UBC-AS: A Graph Based Unsupervised System for Induction and Classification

Eneko Agirre and Aitor Soroa
IXA NLP Group
UBC
Donostia, Basque Contry
{e.agirre,a.soroa}@si.ehu.es

Abstract

This paper describes a graph-based unsupervised system for induction and classification. The system performs a two stage graph based clustering where a co-occurrence graph is first clustered to compute similarities against contexts. The context similarity matrix is pruned and the resulting associated graph is clustered again by means of a random-walk type algorithm. The system relies on a set of parameters that have been tuned to fit the corpus data. The system has participated in tasks 2 and 13 of the SemEval-2007 competition, on word sense induction and Web people search, respectively, with mixed results.

1 Introduction

This paper describes a graph-based unsupervised system for induction and classification. Given a set of data to be classified, the system first induces the possible clusters and then clusters the data accordingly. The paper is organized as follows. Section 2 gives an description of the general framework of our system. Sections 3 and 4 presents in more detail the implementation of the framework for the Semeval-2007 WEPS task (Artiles et al., 2007) and Semeval-2007 sense induction task (Agirre and Soroa, 2007), respectively. Section 5 presents the results obtained in both tasks, and Section 6 draws some conclusions.

2 A graph based system for unsupervised classification

The system performs a two stage graph based clustering where a co-occurrence graph is first clustered to compute similarities against contexts. The context similarity matrix is pruned and the resulting associated graph is clustered again by means of a random-walk type algorithm. We will see both steps in turn.

First step: calculating hub score vectors

In a first step, and for each entity to be clustered, a graph consisting on context word co-occurrences is built. Vertices in the co-occurrence graph are words and two vertices share an edge whenever they co-occur in the same context. Besides, each edge receives a weight, which indicates how strong the incident vertices relate each other.

As shown in (Véronis, 2004), co-occurrence graphs exhibit the so called small world structure (Watts and Strogatz, 1998) and, thus, they contain highly dense subgraphs which will represent the different clusters the entity may have. For identifying these clusters we have implemented two algorithms based on the notion of centrality of the vertices, where some highly dense vertices, called "hubs", are chosen as representatives of each cluster. The algorithms are the HyperLex algorithm (Véronis, 2004) and the HITS algorithm (Kleinberg, 1999).

Once the hubs are identified, the minimum spanning tree (MST) of the co-occurrence graph is computed. The root elements of the MST are precisely the induced hubs and each vertex of the original graph —and, thus, each word of the corpus— is attached to exactly one of these hubs, at a certain distance. Note that the MST can be considered as a single link clustering over the co-occurrence graph.

The original contexts are then taken one by one and scored according to the MST in the following way: each word in the context receives a set of score vectors, with one score per hub, where all scores are

0 except for the one corresponding to the hub where it is placed[1], which will receive a socre $d(h_i, v)$, which is the distance between the hub h_i and the node representing the word v in the MST. Thus, $d(h_i, v)$ assigns a score of 1 to hubs and the score decreases as the nodes move away from the hub in the MST. As a consequence, each context receives a hub score vector, which is just the sum of the score vectors of all the words in the context.

At this point we can use the hub score vectors to create clusters of contexts, just assigning to each context the hub with maximum score. This process is thoroughly explained in (Agirre et al., 2006). One of the problems of such an approach comes from the tendency of the system to produce a high number of hubs, somehow favouring small micro-clusters over coarse ones. Knowing in advance that the number of clusters in the tasks we will participate in would not be very high, we decided to perform a second stage and re-cluster again the results obtained in the first step, using a different graph-based technique. Re-clustering also gives us the opportunity to feed the system with additional data, as will be explained below.

Second step: clustering via MCL

In this second stage, we compute a square matrix with as many rows/columns as contexts, and where each element represents the relatedness between two contexts, just computing the cosine distance of its (normalized) hub score vectors obtained in the first step. We prune each row in the matrix and keep only the element with maximum values, so that the percentage of the kept elements' sum respect the total is below a given threshold. The resulting matrix M represents the adjacency matrix of a directed weighted graph, where vertices are contexts and edges represent the similarity between them. We can feed the matrix M with external information just by calculating another dissimilarity matrix between contexts and lineally interpolating the matrices with a factor.

