Clustering and Classification in Text Collections Using Graph Modularity

Grigory Pivovarov GBPIVO@MS2.INR.AC.RU
Institute for Nuclear Research
Russian Academy of Sciences
Moscow, 117312, Russia

Sergei Trunov TRUNOV7@GMAIL.COM
Institute for Institutional Analyses
Higher School of Economics
Moscow, 109028, Russia

Editor: Not Known

Abstract

A new fast algorithm for clustering and classification of large collections of text documents is introduced. The new algorithm employs the bipartite graph that realizes the word-document matrix of the collection. Namely, the modularity of the bipartite graph is used as the optimization functional. Experiments performed with the new algorithm on a number of text collections had shown a competitive quality of the clustering (classification), and a record-breaking speed.

Keywords: Text Clustering, Text Classification, Modularity

1. Introduction

We explore a possibility of clustering (or classification) of documents. Clustering and classification are methods for information retrieval (for a recent review see Berry (2003)). The possibility we explore consists in combining two ideas considered previously.

The first idea is co-clustering (Dhillon, 2001), (Zha et al., 2001). Co-clustering clusters along with the documents the words used in the documents. As an outcome, clusters of documents are generated along with corresponding clusters of words. This approach features the following advantages: Clusters of words generated as a byproduct of the approach can be used for interpretation of the clusters of documents; In the classification tasks, it is possible to use in the training sets separate words along with documents. The standard algorithm used within the co-clustering approach to reach the result is the spectral clustering (von Luxburg, 2007). (Computationally, spectral clustering finds eigenvectors of the graph laplacian. With a number of tricks, the eigenvectors are used for clustering.)

The second idea is modularity (Newman, 2006). Modularity is a class of optimization functionals introduced in the studies of graph clustering. Let us compare modularity to other optimization functionals appearing within the widely used approach to clustering based on generative models (Zhong and Ghosh, 2005). These optimization functionals are various "distances" between the data and the model. Optimization consists in finding parameters of the model yielding the minimal distance. In contrast, the modularity is optimal when a "distance" between the data and a null-model is maximal. The null-model is a key notion for the modularity idea. The null-model models the data without structure (the most random data). Concretely, modularity is defined as follows. A functional on graph partitionings is picked out. Modularity is an additive or multiplicative difference of the value the functional takes on the graph under study and the mean value it takes on the null-model. From the above comparison we conclude that using modularity is in a way less demanding than using generative models, because it is easier to model randomness than specific data. Comparison of modularity-based approaches and generative models approaches is attempted in (Karrer and Newman, 2011), (Bickel and Chen, 2009).

Modularity has been used in text clustering (Grineva et al., 2009). In this attempt, a dense weighted graph has been clustered. The nodes of the graph are the documents, all the documents are potentially linked to one another, the edges have weights characterizing similarity of the linked documents.

In this paper we apply the modularity of (Newman, 2006) to the bipartite word-document graph. This is a very sparse bipartite graph G whose nodes are documents and words, and edges are between documents and words contained in them. The sparsity of G makes our approach practical.

More technically, our work is based on two facts. First, the modularity can be optimized with fast and efficient algorithms (Blondel et al., 2008) that have complexity proportional to the number of links. (Here we point out that we independently developed an algorithm similar to the so called Louvain algorithm (Blondel et al., 2008) before the paper (Blondel et al., 2008) appeared. We had used it in 2007 to cluster the citation graph of the papers from `http://arxiv.org`. The results of this clustering are accessible via `http://xstructure.inr.ac.ru`.) Second, the density of the graphs in our experiments was in the range from 0.0015 to 0.006 [1]. For such graphs, $|E| \propto |V| \log |V|$, where $|E|$ is the number of edges and $|V|$ is the number of vertexes of the graph. Also, the number of vertexes in our graph equals approximately the number of documents in the collection.

