Word Sense Induction & Disambiguation Using Hierarchical Random Graphs

Ioannis P. Klapaftis
Department of Computer Science
University of York
United Kingdom
gianni s@cs.york.ac.uk

Suresh Manandhar
Department of Computer Science
University of York
United Kingdom
suresh@cs.york.ac.uk

Abstract

Graph-based methods have gained attention in many areas of Natural Language Processing (NLP) including Word Sense Disambiguation (WSD), text summarization, keyword extraction and others. Most of the work in these areas formulate their problem in a graph-based setting and apply unsupervised graph clustering to obtain a set of clusters. Recent studies suggest that graphs often exhibit a hierarchical structure that goes beyond simple flat clustering. This paper presents an unsupervised method for inferring the hierarchical grouping of the senses of a polysemous word. The inferred hierarchical structures are applied to the problem of word sense disambiguation, where we show that our method performs significantly better than traditional graph-based methods and agglomerative clustering yielding improvements over state-of-the-art WSD systems based on sense induction.

1 Introduction

A number of NLP problems can be cast into a graph-based framework, in which entities are represented as vertices in a graph and relations between them are depicted by weighted or unweighted edges. For instance, in unsupervised WSD a number of methods (Widdows and Dorow, 2002; Véronis, 2004; Agirre et al., 2006) have constructed word co-occurrence graphs for a target polysemous word and applied graph-clustering to obtain the clusters (senses) of that word.

Similarly in text summarization, Mihalcea (2004) developed a method, in which sentences are represented as vertices in a graph and edges between them are drawn according to their common tokens or words of a given POS category, e.g. nouns. Graph-based ranking algorithms, such as PageRank (Brin and Page, 1998), were then applied in order to determine the significance of sentences. In the same vein, graph-based methods have been applied to other problems such as determining semantic similarity of text (Ramage et al., 2009).

Recent studies (Clauset et al., 2006; Clauset et al., 2008) suggest that graphs exhibit a hierarchical structure (e.g. a binary tree), in which vertices are divided into groups that are further subdivided into groups of groups, and so on, until we reach the leaves. This hierarchical structure provides additional information as opposed to flat clustering by explicitly including organisation at all scales of a graph (Clauset et al., 2008). In this paper, we present an unsupervised method for inferring the hierarchical structure (binary tree) of a graph, in which vertices are the contexts of a polysemous word and edges represent the similarity between contexts. The method that we use to infer that hierarchical structure is the Hierarchical Random Graphs (HRGs) algorithm due to Clauset et al. (2008).

The binary tree produced by our method groups the contexts of a polysemous word at different heights of the tree. Thus, it induces the senses of that word at different levels of sense granularity. To evaluate our method, we apply it to the problem of noun sense disambiguation showing that inferring the hierarchical structure using HRGs provides additional information from the observed graph leading to improved WSD performance compared to: (1)

Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 745–755,
MIT, Massachusetts, USA, 9-11 October 2010.

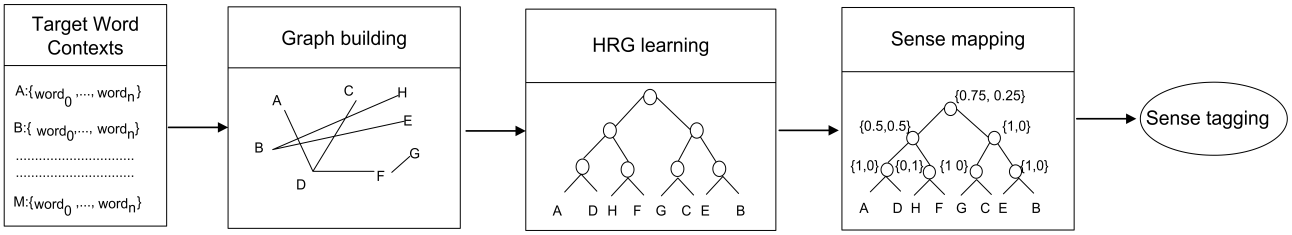


Figure 1: Stages of the proposed method.

simple flat clustering, and (2) traditional agglomerative clustering. Finally, we compare our results with state-of-the-art sense induction systems and show that our method yields improvements. Figure 1 shows the different stages of the proposed method that we describe in the following sections.

2 Related work

Typically, graph-based methods, when applied to unsupervised sense disambiguation represent each word w_i co-occurring with the target word tw as a vertex. Two vertices are connected via an edge if they co-occur in one or more contexts of tw. Once the co-occurrence graph of tw has been constructed, different graph clustering algorithms are applied to induce the senses. Each cluster (induced sense) consists of a set of words that are semantically related to the particular sense. Figure 2 shows an example of a graph for the target word *paper* that appears with two different senses *scholarly article* and *newspaper*.

Véronis (2004) has shown that co-occurrence graphs are small-world networks that contain highly dense subgraphs representing the different clusters (senses) of the target word (Véronis, 2004). To identify these dense regions Véronis's algorithm iteratively finds their hubs, where a hub is a vertex with a very high *degree*. The degree of a vertex is defined to be the number of edges incident to that vertex. The identified hub is then deleted along with its direct neighbours from the graph producing a new cluster.

For example, in Figure 2 the highest degree vertex, *news*, is the first hub, which would be deleted along with its direct neighbours. The deleted region corresponds to the *newspaper* sense of the target word *paper*. Véronis (2004) further processed the identified clusters (senses), in order to assign the rest of graph vertices to the identified clusters by utilising the *minimum spanning tree* of the original graph.

In Agirre et al. (2006), the algorithm of Véronis (2004) is analysed and assessed on the SensEval-3 dataset (Snyder and Palmer, 2004), after optimising its parameters on the SensEval-2 dataset (Edmonds and Dorow, 2001). The results show that the WSD F-Score outperforms the Most Frequent Sense (MFS) baseline by approximately 10%, while inducing a large number of clusters (with averages of 60 to 70).

