Cristina Bosco, Sara Tonelli and Fabio Massimo Zanzotto (dir.)

Proceedings of the Second Italian Conference on Computational Linguistics CLiC-it 2015
3-4 December 2015, Trento

Accademia University Press

Word Sense Discrimination: A gangplank algorithm

Flavio Massimiliano Cecchini and Elisabetta Fersini

DOI: 10.4000/books.aaccademia.1336
Publisher: Accademia University Press
Place of publication: Torino
Year of publication: 2015
Published on OpenEdition Books: 11 November 2016
Serie: Collana dell'Associazione Italiana di Linguistica Computazionale
Electronic ISBN: 9788899200008

Word Sense Discrimination: A gangplank algorithm

Flavio Massimiliano Cecchini, Elisabetta Fersini
Università degli Studi di Milano-Bicocca
Viale Sarca 336, Ed. U14, 20126 Milano
{flavio.cecchini,fersiniel}@disco.unimib.it

Abstract

English. In this paper we present an unsupervised, graph-based approach for Word Sense Discrimination. Given a set of text sentences, a word co-occurrence graph is derived and a distance based on Jaccard index is defined on it; subsequently, the new distance is used to cluster the neighbour nodes of ambiguous terms using the concept of "gangplanks" as edges that separate denser regions ("islands") in the graph. The proposed approach has been evaluated on a real data set, showing promising performance in Word Sense Discrimination.

Italiano. *L'obiettivo di questo articolo è descrivere un approccio di clustering non supervisionato e basato su grafi per individuare e discriminare i differenti sensi che un termine può assumere all'interno di un testo. Partendo da un grafo di cooccorrenze, vi definiamo una distanza fra nodi e applichiamo un algoritmo basato sulle "passerelle", cioè archi che separano regioni dense ("isole") all'interno del grafo. Discutiamo i risultati ottenuti su un insieme di dati composto da tweet.*

1 Introduction

Word Sense Disambiguation is a challenging research task in Computational Linguistics and Natural Language Processing. The main reasons behind the difficulties of this task are ambiguity and arbitrariness of human language: just depending on its context, the same term can assume different interpretations, or senses, in an unpredictable manner. In the last decade, three main research directions have been investigated (Navigli, 2009; Navigli, 2012): 1) supervised (Zhong and Ng, 2010; Mihalcea and Faruque, 2004), 2) knowledge-based (Navigli and Ponzetto, 2012; Schmitz et al., 2012) and 3) unsupervised Word Sense Disambiguation (Dorow and Widdows, 2003; Véronis, 2004), where the last approach is better defined as "induction" or "discrimination". In this paper we focus on the automatic discovery of senses from raw text, by pursuing an unsupervised Word Sense Discrimination paradigm. We are interested in the development of a method that can be generally independent from the register or the linguistic well-formedness of a text document, and, given an adequate pre-processing step, from language. Among the many unsupervised research directions, i.e. context clustering (Schütze, 1998), word clustering (Lin, 1998), probabilistic clustering (Brody and Lapata, 2009) and co-occurrence graph clustering (Widdows and Dorow, 2002) , we committed to the last one, based on the assumption that word co-occurrence graphs can reveal local structural properties tied to the different senses a word might assume in different contexts.

Given a global word co-occurrence graph, the main goal is to exploit the subgraph induced by the neighbourhood of the word to be disambiguated (a "word cloud"). There, we define separator edges ("gangplanks") and use them as the means to cluster the word cloud: the fundamental assumption is that, in the end, every cluster will correspond to a different sense of the word.

The paper is organized as follows. In section 2 we explain how we build our co-occurrence graph and word clouds by means of a weighted Jaccard distance. In section 3 we describe the gangplank algorithm. In section 4 we present the algorithm's results on our data set and their evaluation. In section 5 we give a brief overview on related work and section 6 presents some short conclusions.

2 Building the word graphs

Given a set of sentences, we derive a global co-occurrence word graph. It is a weighted, undirected graph where each node corresponds to a word (token) and there is an edge between two nodes if and only if the corresponding words co-occur in the same sentence. Edge weights are the respective frequencies of such co-occurrences. It has been shown (i Cancho and Solé, 2001) that a word graph like this tends to behave like a small-world, scale-free graph (Watts and Strogatz, 1998). In short, the graph is very cohesive and its cohesion hinges on few nodes from which nearly every other node can be reached. A similar structure can be quite difficult to handle for our purposes, since on the one hand the largest part of the graph tends to behave as a single, inextricable unit, and on the other hand the graph collapses in a myriad connected components if the hub nodes are removed: we are interested in a clustering between the two extremes. To mitigate this problem, a word filtering step is performed. Stopwords and irrelevant word classes (based on part-of-speech tagging), as e.g. adverbs and adjectives, are removed. Only nouns and verbs are retained.