We conclude that in the case under consideration the linearity of the algorithm in the number of edges of the graph almost implies the linearity in the number of documents. In this way we obtain a very fast algorithm. It allows one to cluster (classify) tens of millions of documents in a few hours with a typical computer hardware. Presently, a clustering problem is considered to be a "large scale" if it involves up to 10^5 documents (Vries and Geva, 2010). With our algorithm, it is possible to raise this bar at least up to 10^7 documents.

The paper is organised as follows. In the next section, we outline the algorithm. In the third section, we present the results of experiments applying the new algorithm to various text collections. In the cocluding section, we briefly summarise our achievements.

2. The Algorithm

2.1 Clustering: The Basic Algorithm

In this section, we outline the algorithm we used to maximize the modularity of the bipartite graph G modeling a collection of documents (its vertexes are documents and words of the

[1] The density of a graph is defined as $2|E|/(|V|^2 - 1)$, where $|E|$ is the number of edges and $|V|$ is the number of the vertexes of the graph.

collection; an edge appears between a document and a word if the latter is contained in the former).

The modularity $Q(P, G)$ is a functional defined on the set $\{P\}$ whose members are partitions of the set of vertexes of the graph G (Newman, 2006).

As discussed above the modularity is a difference between the fraction of the edges inside the clusters for the graph under consideration and for the null model. For example, for a simple (unweighted and undirected) graph, the value it takes on a particular partition is

$$Q(P, G) = \sum_{i=1}^{N} \Big(\frac{l_i}{L} - \frac{D_i^2}{4L^2} \Big), \tag{1}$$

where the summation runs over the clusters of the partition, N is the number of clusters, l_i is the number of edges inside the ith cluster, L is the number of graph edges, and D_i is the sum of degrees of vertexes inside the cluster i.

Modularity can be used to determine an invariant of the graph G—the partition P that gives the modularity its maximal value. Generally, computing this invariant is an NP-complete problem (Brandes et al., 2006). There is a number of algorithms for computing an approximation to this invariant (Fortunato, 2010).

For our particular case, where the graph under consideration is a bipartite one, the null model should be modified allowing for the edges to appear randomly only between the two parts of the graph. Accordingly, equation (1) is transformed as follows (Barber, 2007):

$$Q_{bp}(P, G) = \sum_{i=1}^{N} \Big(\frac{l_i}{L} - \frac{D_i^1 D_i^2}{L^2} \Big), \tag{2}$$

where D_i^1 (D_i^2) is the sum of degrees of vertexes inside the first (second) part of the ith cluster, and L is the number of graph edges.

Our algorithm is based on a use of an operation T_P to be defined below. It acts on any partition P' that can be obtained from the partition P involved in its definition by a coarsening, $P' \geq P$ (this means that the subsets of P' can be obtained by merging some subsets of P). The outcome of T_P acting on P' is a new partition whose modularity is not less than the one of P': $Q(T_P P') \geq Q(P')$. (Here and below we omit the second argument of $Q(P, G)$ because the graph G is fixed.) This is the basic property of the operation T_P: its action "improves" the partition. The definition of T_P does not use any specific property of the quality functional Q, and can be given for any particular choice of the latter. We stress that T_P depends on the particular choice of the quality functional Q.

To define T_P, we introduce an arbitrary numbering of the elements v, $v \in P$ (the notation v originates from the most refined partition of G whose members are separate vertexes). After that, instead of the set of elements v of the partition P we deal with the set of their numbers, $v \in \{1, 2, \ldots, |P|\}$.

The next step is to introduce coordinates on the set of $P' \geq P$. Each P' can be considered as a point in a space with $|P|$ discrete coordinates; each coordinate takes an integer value from 1 to $|P|$. Indeed, each P' defines an equivalence relation on the numbers: $v' \sim v$ if v and v' belong to the same subset of P'. The vth coordinate of P' can be defined as follows:

$$P'_v = \max_{v' \sim v} v' \tag{3}$$

So, by this formula, any set v is mapped to the set inside the same cluster of $|P'|$ with the maximal number. Inversely, any point $(x_1, \ldots, x_{|P|})$ of the discrete space $\{1, \ldots, |P|\}^{|P|}$ can be interpreted as a partition P' whose members are obtained by merging the subsets of P whose coordinates $x_{v \in P}$ coincide.

