|$ between the corresponding words w_a and w_b satisfies $|b - a| \leq D$, **then** add edge $\{s_a, s_b\}$ to $\mathcal{G}_\mathcal{L}$ (preferably with edge-weights). [cf. Mihalcea (2005) or Sinha and Mihalcea (2007)] (Note that this subgraph is a hybrid, because only its vertices belong to $\mathcal{G}$)

In practice, subgraph edges may be *directed*, *weighted*, *collapsed*, or *filtered*. However to keep the distinctions between subgraph types simple, we do not include this in our formalisation.

2.2 Step 2: Disambiguation

To disambiguate each lemma $\ell_i \in \mathcal{L}$, its corresponding senses, $R(\ell_i) = \{s_{i,1}, s_{i,2}, ..., s_{i,k}\}$, are scored by a graph-based centrality measure ϕ, over subgraph $\mathcal{G}_\mathcal{L}$, to estimate the most appropriate sense, $\hat{s}_{i,*} = \arg\max_{s_{i,j} \in R(\ell_i)} \phi(s_{i,j})$. The estimated sense $\hat{s}_{i,*}$ is then assigned to word w_i.

2.3 Algorithm for Conventional Approach

With both steps formalised, we can now illustrate the conventional subgraph approach in Algorithm 1. Let $\mathcal{L}$ be taken as *input*, and let the disambiguation results $\mathcal{D} = \{\hat{s}_{1,*}, ..., \hat{s}_{m,*}\}$ be produced as *output* to assign to $\mathcal{W} = (w_1, ..., w_m)$.

Algorithm 1: Conventional Approach

Input: $\mathcal{L}$
Output: $\mathcal{D}$
$\mathcal{D} \leftarrow \emptyset$;
$\mathcal{G}_\mathcal{L} \leftarrow \texttt{ConstructSubGraph}\,(\mathcal{L})$;
foreach $\ell_i \in \mathcal{L}$ **do**
    $\hat{s}_{i,*} \leftarrow \arg\max_{s_{i,j} \in R(\ell_i)} \phi(s_{i,j})$;
    put $\hat{s}_{i,*}$ in $\mathcal{D}$;

To begin with, $\mathcal{D}$ is initialised as an empty set and $\texttt{ConstructSubGraph}(\mathcal{L})$ constructs one of the three subgraphs described in section 2.1. Next for each $\ell_i \in \mathcal{L}$, by running a graph based centrality measure ϕ over $\mathcal{G}_\mathcal{L}$, the most appropriate sense $\hat{s}_{i,*}$ is estimated, and placed in set $\mathcal{D}$. Effectively, $\mathcal{L}$ is a context window based on document or sentence size, therefore this algorithm is run for each context window division. Note that Algorithm 1 would require a little extra complexity to handle local edge subgraphs, due to its context window needing to satisfy $\mathcal{L} = \{\ell_{i-D}, ..., \ell_{i+D}\}$.

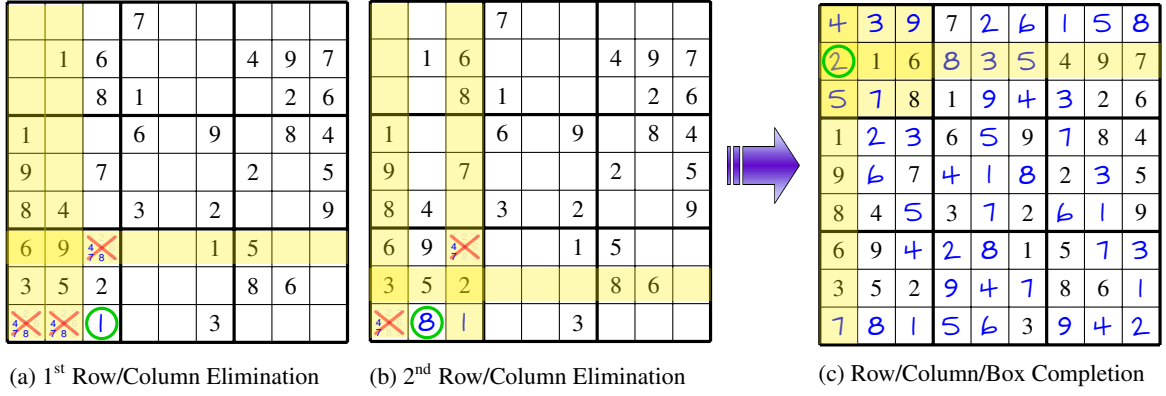---

[1] For a detailed explanation of the processes leading up to lemmatisation (and beyond), *see* Navigli (2009, p12)

[2] 'Local' describes the *local context*, typically this is the 2 or 3 words either side of a word, *see* Yarowsky (1993)

(a) 1ˢᵗ Row/Column Elimination   (b) 2ⁿᵈ Row/Column Elimination   (c) Row/Column/Box Completion

Figure 1: Iterative Solving of Sudoku Grids

3 The Iterative Subgraph Approach

3.1 What is Iterative WSD?

The key observation to make about the conventional approach in Algorithm 1, is for input $\mathcal{L}$, constructing subgraph $\mathcal{G}_\mathcal{L}$ and performing disambiguation are two ordered atomic steps. Notice that there is no iteration between them, because the first step of subgraph construction is never revisited for each $\mathcal{L}$. For the conventional process to be iterative, then for $\ell_a, \ell_b \in \mathcal{L}$ a previous disambiguation of ℓ_a, would need to influence a consecutive disambiguation of ℓ_b, through an iterative re-construction of $\mathcal{G}_\mathcal{L}$ between each disambiguation. This key difference illustrated by Figure 2, is the level of iterative WSD we aspire to.