2.1 A weighted Jaccard distance

Given the previously outlined word graph, we introduce a graph-based distance between nodes derived from Jaccard index that will be enclosed in our clustering algorithm. Given a graph G, each neighbourhood of a node w in G is treated as a multiset[1], where its elements correspond to the neighbour nodes of w and their multiplicity is the weight of the edge that connects them to w, i.e. the number of times they co-occur with w. We have defined the "automultiplicity" of w in this multiset as the greatest weight between all such edges. Given two multisets A and B, now we can define the weighted Jaccard distance as

$$1 - \frac{|A \cap B|}{|A \cup B|},$$

where the intersection is the multiset with the least multiplicity for each element of both (possibly 0, so not counting it) and the union is the multiset with the greatest multiplicity for each element of both. The cardinality of a multiset is the sum of all the multiplicities of its elements. The weighted Jaccard distance provides a measure of how much the contexts of two words overlap, taking into account the importance of each context by means of the weights. A distance of 1 means that contexts do not overlap, and on the contrary a distance of 0 implies a complete overlap. The weighted Jaccard distance can of course be expanded to neighbourhoods of depth greater than 1: for increasing depths, we would take into account contexts of neighbour words, contexts of contexts, and so on. This means that the greater the depth, the less significant the Jaccard distance becomes. It can be shown that, for depth d, any two elements have Jaccard distance (strictly) less than 1 if and only if the shortest path connecting them consists of at most $2d$ edges. This lemma will be used in the next section.

2.2 Word clouds

Given a word w of interest, we extract from G the subgraph G_w induced by the open neighbourhood[2] of w, originating a "word cloud". We again perform word filtering and remove redundant words, this time using principal component analysis, retaining just words that contribute the most to the first, most important component, and considering the corresponding subgraph of G_w (we will denote it by the same notation).

On it, we can define a local weighted Jaccard distance, as explained before. This allows us the transition from the word (sub)graph to a word metric space. From the metric space we build again a weighted, undirected "distance graph" D_w, where two nodes are connected by an edge if and only if their weighted Jaccard distance is strictly less than 1, and weights are the distances between words. As noted at the end of section 2.1, this operation practically coincides with the expansion of G_w where we add edges between nodes that are 2 steps away from each other and reassign a weight corresponding to distance to each edge.

3 The gangplank clustering algorithm

3.1 The algorithm

Our objective is a clustering of D_w that maximizes intra-cluster connections and minimizes inter-cluster connections. As explained in Section 2, our assumption is that clusters of a word cloud

[1] A multiset is a set where an element can recur more than once, and can be defined as a set of couples of the type (element, multiplicity) (Aigner, 2012).

[2] In our case, we consider neighbourhoods of degree 1.

will define different senses, implicitly identified with the clusters themselves.

Our approach focuses on edges. We define an edge e in D_w connecting two nodes u and v to be a gangplank if its weight is strictly greater than the mean of the weights of the edges departing from both its ends u and v: if this happens, then edge e is keeping distant the two local graph's "halves" it connects (the neighbourhood of u excluding v and viceversa). In other words, the two aforementioned halves on both sides of e, seen as different subgraphs of D_w, are on their own more tightly connected regions than the subgraph induced by the union of u's and v's neighbourhoods (and thus including e) would be. To each node v we can assign the number $g(v)$ of gangplanks going out from it; $g(v)$ will be comprised between 0 and $\deg(v)$. We also define the ratio $\gamma(v) = \frac{g(v)}{\deg(v)}$. The smaller $\gamma(v)$, the more we deem v to be in the middle of a tightly connected area, or island.

Our clustering algorithm doesn't set a predetermined number of clusters. It starts by ranking each node v by the ratio $\gamma(v)$ and takes the node with the smallest γ as the seed of the first cluster K. We start then a cycle of expansion and reduction steps. In the expansion step, we add all the neighbours of K to K. Then, in the reduction step, we begin discarding from K all the nodes whose connections are undermining the homogeneous nature of cluster K. More precisely, we rank the nodes in K by the ratio $\gamma_K(u) = \frac{g_K(u)}{\deg_K(u)}$, where $g_K(u)$ and $\deg_K(u)$ are defined as $g(u)$ and $\deg(u)$, but with respect to the subgraph of D_w induced by K. Then, we remove from K the node with the greatest non-zero γ_K, if there is any. Thereafter we update γ_K for each remaining node in K and again remove the node with the greatest non-zero ratio. We continue the reduction step until we can no longer remove any node, and hence no gangplanks are left in cluster K. The cluster's seed will never be removed. Once the reduction step has finished, the expansion and reduction step is repeated, this time ignoring any previously discarded node. The cycle continues until no further expansion is possible. At this point we have obtained the first cluster. Now, we select the yet unclustered node with greatest γ in D_w and start a new cycle of expansion and reduction steps for the corresponding new cluster, and so on, until every node has been clustered.