Another graph-based method is presented in (Dorow and Widdows, 2003). They extract only noun neighbours that appear in conjunctions or disjunctions with the target word. Additionally, they extract second-order co-occurrences. Nouns are represented as vertices, while edges between vertices are drawn, if their associated nouns co-occur in conjunctions or disjunctions more than a given number of times. This co-occurrence frequency is also used to weight the edges. The resulting graph is then pruned by removing the target word and vertices with a low degree. Finally, the MCL algorithm (Dongen, 2000) is used to cluster the graph and produce a set of clusters (senses) each one consisting of a set of contextually related words.

Chinese Whispers (CW) (Biemann, 2006) is a parameter-free[1] graph clustering method that has been applied in sense induction to cluster the co-occurrence graph of a target word (Biemann, 2006), as well as a graph of collocations related to the target word (Klapaftis and Manandhar, 2008). The evaluation of the collocational-graph method in the SemEval-2007 sense induction task (Agirre and Soroa, 2007) showed promising results.

All the described methods for sense induction ap-

[1] One needs to specify only the number of iterations. The number of clusters is generated automatically.

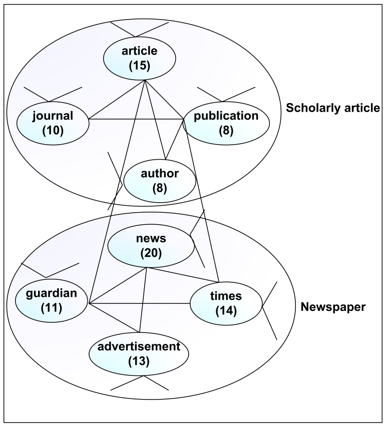
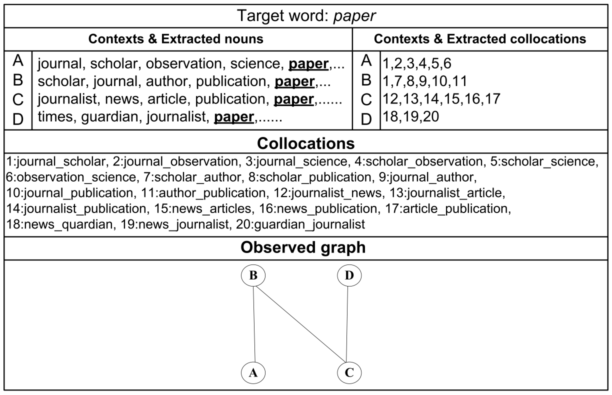


Figure 2: Graph of words for the target word *paper*. Numbers inside vertices correspond to their degree.



Target word: *paper*			
Contexts & Extracted nouns		**Contexts & Extracted collocations**	
A B C D	journal, scholar, observation, science, **paper**,... scholar, journal, author, publication, **paper**,... journalist, news, article, publication, **paper**,...... times, guardian, journalist, **paper**,......	A B C D	1,2,3,4,5,6 1,7,8,9,10,11 12,13,14,15,16,17 18,19,20
Collocations			
1:journal_scholar, 2:journal_observation, 3:journal_science, 4:scholar_observation, 5:scholar_science, 6:observation_science, 7:scholar_author, 8:scholar_publication, 9:journal_author, 10:journal_publication, 11:author_publication, 12:journalist_news, 13:journalist_article, 14:journalist_publication, 15:news_articles, 16:news_publication, 17:article_publication, 18:news_quardian, 19:news_journalist, 20:guardian_journalist			
Observed graph			

Figure 3: Running example of graph creation

ply flat graph clustering methods to derive the clusters (senses) of a target word. As a result, they neglect the fact that their constructed graphs often exhibit a hierarchical structure that is useful in several tasks including word sense disambiguation.

3 Building a graph of contexts

This section describes the process of creating a graph of contexts for a polysemous target word. Figure 3 provides a running example of the different stages of our method. In the example, the target word *paper* appears with the *scholarly article* sense in the contexts *A*, *B*, and with the *newspaper* sense in the contexts *C* and *D*.

3.1 Corpus preprocessing

Let bc denote the base corpus consisting of the contexts containing the target word tw. In our work, a context is defined as a paragraph[2] containing the target word.

The aim of this stage is to capture nouns contextually related to tw. Initially, the target word is removed from bc and part-of-speech tagging is applied to each context. Following the work in (Véronis, 2004; Agirre et al., 2006) only nouns are kept and lemmatised. In the next step, the distribution of each noun in the base corpus is compared to the distribution of the same noun in a reference corpus[3] using the log-likelihood ratio (G^2) (Dunning, 1993). Nouns with a G^2 below a pre-specified threshold (parameter p_1) are removed from each paragraph of the base corpus. The upper left part of Figure 3 shows the words kept as a result of this stage.

3.2 Graph creation

Graph vertices: To create the graph of vertices, we represent each context c_i as a vertex in a graph G.

Graph edges: Edges between the vertices of the graph are drawn based on their similarity, defined in Equation 1, where $sim_{cl}(c_i, c_j)$ is the *collocational weight* of contexts c_i, c_j and $sim_{wd}(c_i, c_j)$ is their bag-of-words weight. If the edge weight $W(c_i, c_j)$ is above a prespecified threshold (parameter p_3), then an edge is drawn between the corresponding vertices in the graph.

$$W(c_i, c_j) = \frac{1}{2}(sim_{cl}(c_i, c_j) + sim_{wd}(c_i, c_j)) \quad (1)$$

Collocational weight: The limited polysemy of collocations can be exploited to compute the similarity between contexts c_i and c_j. In our setting, a collocation is a juxtaposition of two nouns within the same context. Thus, given a context c_i, each of its nouns is combined with any other noun yielding a total of $\binom{N}{2}$ collocations for a context with N nouns. Each collocation, cl_{ij} is weighted using the log-likelihood ratio (G^2) (Dunning, 1993) and is filtered out if the G^2 is below a prespecified threshold (parameter p_2). At the end of this process, each context c_i of tw is associated with a vector of collocations (v_i). The upper right part of Figure 3 shows the collocations associated with each context of our example.