Now the functional Q can be considered as a function of $|P|$ discrete arguments:

$$Q(P') = Q(P'_1, ..., P'_{|P|}); \tag{4}$$

each argument runs from 1 to $|P|$. We are looking for the maximum of this function.

To approximate the maximum, we can take any starting P' and use the discrete cyclic coordinate descent method (Luenberger, 1973) to obtain a point $T_P P'$ improving the partition P', $Q(T_P P') \geq Q(P')$. This concludes our definition of the operation T_P.

The operation T_P can be used to describe the previously introduced Louvain algorithm (Blondel et al., 2008). Indeed, the Louvain algorithm yields the partition ending the sequence of partitions $P_n = T_{P_{n-1}} P_{n-1}$ that starts from the most refined partition P_0 whose members are the vertexes.

Experimenting with classification of text collections, we have found that it is advantageous to use another sequence of partitions approaching the maximum, $P_n = T_{P_0} T_{P_{n-1}} P_{n-1}$. So, we start with the most refined partition P_0. The first step of the process yields $P_1 = T_{P_0} P_0$ (this is the case because $T_P^2 = T_P$ for any P), the second, $P_2 = T_{P_0} T_{P_1} P_1$, and so on. The process stops when its next step yields a partition whose modularity coincides with the one obtained on the previous step.

Comparing this algorithm to the Louvain algorithm we point out that, in contrast to the Louvain algorithm, each step of our algorithm does not necessarily coarsen the partition, i.e. our P_n is not always more coarse than P_{n-1}. The results we obtain appear to be more accurate (in the sense to be defined latter on) than the ones obtained with the Louvain algorithm.

This concludes the general description of our algorithm.

2.2 Clustering: Finetuning

Handmade classifications of large text collections have a number of classification levels. For example, the online arxive `arxiv.org` has three classification levels (e.g. Physics—Condensed Matter—Superconductivity), and the huge collection of web sites `dmoz.org` has more than three classification levels (the actual number of levels depends on the subject field). Such levels are not described with the above approach employing the modularity function.

A handle on this is provided by the parametric modularity introduced in (Reichardt and Bornholdt, 2006),(Lambiotte, 2010). It is defined as follows:

$$Q_{bp}(P, G, \lambda) = \sum_{i=1}^{N} \Big(\frac{l_i}{L} - \lambda \frac{D_i^1 D_i^2}{L^2} \Big), \tag{5}$$

where an extra real positive parameter λ had appeared.

Let us give an example clarifying the meaning of the new parameter λ. Consider a graph G_K which consists of K copies of the graph G. Let the modularity of G reach its maximum value on the partition P_{max}. This G_K gives a simple model of a graph with two classification levels naturally present: the upper level P_2 has as its classes the separate copies of G, while the ground level P_1 of the classification subdivides each copy of G on the subgraphs participating in P_{max}. With this notations, $Q(G_K, P_1) = Q_+(G, P_{max}) - Q_-(G, P_{max})/K$, where Q_+ (Q_-) denotes the first (second) term in the right hand side of (2). Also, $Q(G_K, P_2) = 1 - 1/K$. Because $Q_\pm < 1$, at large K, $Q(G_K, P_2) > Q(G_K, P_1)$. We conclude that in this case the modularity is unable to resolve the ground level of the classification if the number of subclasses at the upper level K is large enough (practically, this takes place at $K \sim 10$). We can speculate that there is a "resolution limit" beyond which the modularity is unable to resolve the substructures in a graph. (For more on this see (Fortunato and Barthélemy, 2007)).