(a) Conventional Approach

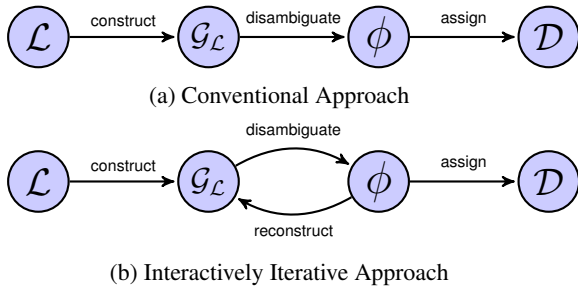(b) Interactively Iterative Approach

Figure 2: The Key Difference In Approach

It is important to note, the term *iterative* can already be found in WSD literature, therefore we take the opportunity here to make a distinction. Firstly, a graph based centrality measure ϕ may be iterative, such as PageRank (Brin and Page, 1998) or Hyperlink-Induced Topic Search (HITS) (Kleinberg, 1999). In the experiments by Mihalcea (2005) in which PageRank was run over *local edge* subgraphs (as described in 2.1 (c)), it is easy to perceive the WSD process itself as iterative.

Iteration can again be taken further, as observed with Personalised PageRank in which Agirre and Soroa (2009) apply the idea of biasing values in the random surfing vector, v, (*see* (Haveliwala, 2003)). For their run labelled "Ppr_w2w", in order to avoid senses anchored to the same lemma assisting each other's ϕ score, the random surfing vector v is iteratively updated as ℓ_i changes, to ensure context senses $s_{a,j} \in v$ such that $a \neq i$ are the only senses that receive probability mass.
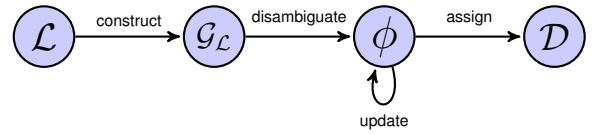

Figure 3: Atomically Iterative Approach

In summary, iteration in the literature either describes ϕ as being iterative or being iteratively adjusted, both of which are contained in the disambiguation step alone as shown in Figure 3. This is iteration at the atomic level and should not be conflated with the interactive level of iteration that we propose as seen in Figure 2 (b).

3.2 Iteratively Solving a Sudoku Grid

In Figures 1 (a), (b), and (c), we observe the solving of a Sudoku puzzle, in which the numbers from 1 to 9 must be assigned only once to each *column*, *row*, and 3x3 *square*. Each time a number is assigned and the Sudoku grid is updated, this is an *iteration*. For example, in the south west square of grid (a) (i.e. Figure 1 (a)) unknown cells can be assigned $\{1, 4, 7, 8\}$. Given that 1 has already been assigned to the 7ᵗʰ row and the 1ˢᵗ and 2ⁿᵈ columns, this singles it down to one cell it can be assigned to. The iteration of grid

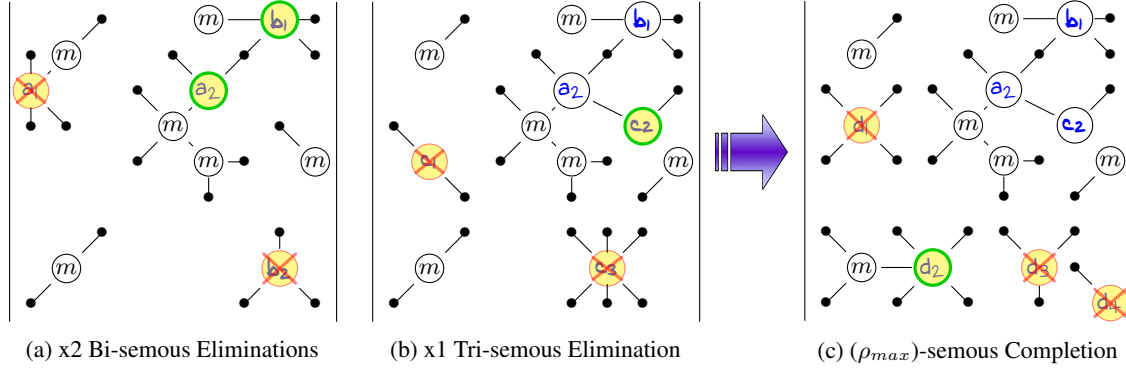(a) x2 Bi-semous Eliminations   (b) x1 Tri-semous Elimination   (c) (ρ_{max})-semous Completion

Figure 4: Iterative Disambiguating of Subgraphs

(a), now makes possible the iteration of grid (b) to eliminate the number 8 as the only possibility for its assigned cell. This iterative process continues until we reach the completed puzzle in grid (c). Therefore in WSD terminology, with each cell we *disambiguate*, a new grid is *constructed*, in which knowledge is passed on to each consecutive iteration.

Continuing with this line of thought, each unsolved cell is *ambiguous*, with a degree of *polysemy* ρ, such that $\rho_{max} \leq 9$. Again, the initial Sudoku grid has pre-solved cells, of which are *monosemous*. This brings us to another key observation. Typically in Sudoku, it is necessary to solve the least polysemous cells first, before you can solve the more polysemous cells with a certainty. As the conventional approach exhibits no Sudoku-like iteration, cells are solved without regard to the ρ value of the cell, or any interactive exploitation of previously solved cells.