[2] Our definition of *context* is equivalent to an instance of the target word in the SemEval-2007 sense induction task dataset (Agirre and Soroa, 2007).

[3] The British National Corpus, 2001, Distributed by Oxford University Computing Services.

Given two contexts c_i and c_j, we calculate their collocational weight using the Jaccard coefficient on the collocational vectors, i.e. $sim_{cl}(c_i, c_j) = \frac{|v_i \cap v_j|}{|v_i \cup v_j|}$. The selection of Jaccard is based on the work of Weeds et al. (2004), who analyzed the variation in a word's distributionally nearest neighbours with respect to a variety of similarity measures. Their analysis showed that there are three classes of measures, i.e. those selecting distributionally more general neighbours (e.g. cosine), those selecting distributionally less general neighbours (e.g. AMCRM-Precision (Weeds et al., 2004)) and those without a bias towards the distributional generality of a neighbour (e.g. Jaccard). In our setting, we are interested in calculating the similarity between two contexts without any bias. We selected Jaccard, since the rest of that class's measures are based on pointwise mutual information that assigns high weights to infrequent events.

Bag-of-words weight: Estimating context similarity using collocations may provide reliable estimates regarding the existence of an edge in the graph, however, it also suffers from data sparsity. For this reason, we also employ a bag-of-words model. Specifically, each context c_i is associated with a vector g_i that contains the nouns kept as result of the corpus preprocessing stage. The upper left part of Figure 3 shows the words associated with each context of our example. Given two contexts c_i and c_j, we calculate their bag-of-words weight using the Jaccard coefficient on the word vectors, i.e. $sim_{wd}(c_i, c_j) = \frac{|g_i \cap g_j|}{|g_i \cup g_j|}$.

The collocational weight and bag-of-words weight are averaged to derive the edge weight between two contexts as defined in Equation 1. The resulting graph of our running example is shown on the bottom of Figure 3. This graph is the input to the *hierarchical random graphs* method (Clauset et al., 2008) described in the next section.

4 Hierarchical Random Graphs for sense induction

In this section, we describe the process of inferring the hierarchical structure of the graph of contexts using *hierarchical random graphs* (Clauset et al., 2008).

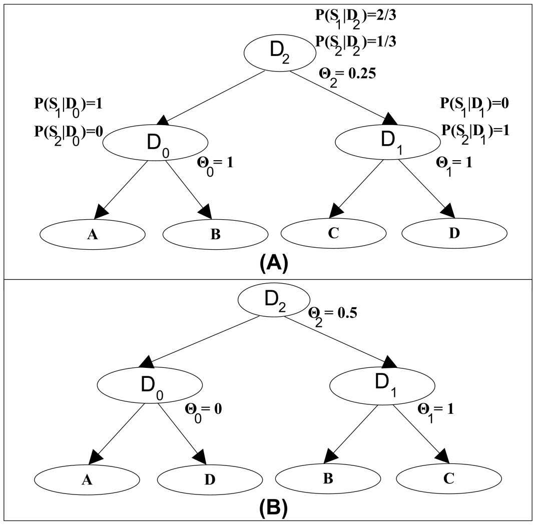


Figure 4: Two dendrograms for the graph in Figure 3.

4.1 The Hierarchical Random Graph model

A dendrogram is a binary tree with n leaves and $n-1$ parents. Figure 4 shows an example of two dendrograms with 4 leaves and 3 parents. Given a set of n contexts that we need to arrange hierarchically, let us denote by $G = (V, E)$ the graph of contexts, where $V = \{v_0, v_1 \ldots v_n\}$ is the set of vertices, $E = \{e_0, e_1 \ldots e_m\}$ is the set of edges and $e_k = \{v_i, v_j\}$.

Given an undirected graph G, each of its n vertices is a leaf in a dendrogram, while the internal nodes of that dendrogram indicate the hierarchical relationships among the leaves. We denote this organisation by $D = \{D_1, D_2, \ldots D_{n-1}\}$, where each D_k is an internal node. Every pair of nodes (v_i, v_j) is associated with a unique D_k, which is their lowest common ancestor in the tree. In this manner D partitions the edges that exist in G.

The primary assumption in the hierarchical random graph model is that edges in G exist independently, but with a probability that is not identically distributed. In particular, the probability that an edge $\{v_i, v_j\}$ exists in G is given by a parameter θ_k associated with D_k, the lowest common ancestor of v_i and v_j in D. In this manner, the topological structure D and the vector of probabilities $\vec{\theta}$ define the HRG given by $H(D, \vec{\theta})$ (Clauset et al., 2008).

4.2 HRG parameterisation

Assuming a uniform prior over all HRGs, the target is to identify the parameters of D and $\vec{\theta}$, so that the chosen HRG is statistically similar to G. Let D_k be an internal node of dendrogram D and $f(D_k)$ be the number of edges between the vertices of the subtrees of the subtree rooted at D_k that actually exist in G. For example, in Figure 4(A), $f(D_2) = 1$, because there is one edge in G connecting vertices B and C. Let $l(D_k)$ be the number of leaves in the left subtree of D_k, and $r(D_k)$ be the number of leaves in the right subtree. For example in Figure 4(A), $l(D_2) = 2$ and $r(D_2) = 2$. The likelihood of the hierarchical random graph $(D, \vec{\theta})$ is defined in Equation 2, where $A(D_k) = l(D_k)r(D_k) - f(D_k)$.

$$L(D, \vec{\theta}) = \prod_{D_k \in D} \theta_k^{f(D_k)} (1 - \theta_k)^{A(D_k)} \quad (2)$$