Now consider the performance of the parametric modularity on the above graph G_K. In this example, the graph is not a bipartite one. So, we take as a parametric modularity the quantity $Q(G, P, \lambda) = \sum_{i=1}^{N} \left(l_i/L - \lambda D_i^2/(4L^2) \right)$. Compare this formula with the above definition of the modularity for nonbipartite graphs (1). For this case, take $\lambda = K$. We have $Q(G_K, P_1, K) = Q(G, P_{max})$, and $Q(G_K, P_2, K) = 0$. We conclude that taking $\lambda = K$ enables the parametric modularity to see namely the ground level of the classification.

The big question in using the parametric modularity is how to find the "good values" of the parameter λ. As we have seen, λ has a meaning of the number of clusters on the upper level of clustering, and we normally do not know it beforehand. At this moment we do not give any prescription on defining λ. In what follows, we use the parametric modularity to find our classifications. We always give the value of λ with which one or another classification had been obtained.

What we can state is that varying λ is a useful tool. In our experiments, λ was varied from 1 to 300.

2.3 Clustering: Tidying up

Applying the above clustering algorithm to various large graphs we observed appearance of long tails in the distribution of the clusters in the number of vertexes: Typically, along with a few large clusters, we obtain a large number of relatively small clusters. And the smaller is the cluster, the harder to interpret it. Also, it seems that the appearance of small clusters is not infrequently caused by minor peculiarities in the data.

In the results we present below, the vertexes of the clusters belonging to the long tails are redistributed among a few large clusters. In this section, we describe the procedure of this redistribution of the "astray" vertexes.

The redistribution was obtained with an operation similar to the above T_P. This operation, R_N, depends on a natural number N. It acts on any partition P with number of clusters larger than N, $|P| > N$.

First, the redistribution operation R_N orders the clusters of the partition P by their size. Next, all the vertexes not included in the first N clusters are counted. Let the number of these astray vertexes be M. A redistribution of the astray vertexes among the N largest clusters can be pointed out with the set of coordinates $(x_1, ..., x_M)$. The value taken by the coordinate x_k equals the number of the large cluster the kth astray vertex is redistributed to.

As in the operation T_P, the optimal point in the space with the above coordinates is determined by the modularity with the discrete cyclic coordinate descent method (Luenberger, 1973). The only undetermined ingredient in the definition of the redistribution operation R_N is the starting point for the descent.

The starting point for the descent was determined with the following procedure. The value of the first coordinate x_1 was determined by the optimal number of the large cluster for placing the first astray vertex in under the condition that the rest of the astray vertexes are considered as separate clusters. The value of the second coordinate x_2 was determined similarly but under condition that the first of the astray vertexes is already placed into the large cluster number x_1, and so on.

Previously we described the sequence of partitions $P_n = T_{P_0} T_{P_{n-1}} P_{n-1}$. It stops on a partition P. Our final result is $P_f = T_{P_0} R_N P$, where $N < |P|$ is the number of clusters we choose to be present in the final clustering. As before, the leftest operation T_{P_0} improves the clustering (its action determines the optimal cluster for each vertex among the clusters obtained by the action of the redistribution operation R_N).

2.4 Classification

A classification problem is given if a subset of the classification indexes is already given (the training set), and the rest should be generated. To clarify, the number of classes is preset to N. For a subset of vertexes (the training set) the correct classes are known. For the rest of vertexes (testing set) the correct classes should be determined. We attempt to solve the classification problem using its analogy to the problem of redistribution of the astray vertexes of the previous subsection.

To solve the problem using modularity, we point out that the correct classification defines a partition on the set of documents obtained from joining the training and testing sets. The members of this partition are the classes consisting from the documents of the training set with addition of the correctly attributed documents from the testing set. We assume that this partition is the one that maximizes the parameterized modularity at a certain value of the parameter λ. If λ is known, this is a problem of maximization with constraints. The constrains fix the number of clusters to N and the distribution among the clusters for the training set.

We look for approximate solution of this problem using the above redistribution operation R_N. Our approximation to the optimal classification is $P_c = R_N P$ where P is the partition with the training set correctly distributed and each of the rest of vertexes belonging to its own cluster.