3.3 Iteratively Constructing a Subgraph

In our 'Sudoku style' approach, we propose disambiguating each ℓ_i in order of increasing polysemy ρ, iteratively reconstructing subgraph $\mathcal{G}_\mathcal{L}$ to reflect 1) previous disambiguations and 2) the ρ value of lemmas being disambiguated in the current iteration. This is illustrated in Figures 4 (a), (b), and (c) above.

Let m-labelled vertices describe monosemous lemmas. In graph (a) (i.e. Figure 4) we observe two bi-semous lemmas, a and b, in which our arbitrary graph-based centrality measure ϕ has selected the second sense of a (i.e. a_2) and the first sense of b (i.e. b_1) to be placed in $\mathcal{D}$. For the next iteration, you will notice the alternative senses for a and b are removed from $\mathcal{G}_\mathcal{L}$ for the disambiguation of tri-semous lemma c. The second sense of

lemma c manages to be selected by ϕ with the help of the previous disambiguation of lemma a. This interactive and iterative process continues until we reach the most polysemous lemma, which in our example is d with $\rho_{max} = 4$ in graph (c).

3.4 Algorithm for Iterative Approach

We can formally describe what is happening in Figure 4 with Algorithm 2. Effectively, this is a recreation of Algorithm 1, which highlights the differences in the conventional and iterative approach.

Algorithm 2: Iterative Approach

Input: $\mathcal{L}$
Output: $\mathcal{D}$
$\mathcal{D} \leftarrow$ GetMonosemous $(\mathcal{L})$;
$\mathcal{A} \leftarrow \emptyset$;
for $\rho \leftarrow 2$ **to** ρ_{max} **do**
    $\mathcal{A} \leftarrow$ AddPolysemous $(\mathcal{L}, \rho)$;
    $\mathcal{G}_\mathcal{L} \leftarrow$ ConstructSubGraph $(\mathcal{A},\mathcal{D})$;
    foreach $\ell_i \in \mathcal{A}$ **do**
        $\hat{s}_{i,*} \leftarrow \arg\max_{s_{i,j} \in R(\ell_i)} \phi(s_{i,j})$;
        if $\hat{s}_{i,*}$ *exists* **then**
            remove ℓ_i from $\mathcal{A}$;
            put $\hat{s}_{i,*}$ in $\mathcal{D}$;

Firstly, as it reads GetMonosemous$(\mathcal{L})$ places all the senses of the monosemous lemmas into the set of *disambiguated* lemmas $\mathcal{D}$. This is the equivalent of copying out an unsolved Sudoku grid onto a piece of paper and adding in all the initial hint numbers. Next the set $\mathcal{A}$ which holds all *ambiguous* lemmas of polysemy $\leq \rho$ is initialised as an empty set. Now we are ready to iterate through values of ρ, beginning from the first iteration, by adding all bi-semous lemmas to

$\mathcal{A}$ with the function AddPolysemous($\mathcal{L}, \rho$), notice ρ places a restriction on the degree of polysemy a lemma $\ell_i \in \mathcal{L}$ can have before being added to $\mathcal{A}$.

We are now ready to create the first subgraph $\mathcal{G}_\mathcal{L}$ with function ConstructSubGraph($\mathcal{A}, \mathcal{D}$). This previously used function in Algorithm 1, is now modified to take the ambiguous lemmas of polysemy $\leq \rho$ in set $\mathcal{A}$ and previously disambiguated lemma senses in set $\mathcal{D}$. The resulting graph has a limited degree of polysemy and is constructed based on previous disambiguations.

From this point on the given graph centrality measure ϕ is run over $\mathcal{G}_\mathcal{L}$. For the lemmas that are disambiguated, they are removed from $\mathcal{A}$ and the selected sense is added to $\mathcal{D}$. For those lemmas that are not (i.e. $\hat{s}_{i,*}$ does not exist[3]) they remain in $\mathcal{A}$ to be involved in reattempted disambiguations in consecutive iterations. As more lemmas are disambiguated, it is more likely that previously difficult to disambiguate lemmas become much easier to solve, just like at the end of a Sudoku puzzle it gets easier as you get closer to completing it.

4 Evaluations

In our evaluations we set out to understand a number of aspects. The first evaluation is a *proof of concept*, to understand whether an iterative approach to subgraph WSD can in fact achieve better performance than the conventional approach. The second set of experiments seeks to understand how the iterative approach works and the performance *benefits* and *penalties* of implementing the iterative approach. Finally the third experiment is an *elementary attempt* at optimising the iterative approach to defeat the MFS baseline.

4.1 LKB & Dataset

For an evaluation, we have chosen the multilingual LKB known as BabelNet (Navigli and Ponzetto, 2012a). It weaves together several other LKBs, most notably WordNet (Fellbaum, 1998) and Wikipedia. It also can be easily accessed with the BabelNet API, of which we have built our code base around. All experiments are conducted on the most recent SemEval WSD dataset, of which is the SemEval 2013 Task 12 Multilingual WSD (English) data set.

---

[3]This can happen if ℓ_i does not map to any senses, or alternatively all the senses that are mapped to are filtered out of the subgraph before disambiguation (explained later).