The probabilities θ_k that maximise the likelihood of a dendrogram D can be easily estimated using the method of MLE i.e $\overline{\theta}_k = \frac{f(D_k)}{l(D_k)r(D_k)}$. Substituting this into Equation 2 yields Equation 3. For numerical reasons, it is more convenient to work with the logarithm of the likelihood which is defined in Equation 4, where $h(\overline{\theta}_k) = -\overline{\theta}_k \log \overline{\theta}_k - (1 - \overline{\theta}_k) \log (1 - \overline{\theta}_k)$.

$$L(D) = \prod_{D_k \in D} [\overline{\theta}_k^{\overline{\theta}_k} (1 - \overline{\theta}_k)^{1 - \overline{\theta}_k}]^{l(D_k)r(D_k)} \quad (3)$$

$$\log L(D) = - \sum_{D_k \in D} h(\overline{\theta}_k) l(D_k) r(D_k) \quad (4)$$

As can be observed, each term $-l(D_k)r(D_k)h(\overline{\theta}_k)$ is maximised when θ_k approaches 0 or 1. This means that high-likelihood dendrograms partition vertices into subtrees, such that the connections among their vertices in the observed graph are either very rare or very common (Clauset et al., 2008). For example, consider the two dendrograms in Figures 4(A) and 4(B). We observe that 4(A) is more likely than 4(B), since it provides a better division of the network leaves. Particularly, the likelihood of 4(A) is $L(D_1) = (1^1 \cdot (1-1)^1) \cdot (1^1 \cdot (1-1)^1) \cdot (0.25^1 \cdot (1 - 0.25)^3) = 0.105$, while the likelihood of 4(B) is $L(D_2) = (0^0 \cdot (1-0)^1) \cdot (1^1 \cdot (1-1)^1) \cdot (0.5^2 \cdot (1 - 0.5)^2) = 0.062$.

4.2.1 MCMC sampling

Finding the values of θ_k using the MLE method is straightforward. However, this is not the case for maximising the likelihood function over the space of all possible dendrograms. Given a graph with n vertices, i.e. n leaves in each dendrogram, the total number of different dendrograms is superexponential ($(2n-3)!! \approx \sqrt{2}(2n)^{n-1}e^{-n}$) (Clauset et al., 2006).

To deal with this problem, we use a Markov Chain Monte Carlo (MCMC) method that samples dendrograms from the space of dendrogram models with probability proportional to their likelihood. Each time MCMC samples a dendrogram with a new highest likelihood, that dendrogram is stored. Hence, our goal is to choose the highest likelihood dendrogram once MCMC has converged.

Following the work in (Clauset et al., 2008), we pick a set of transitions between dendrograms, where a transition is a re-arrangement of the subtrees of a dendrogram. In particular, given a current dendrogram D_{curr}, each internal node D_k of D_{curr} is associated with three subtrees of D_{curr}. For instance, in Figure 5A, the subtrees st_1 and st_2 are derived from the two children of D_k and the third st_3 from its sibling. Given a current dendrogram, D_{curr}, the algorithm proceeds as follows:

1. Choose an internal node, $D_k \in D_{curr}$ uniformly.

2. Generate two possible new configurations of the subtrees of D_k (See Figure 5).

3. Choose one of the configurations uniformly to generate a new dendrogram, D_{next}.

4. Accept or reject D_{next} according to Metropolis-Hastings (MH) rule.

5. If transition is accepted, then $D_{curr} = D_{next}$.

6. GOTO 1.

According to MH rule (Newman and Barkema, 1999), a transition is accepted if $\log L(D_{next}) \geq \log L(D_{curr})$; otherwise the transition is accepted with probability $\frac{L(D_{next})}{L(D_{curr})}$. These transitions define an ergodic Markov chain, hence its stationary distribution can be reached (Clauset et al., 2008).

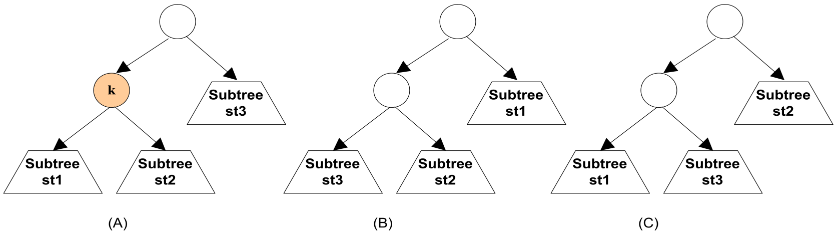


Figure 5: (A) current configuration for internal node D_k and its associated subtrees (B) first alternative configuration, (C) second alternative configuration. Note that swapping $st1$, $st2$ in (A) results in an equivalent tree. Hence, this configuration is excluded.

In our experiments, we noticed that the algorithm converged relatively quickly. The same behaviour (roughly $O(n^2)$ steps) was also noticed in Clauset et al. (2008), when considering graphs with thousands of vertices.

5 HRGs for sense disambiguation

5.1 Sense mapping

The output of HRG learning is a dendrogram D with n leaves (contexts) and $n-1$ internal nodes. To perform sense disambiguation, we mapped the internal nodes to gold standard senses using a sense-tagged corpus. Such a sense-tagged corpus is needed when induced word senses need to be mapped to a gold standard sense inventory.

Instead of using a hard mapping from the dendrogram internal nodes to the Gold Standard (GS) senses, we use a soft probabilistic mapping and calculate $P(s_k|D_i)$, i.e the probability of sense s_k given node D_i. Let $F(D_i)$ be the set of training contexts grouped by internal node D_i. Let $F'(s_k)$ be the set of training contexts that are tagged with sense s_k. Then the conditional probability, $P(s_k|D_i)$, is defined in Equation 5.

$$P(s_k|D_i) = \frac{|F(D_i) \cap F'(s_k)|}{|F(D_i)|} \qquad (5)$$

Table 1 provides a sense-tagged corpus for the running example of Figure 3. Using this corpus and the tree in Figure 4(A), $P(s_1|D_2) = \frac{2}{3}$ and $P(s_2|D_2) = \frac{1}{3}$. In Figure 4(A) the rest of the calculated conditional probabilities are given.