3. The Experiment

Four document collections have been used for testing our algorithm. Three of them are among well known classical test collections—`20 Newsgroups, Reuters 21578`, and `WEBKB4`. We used pre-processed versions of these collections (Cardoso-Cachopo). The fourth collection (`TripAdvisor dataset`) is a collection of travelers reviews of the hotels they stayed in obtained via the popular resource `tripadvisor.com` (OpinionAnalysisCorpus). In this collection, all the reviews were classified into two classes—the positive and negative reviews.

Table 1 gives parameters of the collections. All four collections were used for clustering and classification.

Dataset	Total ♯ of docs	♯ of training docs	♯ of test docs	♯ of classes
`20 Newsgroups`	18821	11293	7528	20
`Reuters-21578`	7674	5485	2189	8
`WebKB4`	4199	2803	1396	4
`TripAdvisor`	3000	1800	1200	2

Table 1: Parameters of the text collections

The performance has been measured with the standard quality functionals. For clustering, the performance has been measured with the Purity (Manning et al., 2008) and Normalized Mutual Information (NMI) (Manning et al., 2008) (see below). For classification, it has been measured with micro and macro F1-measures (Manning et al., 2008) (see below).

The bipartite graphs have been formed using stemming, removing stop-words (the stop-list included 770 words) and rare words involved in less than five documents. Besides the graphs representing the document-word pairs, we also constructed larger graphs representing the document-word and document-bigram pairs (the bigram is a sequence of two words involved in a document).

Table 2 gives parameters of the obtained graphs.

Dataset	♯ of vertexes, $G1$	♯ of links, $G1$	♯ of vertexes, $G2$	♯ of links $G2$
`20 Newsgroups`	43000	1000300	131000	2000020
`Reuters-21578`	13000	255000	25000	424000
`WebKB4`	9500	275000	27000	500000
`TripAdvisor`	5400	150000	11200	193000

Table 2: Parameters of the bipartite graphs ($G1$ is the document-word graph, $G2$ is the graph with bigrams included)

We used unit weights for the links in the graphs. (Experimenting with weighted links—we tested the standard tf-idf weights and weights generated via normalization by the document length in the ℓ_2-norm—had not shown improvement sufficient to justify the trouble of using them.)

3.1 Experiment: Clustering

The clustering was performed by the following protocol. For each testing collection, optimization of the parameterized modularity was performed for a sequence of values $\lambda = 1, 1.5, 2, \ldots$ with the objective of finding the suboptimal value of λ.

As mentioned above, the quality of clustering was measured with the Normalized Mutual Information (NMI) and Purity functionals. These functionals are maximal when the generated clustering coincides with a given "correct" clustering. Below we give the formulas for computing these functionals. The clusters of the given "correct" clustering are called classes. The NMI functional reads

$$\mathtt{NMI} = \frac{\sum_l^C \sum_m^K N_{l,m} \log\left(N N_{l,m}/(N_l N_m)\right)}{\sqrt{\sum_m^K N_m \log(N_m/N) \sum_l^C N_l \log(N_l/N)}}. \tag{6}$$

Here summation in l is over the classes, in m over the generated clusters, N is the total number of documents, N_l (N_m) is the number of documents in class l (cluster m), $N_{l,m}$ is the number of documents in the overlap between class l and cluster m, The NMI takes its values in the interval $(0, 1)$, and measures a similarity between the generated clustering and the known partitioning into classes.

For completeness, and to facilitate comparison with other algorithms, we also computed a similar quality criterion—the Purity:

$$\mathtt{Purity} = \frac{\sum_m^K \max_l N_{l,m}}{N}. \tag{7}$$

Table 3 gives the clustering results. It shows that the optimization in the value of λ, and the use of bigrams improves the quality of clustering (measured with NMI) considerably.