4.2 Graph Centrality Measures Evaluated

To demonstrate the effectiveness of our iterative approach, we selected a range of WSD graph-based centrality measures often experimented with in the literature. Firstly ϕ does not need to be a complicated measure, this is demonstrated by the success of ranking senses by their number of incoming and outgoing edges. Even though it is very simple, it performs surprisingly well against others for both In-Degree (Navigli and Lapata, 2007) and Out-Degree (Navigli and Ponzetto, 2012a)

Next we employ graph centrality measures that are primarily used to disambiguate the *semantic web*, such as PageRank (Brin and Page, 1998), HITS Kleinberg (1999), and a *personalised* PageRank (Haveliwala, 2003); which have since been applied to WSD by Mihalcea (2005), Navigli and Lapata (2007), and Agirre and Soroa (2009) respectively. We also include Betweeness Centrality (Freeman, 1979) which is taken from the analysis of social networks.

These methods are well known and applied across many disciplines, therefore we will leave it to the reader to follow up on the specifics of these graph centrality measures. However we do explicitly define our last measure, Sum Inverse Path Length (Navigli and Ponzetto, 2012a; Navigli and Ponzetto, 2012b) in Equation (1) which was designed with WSD in mind, thus is less well known.

$$\phi(s) = \sum_{p \in P_{s \to c}} \frac{1}{e^{|p|-1}} \qquad (1)$$

This measure scores a sense by summing up the scores of all paths that connect to other senses in $\mathcal{G}_\mathcal{L}$ (i.e. senses that are not intermediate nodes, but have a mapping back to a lemma in the context window $\mathcal{L}$). In the words of Navigli and Ponzetto (2012a), $P_{s \to c}$ is the set of paths connecting s to other senses of context words, with $|p|$ as the number of edges in the path p and each path is scored with the exponential inverse decay of the path length.

4.3 Experiment 1: Proof of Concept
4.3.1 Experiment 1: Setup
For this experiment we simply set out to see how the iterative approach performed compared to the conventional approach in a range of experimental conditions. Directed and unweighted subgraphs were used, namely subtree paths and shortest paths subgraphs with $L = 2$. To address the issue of

$\mathcal{G_L}$	ϕ	Conventional Doc			Iterative Doc			Improvement		
		P	R	F	P	R	F	ΔP	ΔR	ΔF
SubTree Paths	In-Degree	**61.70**	**55.51**	**58.44**	**65.39**	**63.74**	**64.55**	+3.69	+8.23	+6.11
	Out-Degree	54.23	48.78	51.36	57.70	56.23	56.96	+3.47	+7.45	+5.59
	Betweenness Centrality	59.29	53.34	56.15	63.43	61.82	62.61	+4.14	+8.48	+6.46
	Sum Inverse Path Length	56.58	50.90	53.59	58.86	57.37	58.11	+2.28	+6.47	+4.51
	HITS(hub)	54.69	49.20	51.80	59.71	58.20	58.95	**+5.03**	**+9.00**	**+7.15**
	HITS(authority)	57.45	51.68	54.41	61.62	60.06	60.83	+4.18	+8.38	+6.42
	PageRank	60.09	54.06	56.92	64.07	62.44	63.24	+3.97	+8.38	+6.33
Shortest Paths	In-Degree	**63.06**	**56.08**	**59.36**	65.36	63.06	64.19	+2.30	+6.98	+4.83
	Out-Degree	57.07	50.75	53.72	61.14	58.92	60.01	+4.07	+8.17	+6.29
	Betweenness Centrality	60.33	53.65	56.79	**65.52**	**63.22**	**64.35**	**+5.20**	**+9.57**	**+7.56**
	Sum Inverse Path Length	57.53	51.16	54.16	61.19	58.98	60.06	+3.66	+7.81	+5.90
	HITS(hub)	57.48	51.11	54.11	62.14	59.96	61.03	+4.67	+8.85	+6.92
	HITS(authority)	60.91	54.16	57.34	63.54	61.30	62.40	+2.63	+7.14	+5.06
	PageRank	60.33	53.65	56.79	64.83	62.55	63.67	+4.50	+8.90	+6.87

Table 1: Improvements of using the Iterative Approach at the Document Level

$\mathcal{G_L}$	ϕ	Conventional Sent			Iterative Sent			Improvement		
		P	R	F	P	R	F	ΔP	ΔR	ΔF
SubTree Paths	In-Degree	**60.83**	**50.70**	**55.30**	**61.80**	**56.23**	**58.88**	+0.96	+5.54	+3.58
	Out-Degree	56.18	46.82	51.07	59.64	54.11	56.74	+3.46	+7.29	+5.67
	Betweenness Centrality	59.40	49.51	54.01	61.66	56.08	58.74	+2.26	+6.57	+4.73
	Sum Inverse Path Length	56.67	47.23	51.52	59.45	54.01	56.60	+2.78	+6.78	+5.08
	HITS(hub)	55.49	46.25	50.45	59.51	54.06	56.65	**+4.02**	**+7.81**	**+6.20**
	HITS(authority)	56.80	47.34	51.64	60.30	54.84	57.44	+3.50	+7.50	+5.80
	PageRank	59.71	49.77	54.29	60.56	55.04	57.67	+0.84	+5.28	+3.38
Shortest Paths	In-Degree	**58.13**	**32.75**	**41.89**	63.79	42.11	50.73	+5.66	+9.36	+8.84
	Out-Degree	54.64	30.78	39.38	61.79	40.66	49.05	**+7.15**	+9.88	+9.67
	Betweenness Centrality	57.94	32.64	41.76	**64.11**	**42.32**	**50.98**	+6.17	+9.68	+9.22
	Sum Inverse Path Length	55.65	31.35	40.11	62.39	41.02	49.50	+6.74	+9.67	+9.39
	HITS(hub)	56.11	31.61	40.44	62.74	41.28	49.80	+6.63	+9.67	+9.36
	HITS(authority)	55.74	31.40	40.17	62.75	41.39	49.88	+7.01	**+9.98**	**+9.70**
	PageRank	56.84	32.02	40.97	63.17	41.70	50.23	+6.33	+9.67	+9.27