Finally, we apply the Markov Clustering (MCL) algorithm (van Dongen, 2000) over the graph M for calculating the final clusters. MCL is a graph-clustering algorithm based on simulation of stochastic flows in graphs, its main idea being that random walks within the graph will tend to stay in the same cluster rather than jump between clusters. MCL has the remarkable property that there is no need to a-priori decide how many clusters it must find. However, it has some parameters which will influence the granularity of the clusters.

In fact, the behavior of the whole process relies on a number of parameters, which can be divided in several groups:

- Parameters for calculating the hubs
- Parameters for merging the hubs information with external information in the matrix M (α)
- The threshold for pruning the graph (δ)
- Parameters of the MCL algorithm (I, inflation parameter)

In sections 3 and 4 we describe the parameters we actually used for the final experiments, as well as how the tuning of these parameters has been performed for the two tasks.

3 Web People Search task

In this section we will explain in more detail how we implemented the general schema described in the previous section to the "Web People Search" task (Artiles et al., 2007). The task consist on disambiguating person names in a web searching scenario. The input consists on web pages retrieved from a web searching engine using person names as a query. The aim is to determine how many referents (people with the same name) exist for that person name, and classify each document with its corresponding referent. There is a train set consisting on 49 names and 100 documents per name. The test setting consist on 30 unrelated names, with 100 document per name. The evaluation is performed following the "purity" and "inverse purity" measures. Roughly speaking, purity measures how many classes they are in each cluster (like the precision measure). If a cluster fits into one class, the purity equals to 1. On the other side, inverse purity measures how many clusters they are in each class (recall). The final figure is obtained by combining purity and inverse purity by means of the standard F-Measure with $\alpha = 0.5$.

The parameters of the system were tuned using the train part of the corpus as a development set. As usual, the parameters that yielded best results were used on the test part.

[1] Note that each word will be attached to exactly one hub in the MST.

We first apply a home-made wrapper over the html files for retrieving the text chunks of the pages, which is usually mixed with html tags, javascript code, etc. The text is split into sentences and parsed using the FreeLing parser (Atserias et al., 2006). Only the lemmas of nouns are retained. We filter the nouns and keep only back those words whose frequency, according to the British National Corpus, is greater than 4. Next, we search for the person name across the sentences, and when such a sentence is found we build a context consisting on its four predecessor and four successors, i.e., contexts consists on 9 sentences. At the end, each document is represented as a set of contexts containing the person name. Finally, the person names are removed from the contexts.

For inducing the hubs we apply the HyperLex algorithm (Véronis, 2004). Then, the MST is calculated and every context is assigned with a hub score vector. We calculate the hub score vector of the whole document by averaging the score vectors of its contexts. The M matrix of pairwise similarities between documents is then computed and pruned with a threshold of 0.2, as described in section 2.

We feed the system with additional data about the topology of the pages over the web. For each document d_i to be classified we retrieve the set of documents P_i which link to d_i. We use the publicly available API for Microsoft Search. Then, for each pair of documents d_i and d_j we calculate the number of overlapping documents linking to them, i.e., $l_{ij} = \#\{P_i \cap P_j\}$ with the intuition that, the more pages point to the two documents, the more probably is that they both refer to the same person. The resulting matrix, M^L is combined with the original matrix M to give a final matrix M', by means of a linear interpolation with factor of 0.2, i.e. $M' = 0.2M + 0.8M_L$. Finally, the MCL algorithm is run over M' with an inflation parameter of 5.

4 Word Sense Induction and Discrimination task

The goal of this task is to allow for comparison across sense-induction and discrimination systems, and also to compare these systems to other supervised and knowledge-based systems. The input consist on 100 target words (65 verbs and 35 nouns), each target word having a set of contexts where the word appears. The goal is to automatically induce the senses each word has, and cluster the contexts accordingly. Two evaluation measures are provided: and unsupervised evaluation (FScore measure) and a supervised evaluation, where the organizers automatically map the induced clusters onto senses. See (Agirre and Soroa, 2007) for more details.

In order to improve the overall performance, we have clustered the 35 nouns and the 65 verbs separately. In the case of nouns, we have filtered the original contexts and kept only noun lemmas, whereas for verbs lemmas of nouns, verbs and adjectives were hold.

The algorithm for inducing the hubs is also different among nouns and verbs. Nouns hubs are induced with the usual HyperLex algorithm (just like in section 3) but for identifying verb hubs we used the HITS algorithm (Kleinberg, 1999), based on preliminary experiments.