Dataset	λ	$G1$			$G2$		
		♯ of clusters	**NMI**	**Purity**	**♯ of clusters**	**NMI**	**Purity**
`20 Newsgroups`	1	9	0.58	0.38	18	0.59	0.43
	2.5	93	0.52	0.62	118	0.60	0.68
	2.5	*20*	*0.59*	*0.61*	***20***	***0.63***	***0.67***
`Reuters-21578`	1	6	0.56	0.80	***5***	***0.63***	***0.84***
`WebKB4`	1	11	0.35	0.70	14	0.34	0.73
	1	*4*	*0.37*	*0.67*	*4*	*0.41*	*0.70*
	1.7	12	0.35	0.68	38	0.37	0.7
	1.7	*4*	*0.37*	*0.67*	***4***	***0.46***	***0.76***
`TripAdvisor`	1	3	0.35	0.80	3	0.36	0.81
	1	*2*	***0.59***	***0.92***	*2*	*0.52*	*0.89*

Table 3: Clustering Results. The $G1$ and $G2$ columns give respectively results obtained with the document-word graph and with the graph involving bigrams. λ is the modularity parameter. Numbers in italic were obtained with the projection onto the first K clusters ordered by their size (K equals the number of clusters in the training set). Numbers in bold give our best results.

Table 4 compares our results with the results obtained with other algorithms. The latter were extracted from sources pointed out in the Table 4 caption.

3.2 Experiment: Classification

In the classification experiments, the suboptimal value of the parameter λ was used determined previously in the clustering experiments.

Dataset/Algorithm	**Modularity**	**ExtPLSA**	**MMF**	**SC**	**SKM**	**CLGR**	**NMF**
`20 Newsgroups`	0.63	0.54	0.61	0.46			
`WebKB4`	0.46	0.36		0.45	0.43	0.54	0.45

Table 4: **NMI** values obtained with various methods. Columns are marked with the method names. **Modularity** is the method of this paper; **ExtPLSA** is a version of the probabilistic latent semantic analysis (Kim et al., 2008); **MMF** is a mixture of the Mises-Fisher distributions (Zhong and Ghosh, 2005); **SC** is the spectral clustering (Zhong and Ghosh, 2005); **SKM** are the spherical K-means (Wang et al., 2007); **CLGR** is Clustering with Local and Global Regularization (Wang et al., 2007); **NMF** is the nonnegative matrix factorization (Wang et al., 2007).

There are two standard classification quality measures (Manning et al., 2008), micro-averaged and macro-averaged:

$$\begin{aligned} \texttt{micro-F1} &= \sum_c \frac{TP(c)}{D}, \\ \texttt{macro-F1} &= \sum_c \frac{F(c)}{N}. \end{aligned} \tag{8}$$

Here the sum in the right hand side of the definitions runs over classes; D is the number of documents to be classified, N is the number of classes, $TP(c)$ is the number of correctly classified documents for class c, and $F(c) = 2R(c)P(c)/(R(c) + P(c))$, where $R(c) = TP(c)/N_1(c)$, and $P(c) = TP(c)/N_2(c)$. In the last relations, $N_1(c)$ ($N_2(c)$) are, respectively, the correct (actual) number of the documents from the testing set to be (have been) attributed to class c.

Table 5 gives results of our classification experiments. The same table compares our results to the results obtained with other algorithms.

	Modularity $G1$		**Modularity $G2$**		**SVM**		**N-Bayes**		**K-NN**	
Dataset	**mic**	**mac**	**mic**	**mac**	**mic**	**mac**	**mic**	**mac**	**mic**	**mac**
`20 Newsgroups`	78.70	77.12	82.19	82.78	82.84	83.60	81.03		84.23	79.07
`Reuters-21578`	91.23	76.25	92.77	81.19	96.98	91.50	96.07		85.24	83.2
`WebKB4`	80.66	78.92	85.24	84.74	89.68	88.39	83.52		72.56	
`TripAdvisor`	90.30	90.10	85.60	85.60						

Table 5: The columns **Modularity** $G1$ and **Modularity** $G2$ give the micro- and macro-F1 values obtained with the algorithms of this paper. The rest of the columns list the values obtained with various methods: **SVM** with support vector machine (Cardoso-Cachopo, 2007),(Guo et al., 2004); **N-Bayes** with the naive Bayes (Guo et al., 2004),(Cardoso-Cachopo, 2007); **K-NN** with the K nearest neighbors method (Cardoso-Cachopo, 2007),(Guo et al., 2004).