In the end, we'll obtain a clustering of D_w. However, many clusters will often consist of just one node: these are nodes between more tightly connected areas, which we would like to assign to bigger clusters to avoid dispersion. To this end, we set m_w as the minimum allowed cluster size: m_w is the length of the shortest (filtered) sentence where w appears or 2, whichever is greater. This choice of m_w is motivated by the fact that, in the graph, every sentence forms a clique that we have to consider as a possible cluster. All the clusters whose size is less than m_w are post-processed and their elements reassigned to one of the bigger clusters. Again, we rank these remaining nodes by γ and, starting from the node v with the smallest ratio (the less "noisy" node) and going up, we assign v to the cluster $K_m = \arg\min_K \gamma_K(v)$ (the cluster with less relative gangplank connections to v), until finally all nodes have been clustered. If two or more K_m are eligible, the biggest one is preferred.

A pseudo-code of the proposed gangplank clustering algorithm is reported below.

Algorithm 1 Gangplank clustering algorithm

$\mathcal{K} = \{\}$ ▷ The set of future clusters
$\mathcal{V} = E_{D_w}$ ▷ The set of nodes in D_w
$\mathfrak{s} = \{\}$ ▷ Nodes to be assigned in second step
while $\mathcal{V} \neq \emptyset$ **do**
  $v = \arg\min_{u \in \mathcal{V}} \gamma(u)$
  $K = \{v\}$ ▷ The new, seeded cluster
  $\mathfrak{n} = \{\}$ ▷ Discarded, noisy nodes
  $\mathcal{N} = \bigcup_{u \in K} N_{D_w}(u) \backslash \mathfrak{n}$ ▷ Neighbours of K
  while $\mathcal{N} \neq \mathfrak{n}$ **do**
    $K = K \cup \mathcal{N}$
    while $\exists u \in K \backslash \{v\} : \gamma_K(u) \neq 0$ **do**
      $u = \arg\max_{t \in K} \gamma_K(t)$
      $K = K \backslash \{u\}$
      $\mathfrak{n} = \mathfrak{n} \cup \{u\}$
    end while
    $\mathcal{N} = \bigcup_{u \in K} N_{D_w}(u) \backslash \mathfrak{n}$
  end while
  if $|K| \geq m_w$ **then**
    $\mathcal{K} = \mathcal{K} \cup \{K\}$ ▷ Add the new cluster
  else
    $\mathfrak{s} = \mathfrak{s} \cup K$
  end if
  $\mathcal{V} = \mathcal{V} \backslash K$ ▷ Remove clustered nodes
end while
while $\mathfrak{s} \neq \emptyset$ **do**
  $s = \arg\max_{r \in \mathfrak{s}} \gamma(r)$
  $K_s = \arg\min_{K \in \mathcal{K}} \gamma_{K \cup s}(s)$
  $K_s = K_s \cup \{s\}$ ▷ Expand the cluster
  $\mathfrak{s} = \mathfrak{s} \backslash \{s\}$
end while
return $\mathcal{K}$

3.2 The labelling step

Once we have obtained a given number of cluster-senses relative to the chosen term, we adopt a ma-

Keywords	Tagged tweets	No. of senses	Most common senses ($\geq 10\%$)
blizzard	463	23	snowstorm 43%, video game company 37%
caterpillar	467	23	CAT machines 30%, animal 24%, The Very Hungry Caterpillar 17%, CAT company 16%
england	474	11	country (UK) 65%, national football team 10%, New England (USA) 10%
ford	558	12	Harrison Ford 40%, Ford vehicles 30%, Tom Ford (fashion designer) 25%
india	474	5	country 50%, national cricket team 48%
jfk	474	13	New York airport 61%, John Fitzgerald Kennedy 33%
mcdonald	425	47	McDonald's (restaurants) 38% , McDonald's (company) 31%
mars	440	24	planet 66%, Bruno Mars 17%
milan	594	41	Milano (Italy) 58%, A.C. Milan football team 24%
pitbull	440	7	rapper 49%, dog breed 48%
venice	482	9	Venezia (Italy) 55%, Venice beach (California) 42%

Table 1: Keywords and entities.

jority voting mechanism to label each of the term's occurrences in the original sentences. For each sentence where the disambiguated term appears, we compute the Jaccard distance between the set of the sentence's filtered words and each cluster. Then, we assign the term a label referring to the nearest cluster. It is possible that not every cluster will be assigned to a term's occurrence; these are "weak" clusters that are maybe either too insignificant or too fine-grained.