Table 2: Improvements of using the Iterative Approach at the Sentence Level

senses anchored to the same lemma assisting each other's ϕ score (as discussed in Section 3.1), the SENSE_SHIFTS filter that is provided by the BabelNet API was also applied. This filter removes any path $P_{a \to b}$ such that $s_a, s_b \in R(\ell_i)$. Disambiguation was attempted at the document and sentence level, making use of the eight well-known graph centrality measures listed in section 4.2. For this experiment no means of optimisation were applied. Therefore Personalised PageRank was not used, and traditional PageRank took on a uniform random surfing vector. Default values of 0.85 and 30 for damping factor and maximum iterations were set respectively.

4.3.2 Experiment 1: Observations

First and foremost, it is clear from Table 1 and 2 that the iterative approach outperforms the conventional approach, regardless of the subgraph used, level of disambiguation, or the graph centrality measure employed. Since no graph centrality measure or subgraph were optimised, let this experiment prove that the iterative approach has the potential to improve any WSD system that implements it.

At the document level for both subgraphs the F-Scores were very close to the Most Frequent Sense (MFS) baseline for this task of 66.50. It is notoriously hard to beat and only one team (Gutiérrez et al., 2013) managed to beat it for this task. For all subtree subgraphs, we observe that In-Degree is clearly the best choice of centrality measure, while HITS (hub) enjoys the most improvement. We also observe that applying the iterative approach to Betweenness Centrality on shortest paths is a great combination at both the document and sentence level, most probably due to the measure being based on shortest paths. Furthermore it is

worth noting, the results at the sentence level for all graph centrality measures on shortest path subgraphs are quite poor, but highly improved, this is likely to our restriction of $L = 2$ causing the subgraphs to be much sparser and broken up into many components.

We also provide here an example from the data set in which the incorrect disambiguation of the lemma *cup* via the conventional approach was corrected by the iterative approach. This example is the seventh sentence in the eleventh document (d011.s007). Each word's degree of polysemy is denoted in square brackets.

"Spanish **[1]football players** playing in the All-Star **[4]League** and in powerful **[12]clubs** of the **[2]Premier League** of **[9]England** are during the **[5]year** very active in **[4]league** and local **[8]cup [7]competitions** and there are high-level **[25]shocks** in the **[10]European Cups** and **[2]European Champions League**."

The potential graph constructed from this sentence is illustrated in Figure 5 as a shortest paths subgraph. The darker edges portray the subgraph iteratively constructed up to a polysemy $\rho \leq 8$ (in order to disambiguate *cup*), whereas the lighter edges portray the greater subgraph constructed if the conventional approach is employed. Note that although the lemma *cup* has eight senses, only three are shown due to the application of the previously mentioned SENSE_SHIFTS filter. The remaining five senses of *cup* were filtered out since they were not able to link to a sense up to $L = 2$ hops away that is anchored to an alterative lemma.

- **cup#1** - A small open container usually used for drinking; usually has a handle.

- **cup#7** - The hole (or metal container in the hole) on a golf green.

- **cup#8** - A large metal vessel with two handles that is awarded as a trophy to the winner of a competition.

Given the context, the eighth sense of cup is the correct sense, the type we know as a trophy. For the conventional approach, if ϕ is a centrality measure of Out-Degree then the eighth sense of cup is easily chosen by having one extra outgoing edge than the other two senses for *cup*. Yet if ϕ is a centrality measure of In-Degree or Betweenness Centrality, all three senses of cup now have the same score, zero. Therefore in our results the first sense is chosen which is incorrect. On the other hand, if ϕ is
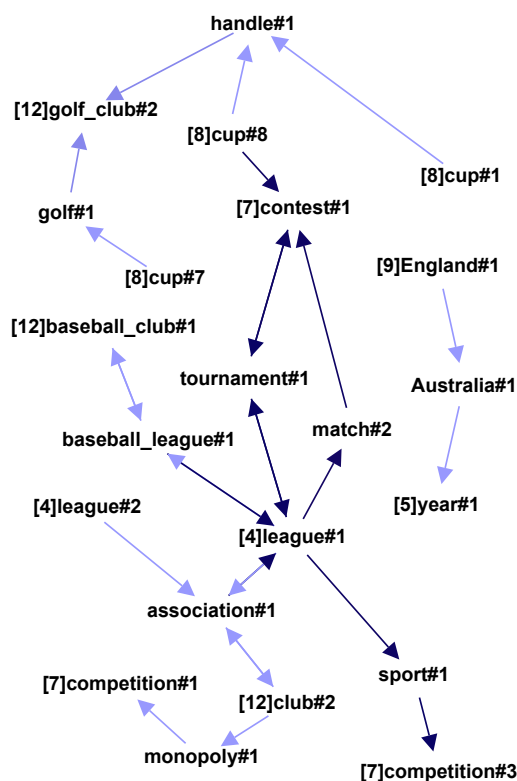

Figure 5: Conventional vs Iterative Subgraph

the subgraph was constructed iteratively with disambiguation results providing feedback to consecutive constructions, this could have been avoided. The shortest paths cup#1→handle#1→golf_club#2 and cup#7→golf#1→golf_club#2 only exist because the sense golf_club#2 (anchored to the more polysemous lemma *club*) is present, if it was not then the SENSE_SHIFTS filter would have removed these alternative senses. This demonstrates that if the senses of more polysemous lemmas are introduced into the subgraph too soon, they can interfere rather than help with disambiguation.