4. Conclusions

We presented a new algorithm for clustering and classification of text collections. Our algorithm optimizes modularity computed for a fundamental object—the word-document bipartite graph.

At a competitive quality of the output, our algorithm's main boast is its speed: Using the results on the clustering of a large web-graph (about one billion of the edges) (Blondel et al., 2008), we estimate the time complexity of the clustering task for a collection of 10 millions of documents (each document about the average size of the documents from `20 Newsgroups` collection) as several hours for a typical hardware.

We conclude that our algorithm can be used for clustering very large document collections in reasonable time. With our algorithm, the size of amenable collections can be increased at least an order of magnitude.

We believe that using our algorithm opens up new possibilities for automated structuring of the enormous number of text documents available via the web.

References

Michael J. Barber. Modularity and community detection in bipartite networks. *Phys. Rev. E*, 76(6):066102, 2007.

Michael W. Berry. *Survey of Text Mining*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

Peter J. Bickel and Aiyou Chen. A nonparametric view of network models and newman girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50): 21068–21073, 2009.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner. Maximizing modularity is hard. *physics/0608255*, 2006.

Ana Cardoso-Cachopo. Datasets for single-label text categorization. `http://web.ist.utl.pt/~acardoso/datasets/`.

Ana Cardoso-Cachopo. *Improving Methods for Single-label Text Categorization*. PhD thesis, IST, october 2007.

Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '01, pages 269–274, New York, NY, USA, 2001. ACM.

S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36, 2007.

Santo Fortunato. Community detection in graphs. *PHYSICS REPORTS*, 486:75, 2010.

Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 661–670. ACM, 2009.

Gongde Guo, Hui wang, David Bell, Yaxin Bi, , Kieran Greer, and Kieran Greer. An knn model-based approach and its application in text categorization. In *Computational Linguistics and Intelligence Text Processing, 5th International Conference, CICLing 2004, Springer, Seoul, Korea*, pages 559–570, 2004.

Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, 83(1):016107, 2011.

Young-Min Kim, Jean-François Pessiot, Massih-Reza Amini, and Patrick Gallinari. An extension of plsa for document clustering. In *CIKM*, pages 1345–1346, 2008.

Renaud Lambiotte. Multi-scale modularity in complex networks. *physics/1004.4268*, 2010.

David G. Luenberger. *Introduction to Linear and Nonlinear Programming.* Addison-Wesley, 1973.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008.

M. E. J. Newman. Modularity and community structure in networks. *PROC.NATL.ACAD.SCI.USA*, 103:8577, 2006.

OpinionAnalysisCorpus. http://www.dsic.upv.es/grupos/nle/resources/corpusPM.zip.

Joerg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *PHYS.REV.E*, 74:016110, 2006.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395 – 416, 2007.

Christopher M. De Vries and Shlomo Geva. Document clustering with k-tree. *cs.IR/1001.0827*, 2010.

Fei Wang, Changshui Zhang, and Tao Li. Regularized clustering for documents. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 95–102, 2007.

Hongyuan Zha, Xiaofeng He, Chris Ding, Horst Simon, and Ming Gu. Bipartite graph partitioning and data clustering. In *Proceedings of the tenth international conference on Information and knowledge management*, CIKM '01, pages 25–32, New York, NY, USA, 2001. ACM.

Shi Zhong and Joydeep Ghosh. Generative model-based document clustering: a comparative study. *Knowl. Inf. Syst.*, 8(3):374–384, 2005.