GS sense	Context ID	Context words
s_1	A	journal, scholar, observation science, **paper**
s_1	B	scholar, scholar, author, publication, **paper**
s_2	D	times, guardian, journalist, **paper**

Table 1: Sense-tagged corpus for the example in Figure 3

5.2 Sense tagging

For evaluation we compared the proposed method against the current state-of-the-art sense induction systems in the WSD task. We followed the setting of SemEval-2007 sense induction task (Agirre and Soroa, 2007). In this setting, the base corpus (bc) (Section 3.1) for a target word consists both of the training and testing corpus. As a result, a testing context c_j of tw is a leaf in the generated dendrogram. The process of disambiguating c_j is straightforward exploiting the structural information provided by HRGs.

$$w(s_k, c_j) = \sum_{D_i \in H(c_j)} P(s_k|D_i) \cdot \theta_i \qquad (6)$$

$$w(s^*, c_j) = \operatorname{argmax} s_k(w(s_k, c_j)) \qquad (7)$$

Let $H(c_j)$ denote the set of parents for context c_j. Then, the weight assigned to sense s_k is the sum of weighted scores provided by each identified parent. This is shown in Equation 6, where θ_i is the probability associated with each internal node D_i from the hierarchical random graph (see Figure 4(A)). This probability reflects the discriminating ability of internal nodes.

Finally, the highest weight determines the winning sense for context c_j (Equation 7). In our example (Figure 4(A)), $w(s_1, C) = (0 \cdot 1 + \frac{2}{3} \cdot 0.25) = 0.16$ and $w(s_2, C) = (1 \cdot 1 + \frac{1}{3} \cdot 0.25) = 1.08$. Hence, s_2 is the winning sense.

Parameter	Range
G^2 word threshold (p_1)	15,25,35,45
G^2 collocation threshold (p_2)	10,15,20
Edge similarity threshold (p_3)	0.05,0.09,0.13

Table 2: Parameter values used in the evaluation.

6 Evaluation

6.1 Evaluation setting & baselines

We evaluate our method on the nouns of the SemEval-2007 word sense induction task (Agirre and Soroa, 2007) under the second evaluation setting of that task, i.e. supervised evaluation. Specifically, we use the standard WSD measures of precision and recall in order to produce their harmonic mean (F-Score). The official scoring software of that task has been used in our evaluation. Note that the unsupervised measures of that task are not directly applicable to our induced hierarchies, since they focus on assessing flat clustering methods.

The first aim of our evaluation is to test whether inferring the hierarchical structure of the constructed graphs improves WSD performance. For that reason our first baseline, Chinese Whispers Unweighted version (*CWU*), takes as input the same unweighted graph of contexts as HRGs in order to produce a flat clustering. The set of produced clusters is then mapped to GS senses using the training dataset and performance is then measured on the testing dataset. We followed the same sense mapping method as in the SemEval-2007 sense induction task (Agirre and Soroa, 2007).

Our second baseline, Chinese Whispers Weighted version (*CWW*), is similar to the previous one, with the difference that the edges of the input graph are weighted using Equation 1. For clustering the graphs of *CWU* and *CWW* we employ, *Chinese Whispers*[4] (Biemann, 2006).

The second aim of our evaluation is to assess whether the hierarchical structure inferred by HRGs is more informative than the hierarchical structure inferred by traditional Hierarchical Clustering (*HAC*). Hence, our third baseline, takes as input a similarity matrix of the graph vertices and performs bottom-up clustering with average-linkage, which has already been used in WSI in (Pantel and Lin, 2003) and was shown to have superior or similar performance to single-linkage and complete-linkage in the related problem of learning a taxonomy of senses (Klapaftis and Manandhar, 2010).

To calculate the similarity matrix of vertices we follow a process similar to the one used in Section 4.2 for calculating the probability of an internal node. The similarity between two vertices is calculated according to the degree of connectedness among their direct neighbours. Specifically, we would like to assign high similarity to pairs of vertices, whose neighbours are close to forming a clique.

Given two vertices (contexts) c_i and c_j, let $N(c_i, c_j)$ be the set of their neighbours and $K(c_i, c_j)$ be the set of edges between the vertices in $N(c_i, c_j)$. The maximum number of edges that could exist between vertices in $N(c_i, c_j)$ is $\binom{|N(c_i,c_j)|}{2}$. Thus, the similarity of c_i, c_j is set equal to the number of edges that actually exist in that neighbourhood divided by the total number of edges that could exist ($\frac{|K(c_i,c_j)|}{\binom{|N(c_i,c_j)|}{2}}$).

The disambiguation process using the HAC tree is identical to the one presented in Section 5.2 with the only difference that the internal probability, θ_i, in Equation 6 does not exist for HAC. Hence, we replaced it with the factor $\frac{1}{|H(D_i)|}$, where $H(D_i)$ is the set of children of internal node D_i. This factor provides lower weights for nodes high in the tree, since their discriminating ability will possibly be lower.

6.2 Results & discussion

Table 2 shows the parameter values used in the evaluation. Figure 6(A) shows the performance of the proposed method against the baselines for $p_3 = 0.05$ and different p_1 and p_2 values. Figure 6(B) illustrates the results of the same experiment using $p_3 = 0.09$. In both figures, we observe that *HRGs* outperform the *CWU* baseline under all parameter combinations. In particular, all of the 12 performance differences for $p_3 = 0.09$ are statistically significant using McNemar's test at 95% confidence level, while for $p_3 = 0.05$ only 2 out of the 12 performance differences were not judged as significant from the test.

The picture is the same for $p_3 = 0.13$, where *CWU* performs significantly worse than for $p_3 =$

[4] The number of iterations for CW was set to 200.