Secondly with each disambiguation at lower levels of polysemy, a more stable context is constructed to perform the disambiguation of much more polysemous lemmas later. Therefore in Figure 5 an iteratively constructed subgraph with *cup* already disambiguated, would mean the other two senses of *cup* would no longer be present. This ensures that club#2 (the correct answer) would have a much stronger chance of being selected than golf_club#2, which would have only one incoming edge from handle#1. Note the conventional approach would lend golf_club#2 one extra incoming edge than club#2 has, which could be problematic if ϕ is a centrality measure of In-Degree.
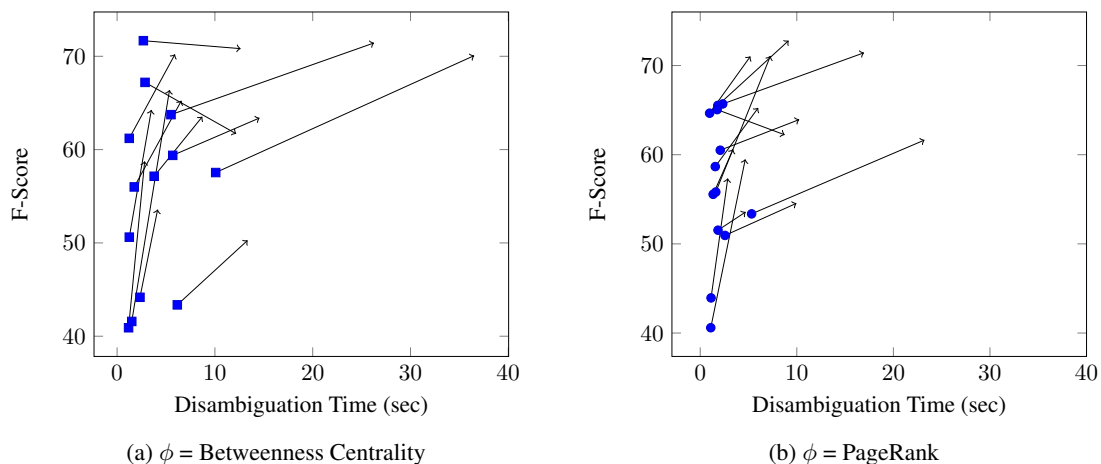
(a) ϕ = Betweenness Centrality

(b) ϕ = PageRank

Figure 6: For each of the 13 documents, performance (F-Score) is plotted against time to disambiguate, for $\mathcal{G}_{\mathcal{L}}$ = Shortest Paths. The squares (PageRank) and circles (Betweenness Centrality) plot the conventional approach. The arrows show the effect caused by applying the iterative approach, with the arrow head marking its F-Score and time to disambiguate.

4.4 Experiment 2: Performance

4.4.1 Experiment 2: Setup

An obvious caveat of the iterative approach is that it requires the construction of several subgraphs as ρ increases, which of course will require extra computation and time which is a penalty for the improved precision and recall. We decided to investigate the extent to which this happens. We selected Betweenness Centrality and PageRank from Experiment 1, in which both use shortest path subgraphs at the document level. This is because a) they acquired good results at the document level and b) with only 13 documents there are less data points on the plots making it easier to read as opposed to the hundreds of sentences.

4.4.2 Experiment 2: Observations

Firstly from Figures 6(a) and (b) we see that there is a substantial improvement in F-Score for almost all documents, except for two for ϕ = Betweenness Centrality and one for ϕ = PageRank. With some exceptions, for most documents the increased amount of time to disambiguate is not unreasonable. For this experiment, applying the iterative approach to Betweenness Centrality resulted in a mean 231% increase in processing time, from 3.54 to 11.73 seconds to acquire a mean F-Score improvement of +8.85. Again for PageRank, a mean increase of 343% in processing time, from 1.95 to 8.64 seconds to acquire a F-Score improvement of +7.16 was observed.

We wanted to investigate why in some cases, the iterative approach can produce poorer results than the conventional approach. We looked at aspects of the subgraphs such as order, size, density, and number of components. Eventually we came to the conclusion that, just like in a Sudoku puzzle, if there are not enough hints to start with, the possibility of finishing the puzzle becomes slim.

Therefore we suspected that if there were not enough monosemous lemmas, to construct the initial $\mathcal{G}_{\mathcal{L}}$, then the effectiveness of the iterative approach could be negated. It turns out, as observed in Figures 7(a) and (b) on the following page that this does effect the outcome. On the horizontal axis, document monosemy represents the percentage of lemmas in a document, not counting duplicates, that are monosemous. The vertical axis on the other hand represents the difference in F-Score between the conventional and iterative approach. Through a simple linear regression of the scatter plot, we observe an increased effectiveness of the iterative approach. This observation is important, because a WSD system may decide on which approach to use based on a document's monosemy.

