Complex Adaptive Systems Modeling
a SpringerOpen Journal

METHODOLOGY Open Access

Clustering datasets by complex networks analysis

Giuliano Armano and Marco Alberto Javarone*

*Correspondence:
marco.javarone@diee.unica.it
DIEE - Department of Electrical and Electronic Engineering University of Cagliari Piazza d'Armi, Cagliari - 09123, Italy

Abstract

This paper proposes a method based on complex networks analysis, devised to perform clustering on multidimensional datasets. In particular, the method maps the elements of the dataset in hand to a weighted network according to the similarity that holds among data. Network weights are computed by transforming the Euclidean distances measured between data according to a Gaussian model. Notably, this model depends on a parameter that controls the shape of the actual functions. Running the Gaussian transformation with different values of the parameter allows to perform multiresolution analysis, which gives important information about the number of clusters expected to be optimal or suboptimal.
Solutions obtained running the proposed method on simple synthetic datasets allowed to identify a recurrent pattern, which has been found in more complex, synthetic and real, datasets.

Keywords: Clustering, Community detection, Complex networks, Multiresolution analysis

Background

Complex networks are used in different domains to model specific structures or behaviors 2010. Relevant examples are the Web, biological neural networks, and social networks 2002, 2004, 2003. Community detection is one of the most important processes in complex network analysis, aimed at identifying groups of highly mutually interconnected nodes, called communities 2004, in a relational space. From a complex network perspective, a community is identified after modeling any given dataset as graph. For instance, a social network inherently contains communities of people linked by some (typically binary) relations –e.g., friendship, sports, hobbies, movies, books, or religion. On the other hand, from a machine learning perspective, a community can be thought of as a cluster. In this case, elements of the domain are usually described by a set of features, or properties, which permit to assign each instance a point in a multidimensional space. The concept of similarity is prominent here, as clusters are typically identified by focusing on common properties (e.g., age, employment, health records).

The problem of clustering multidimensional datasets without a priori knowledge about them is still open in the machine learning community (see, for example, 2010, 2001, 1998). Although complex networks are apparently more suited to deal with relations rather than properties, nothing prevents from representing a dataset as complex network. In fact, the idea of viewing datasets as networks of data has already been developed in previous works. Just to cite few, Heimo et al. 2008 studied the problem of multiresolution module detection in dense weighted networks, using a weighted version of the q-state Potts method. Mucha et al. 2010 developed a generalized framework to study community structures of arbitrary multislice networks. Toivonen et al. 2012 used network methods in analyzing similarity data with the aim to study Finnish emotion concepts. Furthermore, a similar approach has been developed by Gudkov et al. 2008, who devised and implemented a method for detecting communities and hierarchical substructures in complex networks. The method represents nodes as point masses in an $N - 1$ dimensional space and uses a linear model to account for mutual interactions.

The motivation for representing a dataset as graph lies in the fact that very effective algorithms exist on the complex network side to perform community detection. Hence, these algorithms could be used to perform clustering once the given dataset has been given a graph-based representation. Following this insight, in this paper we propose a method for clustering multidimensional datasets in which they are first mapped to weighted networks and then community detection is enforced to identify relevant clusters. A Gaussian transformation is used to turn distances of the original (i.e. feature-based) space to link weights of the complex networks side. As the underlying Gaussian model is parametric, the possibility to run Gaussian transformations multiple times (while varying the parameter) is exploited to perform multiresolution analysis, aimed at identifying the optimal or suboptimal number of clusters.

The proposed method, called *DAN* (standing for Datasets as Networks), makes a step forward in the direction of investigating the possibility of using complex network analysis as a proper machine learning tool. The remainder of the paper is structured as follows: Section Methods describes how to model a dataset as complex network and gives details about multiresolution analysis. For the sake of readability, the section briefly recalls also some informative notion about the adopted community detection algorithm. Section Results and discussion illustrates the experiments and analyzes the corresponding results. The section recalls also some relevant notions of clustering, including two well-known algorithms, used therein for the sake of comparison. Conclusions (i.e. Section Conclusions) end the paper.

Methods

The first step of the *DAN* method consists of mapping the dataset in hand to a complex network. The easiest way to use a complex network for encoding a dataset is to let nodes denote the elements of the dataset and links denote their similarity. In particular, we assume that the weight of a link depends only on the distance among the involved elements. To put the model into practice, we defined a family of Gaussian functions –used for computing the weight between two elements.