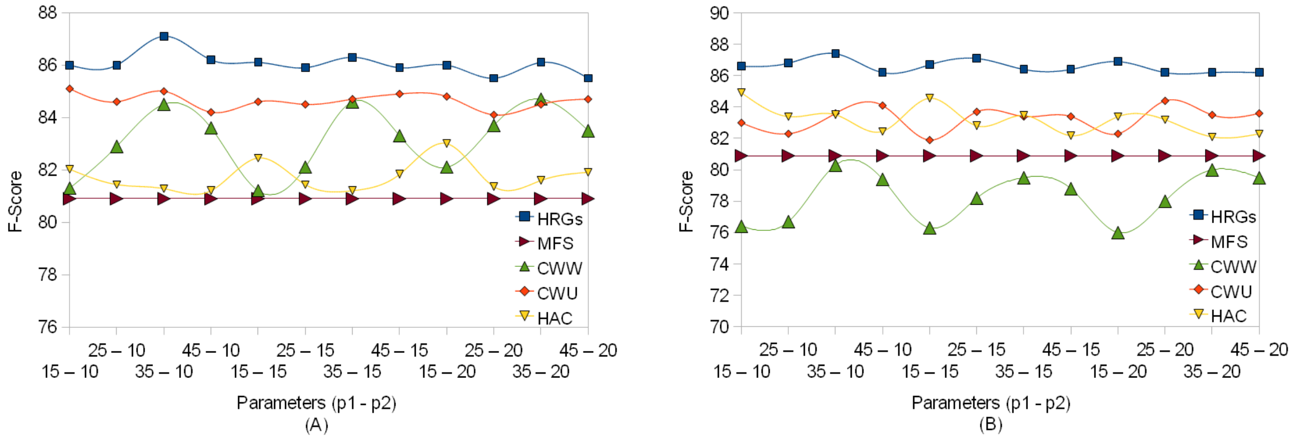


Figure 6: Performance analysis of HRGs, CWU, CWW & HAC for different parameter combinations (Table 2). **(A)** All combinations of p_1, p_2 and $p_3 = 0.05$. **(B)** All combinations of p_1, p_2 and $p_3 = 0.09$.

0.05 and $p_3 = 0.09$. Specifically, the largest performance difference between *HRGs* and *CWU* is 9.4% at $p_1 = 25$, $p_2 = 10$ and $p_3 = 0.13$. Setting the vertex similarity threshold (p_3) equal to 0.13 leads to more sparse and disconnected graphs, which causes *Chinese Whispers* to produce a large number of clusters. This leads to sparsity problems and unreliable mapping of clusters to GS senses due to the lack of adequate training data. In contrast, *HRGs* suffer less at this high threshold, although their performance when $p_3{<}0.13$ is better.

This picture does not change for the weighted version of *Chinese Whispers* (*CWW*) which performs worse than *CWU*. This is because *CWW* produces a smaller number of clusters than *CWU* that conflate the target word senses. It seems that using weighted edges creates a bias towards the MFS, in effect missing rare senses of a target word. This means that a number of words in the bag-of-words context vectors and collocations in the collocational context vectors (Section 3.2) are associated to more than one sense of the target word and most strongly associated to the MFS. As a result, increasing the p_1 threshold to 25 and 35 leads to a higher performance for *CWW*, since many of these words and collocations are filtered out.

Overall, the comparison of *HRGs* against the *CWU* and *CWW* baselines has shown that inferring the hierarchical structure of observed graphs leads to improved WSD performance as opposed to using flat clustering. This is because *HRGs* are able to infer both the hierarchical structure of the graph and include the probabilities, θ_k, associated with each internal node. These probabilities reflect the discriminating ability of each node, offering information missed by flat clustering.

In Figures 6(A) and 6(B) we observe that *HRGs* perform significantly better than HAC. In particular, all of their performance differences are statistically significant for these parameter values. The largest performance difference is 6.0% at $p_1 = 45$, $p_2 = 10$ and $p_3 = 0.05$. However, this picture is not the same when considering a higher context similarity threshold ($p_3 = 0.13$) as Figure 7 shows. In particular, *HRGs* and *HAC* perform similarly for $p_3 = 0.13$, while the majority of performance differences are not statistically significant.

The similar behaviour of *HRGs* and *HAC* at this threshold is caused both by the worse performance of *HRGs* and the improved performance of *HAC* as opposed to lower p_3 values. As it has been mentioned, setting $p_3 = 0.13$ leads to sparse and disconnected graphs. Additionally, the likelihood function (Equation 3) is maximised when the probability, θ_k, of an internal node, D_k, approaches 0 or 1. This creates a bias towards dendrograms, in which a large number of internal nodes have zero probability. These dendrograms might be a good-fit to the observed graph, but not to the GS.

In contrast, *HAC* is less affected, because it never considers creating an internal node, when the maximum similarity among any pair of two candidate

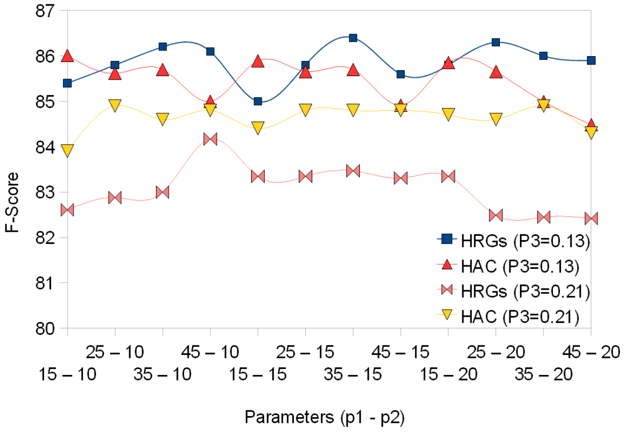


Figure 7: Performance of *HRGs* and *HAC* for different parameter combinations (Table 2). All combinations of p_1, p_2 and $p_3 \geq 0.13$.

subtrees is zero. Additionally, our experiments show that *HAC* is unable to deal with noise when considering sparse graphs (p_3<0.13). For that reason, the F-Score of *HAC* increases as the edge similarity threshold decreases.