The co-occurrence relatedness is also measured differently for verbs: instead of using the original conditional probabilities, the χ^2 measure between words is used. The reason behind is that conditional probabilities, as used in (Véronis, 2004), perform poorly in presence of words which occur in nearly all contexts, giving them an extraordinary high weight in the graph. Very few nouns happen to occur in many contexts, but they are verbs which certainly do (be, use, etc). On the other hand, χ^2 measures to what extent the observed co-occurrences diverge from those expected by chance, so weights of edges incident with very common, non-informant words will be low.

Parameter tuning for both nouns and verbs was performed over the senseval-3 testbed, and the best parameter combination were applied over the sense induction corpus. However, there is a factor we have taken into account in tuning directly over the sense induction corpus, i.e., that the granularity —and thus the number of classes— of senses in OntoNotes (the inventory used in the gold standard) is considerably coarser than in senseval-3. Therefore, we have manually tuned the inflation parameter of the MCL algorithm in order to achieve numbers of clusters between 1 and 4.

A threshold of 0.6 was used when pruning the dissimilarity matrix M for both nouns and verbs. We have tried to feed the system with additional data

System	All	Nouns	Verbs
Best	78.7	80.8	76.3
Worst	56.1	62.3	45.1
Average	65.4	69.0	61.4
UBC-AS	78.7	80.8	76.3

Table 1: Results of Semeval-2007 Task 2. Unsupervised evaluation (FScore).

System	All	Nouns	Verbs
Best	81.6	86.8	76.2
Worst	77.1	80.5	73.3
Average	79.1	82.8	75.0
UBC-AS	78.5	80.7	76.0

Table 2: Results of Semeval-2007 Task 2. Supervised evaluation as recall.

(mostly local and domain features of the context words) but, although the system performed slightly better, we decided that the little gain (which probably was not statistically significant) was no worth the effort.

5 Results

Table 1 shows the results of the unsupervised evaluation in task 2, where our system got the best results in this setting. Table 2 shows the supervised evaluation on the same task, where our system got a ranking of 4, performing slightly worse than the average of the systems.

In Table 3 we can see the results of Semeval-2007 Task 13. As can be seen, our system didn't manage to capture the structure of the corpus, and it got the worst result, far below the average of the systems.

6 Conclusions

We have presented graph-based unsupervised system for induction and classification. The system performs a two stage graph based clustering where a co-occurrence graph is first clustered to compute similarities against contexts. The context similarity matrix is pruned and the resulting associated graph is clustered again by means of a random-walk type algorithm. The system has participated in tasks 2 and 13 of the SemEval-2007 competition, on word sense induction and Web people search, respectively, with mixed results. We did not have time to perform an in-depth analysis of the reasons causing such a different performance. One of the reasons for the failure in the WePS task could be the fact that we

System	$F_{\alpha=0.5}$
Best	78.0
Worst	40.0
Average	60.0
UBC-AS	40.0

Table 3: Results of Semeval-2007 Task 13

were first-comers, with very little time to develop the system, and we used a very basic and coarse preprocessing of the HTML files. Another factor could be that we intentionally made our clustering algorithm return few clusters. We were mislead by the training data provided, as the final test data had more classes on average.

Acknowledgements

This work has been partially funded by the Spanish education ministry (project KNOW) and by the Basque government (project SAIOTEK S-PE06UN31).

References

E. Agirre and A. Soroa. 2007. Semeval-2007 task 2: Evaluating word sense induction and discrimination systems. In *Proceedings of Semeval-2007, Association for Computational Linguistics.*

E. Agirre, D. Martínez, O. López de Lacalle, and A. Soroa. 2006. Two graph-based algorithms for state-of-the-art wsd. In *Proceedings of EMNLP 2006*, pages 585–593. Association for Computational Linguistics, July.

J. Artiles, J. Gonzalo, and S. Sekine. 2007. The semeval 2007 WePS evaluation: Establishing a benchmark for the web people search task. In *Proceedings of Semeval-2007, Association for Computational Linguistics.*

J. Atserias, B. Casas, E. Comelles, M. González, L. Padró, and M. Padró. 2006. Freeling 1.3: Syntactic and semantic services in an open-source NLP library. In *Proceedings of LREC'06*, pages 48–55.

Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632.

Stijn van Dongen. 2000. A cluster algorithm for graphs. Technical Report INS-R0010, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, May.

J. Véronis. 2004. Hyperlex: lexical cartography for information retrieval. *Computer Speech & Language*, 18(3):223–252.

D. J. Watts and S. H. Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June.