With m representing document monosemy, and ΔF representing the change in F-Score induced by the iterative approach, the slopes observed in Figures 7(a) and (b) are denoted by Equations (2) and (3) respectively.

$$\Delta F = 0.53m - 0.11 \qquad (2)$$

$$\Delta F = 0.60m - 3.07 \qquad (3)$$

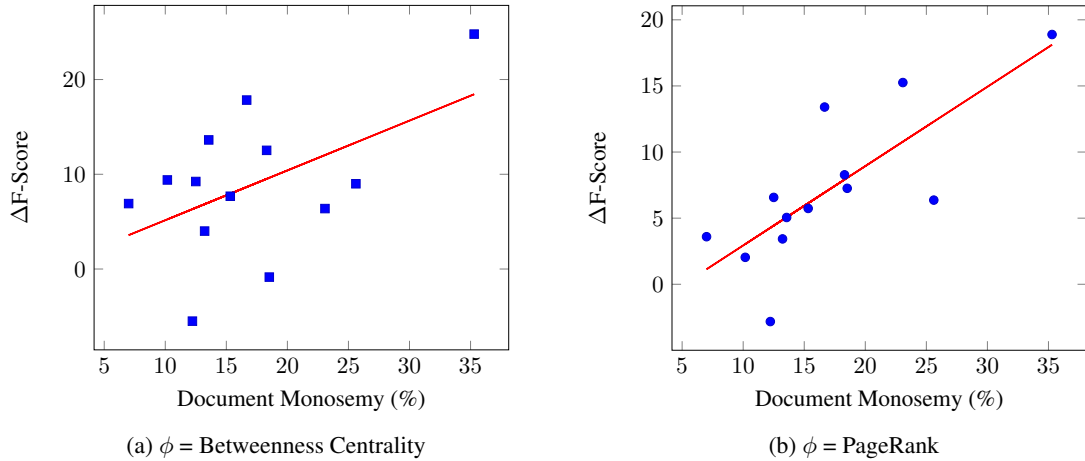(a) ϕ = Betweenness Centrality

(b) ϕ = PageRank

Figure 7: Both PageRank (squares) and Betweenness Centrality (circles) are plotted. Each data plot represents the change in F-Score when the iterative approach replaces the conventional approach with respect to the monosemy of the document.

4.5 Experiment 3: A Little Optimisation

Briefly, we made an effort into optimising the iterative approach with subtree subgraphs, and compared these results with systems from SemEval 2013 Task 12 (Navigli et al., 2013) in Table 3.

Team	System	P	R	F
UMCC-DLSI	Run-2[+]	68.50	68.50	68.50
UMCC-DLSI	Run-3[+]	68.00	68.00	68.00
UMCC-DLSI	Run-1[+]	67.70	67.70	67.70
SUDOKU	It-PPR[M][+]	67.62	67.51	67.56
MACHINE	**MFS**	66.50	66.50	66.50
SUDOKU	It-PPR[M]	67.20	65.49	66.33
SUDOKU	It-PR[U]	64.07	62.44	63.24
SUDOKU	It-PD	63.58	61.41	62.47
DAEBAK!	PD[+]	60.47	60.37	60.42
GETALP	BN-1[+]	58.30	58.30	58.30
SUDOKU	PR[U]	60.09	54.06	56.91
GETALP	BN-2[+]	56.80	56.80	56.80

Table 3: Comparison to SemEval 2013 Task 12

Firstly, we were able to marginally improve our original result as team DAEBAK! (Manion and Sainudiin, 2013), by applying the iterative approach to our Peripheral Diversity centrality measure (It-PD). Next we tried Personalised PageRank (It-PPR[M]) with a surfing vector biased towards only *M*onosemous senses. We also included regular PageRank (It-/PR[U]) with a *U*niform surfing vector as a reference point. It-PPR[M] almost defeated the MFS baseline of 66.50, but lacked recall. To rectify this, the MFS baseline was used as a back-off strategy (It-PPR[M][+])[4], which then led

to us beating the MFS baseline. As for the other teams, GETALP (Schwab et al., 2013) made use of an Ant Colony algorithm, while UMCC-DLSI (Gutiérrez et al., 2013) also made use of PPR, except they based the surfing vector on SemCor (Miller et al., 1993) sense frequencies, set $L = 5$ for shortest paths subgraphs, and disambiguated using resources external to BabelNet. Since their implementation of PPR beats ours, it would be interesting to see how effective the iterative approach could be on their results.

5 Conclusion & Future Work

In this paper we have shown that the iterative approach can substantially improve the results of regular subgraph-based WSD, even to the point of defeating the MFS baseline without doing anything complicated. This is regardless of the subgraph, graph centrality measure, or level of disambiguation. This research can still be extended further, and we encourage other researchers to rethink their own approaches to unsupervised knowledge-based WSD, particularly in regards to the interaction of subgraphs and centrality measures.

Resources

Codebase and resources are at first author's homepage: http://www.stevemanion.com.

---

[4]Note that plus[+] implies the use of a back-off strategy.

References

Eneko Agirre and Aitor Soroa. 2008. Using the Multilingual Central Repository for Graph-Based Word Sense Disambiguation. In *Proceedings of the 6th Language Resources and Evaluation Conference (LREC'08)*, pages 1388–1392, Marrakech, Morocco. ELRA.

Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL'09)*, pages 33–41, Athens, Greece. ACL.

Eneko Agirre, Oier Lopez De Lacalle, Christiane Fellbaum, Shu-Kai Hsieh, Maurizio Tesconi, Monica Monachini, Piek Vossen, and Roxanne Segers. 2010. SemEval-2010 Task 17: All-words Word Sense Disambiguation on a Specific Domain. In *Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval-2010)*, pages 75–80, Uppsala, Sweden. ACL.

Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30:107–117.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts.