To further investigate this issue and test whether *HAC* is able to achieve a higher F-Score than HRGs in higher p_3 values, we executed two more experiments for *HAC* and *HRGs* increasing p_3 to 0.17 and 0.21 respectively. In the first case we observed that the performance of *HAC* remained relatively stable compared to $p_3 = 0.13$, while in the second case the performance of *HAC* decreased as Figure 7 shows. In both cases, *HAC* performed significantly better than *HRGs*.

Overall, the comparison of *HRGs* against *HAC* has shown that *HRGs* perform significantly better than *HAC* when considering connected or less sparse graphs (p_3<0.13). This is due to the fact that *HAC* creates dendrograms, in which connections within the clusters are dense, while connections between the clusters are sparse, i.e. it only considers assortative structures. In contrast, HRGs also consider disassortative dendrograms, i.e. dendrograms in which vertices are less likely to be connected on small scales than on large ones, as well as mixtures of assortative and disassortative (Clauset et al., 2008). This is achieved by allowing the probability θ_k of a node k to vary arbitrarily throughout the dendrogram.

HAC performs similarly or better than *HRGs* for largely disconnected and sparse graphs, because HRGs become biased towards disassortative trees which are not a good fit to the GS (Figure 7). Despite that, our evaluation has also shown that the best performance of *HAC* (F-Score = 86.0% at $p_1 = 15$, $p_2 = 10$, $p_3 = 0.13$) is significantly lower than the best performance of *HRGs* (F-Score = 87.6% at $p_1 = 35$, $p_2 = 10$, $p_3 = 0.09$).

6.3 Comparison to state-of-the-art methods

Table 3 compares the best performing parameter combination of our method against state-of-the-art methods. Table 3 also includes the best performance of our baselines, i.e *HAC*, *CWU* and *CWW*.

Brody & Lapata (2009) presented a sense induction method that is related to Latent Dirichlet Allocation (Blei et al., 2003). In their work, they model the target word instances as samples from a multinomial distribution over senses which are successively characterized as distributions over words (Brody and Lapata, 2009). A significant advantage of their method is the inclusion of more than one layer in the LDA setting, where each layer corresponds to a different feature type e.g. dependency relations, bigrams, etc. The inclusion of different feature types as separate models in the sense induction process can easily be modeled in our setting, by inferring a different hierarchy of target word instances according to each feature type, and then combining all of them to a consensus tree. In this work, we have focused on extracting a single hierarchy combining word co-occurrence and bigram features.

Niu et al. (2007) developed a vector-based method that performs sense induction by grouping the contexts of a target word using three types of features, i.e. POS tags of neighbouring words, word co-occurrences and local collocations. The *sequential information bottleneck* algorithm (Slonim et al., 2002) is applied for clustering. *HRGs* perform slightly better than the methods of Brody & Lapata (2009) and Niu et al. (2007), although the differences are not significant (McNemar's test at 95% confidence level).

Klapaftis & Manandhar (2008) developed a graph-based sense induction method, in which vertices correspond to collocations related to the target word and edges between vertices are drawn ac-

System	Performance (%)
HRGs	87.6
(Brody and Lapata, 2009)	87.3
(Niu et al., 2007)	86.8
(Klapaftis and Manandhar, 2008)	86.4
HAC	86.0
CWU	85.1
CWW	84.7
(Pedersen, 2007)	84.5
MFS	80.9

Table 3: HRGs against recent methods & baselines.

cording to the co-occurrence frequency of the corresponding collocations. The constructed graph is smoothed to identify more edges between vertices and then clustered using Chinese Whispers (Biemann, 2006). This method is related to the basic inputs of our presented method. Despite that, it is a flat clustering method that ignores the hierarchical structure exhibited by observed graphs. The previous section has shown that inferring the hierarchical structure of graphs leads to superior WSD performance.

Pedersen (2007) presented *SenseClusters*, a vector-based method that clusters second order co-occurrence vectors using k-means, where k is automatically determined using the Adapted Gap Statistic (Pedersen and Kulkarni, 2006). As can be observed, *HRGs* perform significantly better than the methods of Pedersen (2007) and Klapaftis & Manandhar (2008) (McNemar's test at 95% confidence level).

Finally, Table 3 shows that the best performing parameter combination of HRGs achieves a significantly higher F-Score than the best performing parameter combination of *HAC*, *CWU* and *CWW*. Furthermore, HRGs outperform the most frequent sense baseline by 6.7%.

7 Conclusion & future work

We presented an unsupervised method for inferring the hierarchical grouping of the senses of a polysemous word. Our method creates a graph, in which vertices correspond to contexts of a polysemous target word and edges between them are drawn according to their similarity. The *hierarchical random graphs* algorithm (Clauset et al., 2008) was applied to the constructed graph in order to infer its hierarchical structure, i.e. binary tree.

The learned tree provides an induction of the senses of a given word at different levels of sense granularity and was applied to the problem of WSD. The WSD process mapped the tree's internal nodes to GS senses using a sense tagged corpus, and then tagged new instances by exploiting the structural information provided by the tree.

Our experimental results have shown that our graphs exhibit hierarchical organisation that can be captured by *HRGs*, in effect providing improved WSD performance compared to flat clustering. Additionally, our comparison against hierarchical agglomerative clustering with average-linkage has shown that *HRGs* perform significantly better than *HAC* when the graphs do not suffer from sparsity (disconnected graphs). The comparison with state-of-the-art sense induction systems has shown that our method yields improvements.