4 Evaluation on tweets

4.1 Data and tagging

In order to evaluate the performance of the proposed approach from a quantitative point of view, a benchmark data set should be employed. However, data sets like SemEval are not ideal, since they don't present enough samples for each word, therefore yielding a sparse and most often unweighted (i.e. all weights are equal to 1) graph. For these reasons, we created an *ad hoc* data set consisting of 5291 tweets in English, downloaded from Twitter on a single day using eleven different keywords; the data set is summarized in Table 1. Keywords were chosen to be common nouns that may possess many different senses, and were the target of our word sense discrimination. To have a basis for evaluation, we manually tagged keywords occurring in the tweets, in order to create a ground truth.

4.2 Evaluation and results

We evaluated the coherence of our clustering and subsequent word labelling with respect to the data set's "true" labels. For each keyword's cluster-derived labelling, we compare that label's context (i.e. all the words in the corresponding filtered sentences) to all the true labels' contexts by means of the Jaccard distance. We then identify the cluster-derived label with its closest true label. We end up with a mapping σ going from some of our clusters to the true labels. To measure the quality of the proposed solution, accuracy's performance has been computed for each disambiguated term, as reported in Table 2. The global accuracy score we obtained is $62{,}56\%$. It could be argued that accuracies are worse whenever the keyword is not polarized on two senses, as is the case for *caterpillar* or *mcdonald*, with many possible senses and no two of them covering together more than 90% of all senses. This could happen because in this case the word cloud will be fractured in many subunits, the gangplanks algorithm will tendentially split them even more and so surfacing labels will be sparse and somewhat inorganic.

For confrontation, we also ran the *Chinese Whispers* algorithm (Biemann, 2006), which uses a simplified form of Markov clustering, on our graphs, obtaining a global accuracy score of $60{,}1\%$ with a mean number of just $2{,}27$ clusters per keyword (a behaviour mentioned in Section 2). Local scores are shown in Table 2. Accuracies are only better when senses are strongly polarized, e.g. for *pitbull* and *england*. In the latter case, just one cluster is found, so the algorithm's accuracy is the same as the percentage of occurrences of the main sense.

5 Related work

An approach similar to ours, at least in the initial graph construction, can be found in (Véronis, 2004). The weights we put on edges substantially differ from his, but, most markedly, Véronis wants to span some trees from some hub nodes in each word cloud. In other words, Véronis's algorithm is more hierarchical in nature, where ours is more aggregative. Similar considerations can also be made for (Hope and Keller, 2013).

6 Conclusions

The main challenge we encountered for our word sense discrimination algorithm was the difficulty of handling a small-world graph. Apart from that, we have to notice that word clustering just rep-

	blizzard	caterpillar	england	ford	india	jfk	mars	mcdonald	milan	pitbull	venice
Labelling accuracy	**49,9**	42,0	50	**87,8**	**82,7**	**67,5**	**75,5**	40	53,2	64,5	**71,0**
No. of clusters	38	20	46	17	28	14	9	15	32	55	34
No. of labels	13	11	7	6	5	4	5	9	16	5	7
Chinese Whispers	43,0	**43,7**	**65,4**	80,1	63,1	61,2	66,4	**40,7**	**58,2**	**81,1**	55,0

Table 2: Local labelling accuracies for the gangplank and Chinese Whispers clustering algorithm. Accuracies in %. *No. of labels* represent how many labels effectively appear in labelled tweets.

resents the last step of a process that starts with pre-processing and tokenization of a text, which are both mostly of supervised nature. Our future goals will be to investigate the relations between text pre-processing and clustering results, and how to render the whole process completely unsupervised.

References

Martin Aigner. 2012. *Combinatorial theory*, volume 234. Springer Science & Business Media.

Chris Biemann. 2006. Chinese whispers: an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the first workshop on graph based methods for natural language processing*, pages 73–80. Association for Computational Linguistics.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 103–111. Association for Computational Linguistics.

Beate Dorow and Dominic Widdows. 2003. Discovering corpus-specific word senses. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 79–82. Association for Computational Linguistics.

David Hope and Bill Keller. 2013. Maxmax: a graph-based soft clustering algorithm applied to word sense induction. In *Computational Linguistics and Intelligent Text Processing*, pages 368–381. Springer.

Ramon Ferrer i Cancho and Richard V Solé. 2001. The small world of human language. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1482):2261–2265.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics.

Rada Mihalcea and Ehsanul Faruque. 2004. Senselearner: Minimally supervised word sense disambiguation for all words in open text. In *Proceedings of ACL/SIGLEX Senseval*, volume 3, pages 155–158.

Roberto Navigli and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.

Roberto Navigli. 2012. A quick tour of word sense disambiguation, induction and related approaches. In *SOFSEM 2012: Theory and practice of computer science*, pages 115–129. Springer.

Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534. Association for Computational Linguistics.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123.

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

Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of small-worldnetworks. *nature*, 393(6684):440–442.

Dominic Widdows and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Zhi Zhong and Hwee Tou Ng. 2010. It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, pages 78–83. Association for Computational Linguistics.