Linton C. Freeman. 1979. Centrality in Social Networks Conceptual Clarification. *Social Networks*, 1(3):215–239.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities*, 26:415–439.

Yoan Gutiérrez, Antonio Fernández Orquín, Franc Camara, Yenier Castañeda, Andy González, Andrés Montoyo, Rafael Muñoz, Rainel Estrada, Dennys D. Piug, Jose I. Abreu, and Roger Pérez. 2013. UMCC_DLSI: Reinforcing a Ranking Algorithm with Sense Frequencies and Multidimensional Semantic Resources to solve Multilingual Word Sense Disambiguation. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval-2013), in conjunction with the Second Joint Conference on Lexical and Computational Semantics (*SEM'13)*, pages 241–249, Atlanta, Georgia. ACL.

Taher H. Haveliwala. 2003. Topic-Sensitive Pagerank: A Context-Sensitive Ranking Algorithm for Web Search. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):784–796.

Adam Kilgarriff. 1998. SENSEVAL: An Exercise in Evaluating Word Sense Disambiguation Programs. In *Proceedings of the 1st Language Resources and Evaluation Conference (LREC'98)*, pages 581–585, Granada, Spain.

Jon M. Kleinberg. 1999. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5):604–632.

Els Lefever and Veronique Hoste. 2010. SemEval-2010 Task 3: Cross-Lingual Word Sense Disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval-2010)*, pages 15–20, Uppsala, Sweden. ACL.

Els Lefever and Veronique Hoste. 2013. SemEval-2013 Task 10: Cross-lingual Word Sense Disambiguation. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval-2013), in conjunction with the Second Joint Conference on Lexical and Computational Semantics (*SEM'13)*, pages 158–166, Atlanta, Georgia. ACL.

Steve L. Manion and Raazesh Sainudiin. 2013. DAEBAK!: Peripheral Diversity for Multilingual Word Sense Disambiguation. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval-2013), in conjunction with the Second Joint Conference on Lexical and Computational Semantics (*SEM'13)*, pages 250–254, Atlanta, Georgia. ACL.

Rada Mihalcea. 2005. Unsupervised Large-Vocabulary Word Sense Disambiguation with Graph-based Algorithms for Sequence Data Labeling. In *Proceedings of the 2005 Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP'05)*, pages 411–418, Vancouver, British Columbia. ACL.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A Semantic Concordance. In *Proceedings of the Workshop on Human Language Technology (HLT'93)*, pages 303–308, Princeton, New Jersey. Morgan Kaufmann Publishers.

Roberto Navigli and Mirella Lapata. 2007. Graph Connectivity Measures for Unsupervised Word Sense Disambiguation. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 1683–1688, Hyderabad, India.

Roberto Navigli and Mirella Lapata. 2010. An Experimental Study of Graph Connectivity for Unsupervised Word Sense Disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):678–692.

Roberto Navigli and Simone Paolo Ponzetto. 2012a. BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. *Artificial Intelligence*, 193:217–250.

Roberto Navigli and Simone Paolo Ponzetto. 2012b. Joining Forces Pays Off: Multilingual Joint Word Sense Disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP/CoNLL'12)*, pages 1399–1410, Jeju Island, South Korea. ACL.

Roberto Navigli and Paola Velardi. 2005. Structural Semantic Interconnections: A Knowledge-based Approach to Word Sense Disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1075–1086.

Roberto Navigli, Kenneth C Litkowski, and Orin Hargraves. 2007. SemEval-2007 Task 07: Coarse-Grained English All-Words Task. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, pages 30–35, Prague, Czech Republic. ACL.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 Task 12: Multilingual Word Sense Disambiguation. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval-2013), in conjunction with the Second Joint Conference on Lexical and Computational Semantics (*SEM'13)*, pages 222–231, Atlanta, Georgia. ACL.

Roberto Navigli. 2009. Word Sense Disambiguation: A Survey. *ACM Computing Surveys*, 41(2):10:1–10:69.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English Tasks: All-Words and Verb Lexical Sample. In *Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL-2)*, pages 21–24, Toulouse, France. ACL.

Ted Pedersen. 2007. Unsupervised Corpus-Based Methods for WSD. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, chapter 6, pages 133–166. Springer, New York.

Simone Paolo Ponzetto and Roberto Navigli. 2010. Knowledge-rich Word Sense Disambiguation Rivaling Supervised Systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, pages 1522–1531, Uppsala, Sweden. ACL.

Didier Schwab, Andon Tchechmedjiev, Jérôme Goulian, Mohammad Nasiruddin, Gilles Sérasset, and Hervé Blanchon. 2013. GETALP: Propagation of a Lesk Measure through an Ant Colony Algorithm. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval -2013), in conjunction with the Second Joint Conference on Lexical and Computational Semantics (*SEM'13)*, pages 232–240, Atlanta, Georgia. ACL.

Ravi Sinha and Rada Mihalcea. 2007. Unsupervised Graph-based Word Sense Disambiguation Using Measures of Word Semantic Similarity. In *Proceedings of the International Conference on Semantic Computing (ICSC'07)*, pages 363–369, Irvine, California. IEEE.

Benjamin Snyder and Martha Palmer. 2004. The English All-Words Task. In *Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL-3)*, pages 41–43, Barcelona, Spain. ACL.

David Yarowsky. 1993. One Sense Per Collocation. In *Proceedings of the ARPA Workshop on Human Language Technology (HLT'93)*, pages 266–271, Morristown, New Jersey. ACL.

David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL'95)*, pages 189–196, Cambridge, Massachusetts. ACL.