Our future work focuses on using different feature types, e.g. dependency relations, second-order co-occurrences, named entities and others to construct our undirected graphs and then applying HRGs, in order to measure the impact of each feature type on the induced hierarchical structures within a WSD setting. Moreover, following the work in (Clauset et al., 2008), we are also working on using MCMC in order to sample more than one dendrogram at equilibrium, and then combine them to a consensus tree. This consensus tree might be able to express a larger amount of topological features of the initial undirected graph.

Finally in terms of evaluation, our future work also focuses on evaluating *HRGs* using a fine-grained sense inventory, extending the evaluation on the SemEval-2010 WSI task dataset (Manandhar et al., 2010) as well as applying HRGs to other related tasks such as taxonomy learning.

Acknowledgements

This work is supported by the European Commission via the EU FP7 INDECT project, Grant No.218086, Research area: SEC-2007-1.2-01 Intelligent Urban Environment Observation System. The authors would like to thank the anonymous reviewers for their useful comments.

References

Eneko Agirre and Aitor. Soroa. 2007. Semeval-2007 Task 02: Evaluating Word Sense Induction and Discrimination Systems. In *Proceedings of SemEval-2007*, pages 7–12, Prague, Czech Republic.

Eneko Agirre, David Martínez, Oier López de Lacalle, and Aitor Soroa. 2006. Two Graph-based Algorithms for State-of-the-art WSD. In *Proceedings of EMNLP-2006*, pages 585–593, Sydney, Australia.

Chris Biemann. 2006. Chinese Whispers - An Efficient Graph Clustering Algorithm and its Application to Natural Language Processing Problems. In *Proceedings of TextGraphs*, pages 73–80, New York, USA.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117.

Samuel Brody and Mirella Lapata. 2009. Bayesian Word Sense Induction. In *Proceedings of EACL-2009*, pages 103–111, Athens, Greece. ACL.

Aaron Clauset, Cristopher Moore, and Mark E. J. Newman. 2006. Structural Inference of Hierarchies in Networks. In *Proceedings of the ICML-2006 Workshop on Social Network Analysis*, pages 1–13, Pittsburgh, USA.

Aaron Clauset, Cristopher Moore, and Mark E. J. Newman. 2008. Hierarchical Structure and the Prediction of Missing Links in Networks. *Nature*, 453(7191):98–101.

Stijn Dongen. 2000. Performance Criteria for Graph Clustering and Markov Cluster Experiments. Technical report, CWI (Centre for Mathematics and Computer Science), Amsterdam, The Netherlands.

Beate Dorow and Dominic Widdows. 2003. Discovering Corpus-specific Word Senses. In *Proceedings of the EACL-2003*, pages 79–82, Budapest, Hungary.

Ted Dunning. 1993. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19(1):61–74.

Phil Edmonds and Beate Dorow. 2001. Senseval-2: Overview. In *Proceedings of SensEval-2*, pages 1–5, Toulouse, France.

Ioannis P. Klapaftis and Suresh Manandhar. 2008. Word Sense Induction Using Graphs of Collocations. In *Proceedings of ECAI-2008*, pages 298–302, Patras, Greece.

Ioannis P. Klapaftis and Suresh Manandhar. 2010. Taxonomy Learning Using Word Sense Induction. In *Proceedings of NAACL-HLT-2010*, pages 82–90, Los Angeles, California, June. ACL.

Suresh Manandhar, Ioannis P. Klapaftis, Dmitriy Dligach, and Sameer S. Pradhan. 2010. Semeval-2010 Task 14: Word Sense Induction & Disambiguation. In *Proceedings of SemEval-2*, Uppsala, Sweden. ACL.

Rada Mihalcea. 2004. Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 20, Morristown, NJ, USA.

Mark Newman and Gerard Barkema. 1999. *Monte Carlo Methods in Statistical Physics*. Oxford: Clarendon Press, New York, USA.

Zheng-Yu Niu, Dong-Hong Ji, and Chew-Lim Tan. 2007. I2R: Three Systems for Word Sense Discrimination, Chinese Word Sense Disambiguation, and English Word Sense Disambiguation. In *Proceedings of SemEval-2007*, pages 177–182, Prague, Czech Republic.

Patrick Pantel and Dekang Lin. 2003. Automatically Discovering Word Senses. In *Proceedings of NAACL-HLT-2003*, pages 21–22, Morristown, NJ, USA.

Ted Pedersen and Anagha Kulkarni. 2006. Automatic Cluster Stopping With Criterion Functions and the gap Statistic. In *Proceedings of the 2006 Conference of the North American Chapter of the ACL on Human Language Technology*, pages 276–279, Morristown, NJ, USA.

Ted Pedersen. 2007. UMND2 : Senseclusters Applied to the Sense Induction Task of Senseval-4. In *Proceedings of SemEval-2007*, pages 394–397, Prague, Czech Republic.

Daniel Ramage, Anna N. Rafferty, and Christopher D. Manning. 2009. Random Walks for Text Semantic Similarity. In *Proceedings of TextGraphs-4*, Suntec, Singapore, August.

Noam Slonim, Nir Friedman, and Naftali Tishby. 2002. Unsupervised Document Classification Using Sequential Information Maximization. In *SIGIR 2002*, pages 129–136, New York, NY, USA. ACM.

Benjamin Snyder and Martha Palmer. 2004. The English All-words Task. In Rada Mihalcea and Phil Edmonds, editors, *In Proceedings of Senseval-3*, pages 41–43, Barcelona, Spain.

Jean Véronis. 2004. Hyperlex: Lexical Cartography for Information Retrieval. *Computer Speech & Language*, 18(3):223–252.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising Measures of Lexical Distributional Similarity. In *Proceedings of COLING-2004*, pages 10–15, Morristown, NJ, USA.

Dominic Widdows and Beate Dorow. 2002. A Graph Model for Unsupervised Lexical Acquisition. In *Proceedings of Coling-2002*, pages 1–7, Morristown, NJ, USA.