Computing similarity among data

Let us briefly recall that a metric space is identified by a set $\mathcal{Z}$, together with a distance function $d : \mathcal{Z} \times \mathcal{Z} \rightarrow \mathbb{R}$, like Euclidean, Manhattan and Chebyshev distances. In *DAN*, the underlying assumption is that a sample s can be described by N features $f_1, f_2, \ldots, f_N$, encoded as real numbers. In other words, the sample can be represented as a vector in an N-dimensional metric space $\mathcal{S}$. Our goal is to generate a fully connected weighted network taking into account the distances that hold in $\mathcal{S}$. Conversely, the complex network space will be denoted as $\mathcal{N}$, with the underlying assumption that for each sample $s_i \in \mathcal{S}$ a corresponding $n_i \in \mathcal{N}$ exists and vice versa. This assumption makes easier to evaluate the proximity value L_{ij} between two $n_i, n_j \in \mathcal{N}$, according to the distance d_{ij} between the corresponding elements $s_i, s_j \in \mathcal{S}$.

Without loss of generality, let us assume that each feature in $\mathcal{S}$ is normalized in $[0, 1]$ and that a function $\psi : \mathbb{R} \rightarrow \mathbb{R}$ exists for computing the similarity among data in $\mathcal{N}$, starting from the value of the distance function in $\mathcal{S}$. In symbols:

$$L(n_i, n_j) = L_{ij} \triangleq \psi(d_{ij}) = \psi(d\,(s_i, s_j)) \tag{1}$$

Evaluating similarity for all pairs of samples in $\mathcal{N}$ (i.e., evaluating their weighted links) allows to generate a fully connected complex network. Moreover, recalling that $\mathcal{S}$ is normalized in $[0, 1]$, we expect $L_{ij} \approx 0$ when $d_{ij} \approx \sqrt{N}$, N being the number of features of the space $\mathcal{S}$. The value $\sqrt{N}$ comes from the following inequality, which holds for any pair of samples $s_i, s_j \in \mathcal{S}$ (represented by their vector representation in terms of the given set of features $\mathbf{r_i}, \mathbf{r_j}$):

$$d_{ij} = \sqrt{\sum_{k=1}^{N} (\mathbf{r_i}[\,k] - \mathbf{r_j}[\,k]\,)^2} \leq \sqrt{N} \tag{2}$$

where $\mathbf{r_i}[\,k]$ denotes the k-th component of $\mathbf{r_i}$.

The adopted community detection algorithm

Community detection is the process of finding communities in a graph (the process is also called "graph partitioning"). From a computational perspective, this is not a simple task and many algorithms have been proposed, according to three main categories: divisive, agglomerative, and optimization algorithms. In our work, we used the Louvain method 2008, an optimization algorithm based on an objective function devised to estimate the quality of partitions. In particular, at each iteration, the Louvain Method tries to maximize the so-called *weighted-modularity*, defined as:

$$Q = \frac{1}{2m} \cdot \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \cdot \delta(s_i, s_j) \tag{3}$$

where A_{ij} is the generic element of the adjacency matrix, k is the degree of a node, m is the total "weight" of the network, and $\delta(s_i, s_j)$ is the Kronecker Delta, used to assert whether a pair of samples belongs to the same community or not.

Multiresolution analysis

Let us recall that multiresolution analysis is performed with the goal of extracting relevant information, useful for identifying the optimal or suboptimal number of communities (hence, of clusters). To perform multiresolution analysis on the network space, a parametric family $\Psi(\lambda) : \mathbb{R} \rightarrow \mathbb{R}$ of functions is required, where λ is a parameter that controls the shape of each ψ function. After setting a value for λ, the corresponding ψ can be used to convert the distance computed for each pair of samples in the given dataset into a proximity value. In particular, the following parametric family of Gaussian functions has been experimented:

$$\Psi(\lambda; x) = e^{-\lambda x^2} \tag{4}$$

As a consequence, L_{ij}, i.e. the weight of the link between two nodes $n_i, n_j \in \mathcal{N}$, can be evaluated according to Equation 4 as follows:

$$L_{ij} \triangleq \psi(\lambda; d_{ij}) = e^{-\lambda d_{ij}^2} \tag{5}$$

where the λ parameter is used as a constant decay of the link.

Following the definition of $\Psi(\lambda; x)$ as $e^{-\lambda x^2}$, multiresolution analysis takes place varying the value of the λ parameter. The specific strategy adopted for varying λ is described in the experimental section. As for now, let us note that an exponential function with negative constant decay ensures that distant points in an Euclidean space are loosely coupled in the network space and vice versa. Moreover, this construction is useful only if $\Psi(\lambda; x)$ models local neighborhoods, which gives further support to the choice of Gaussian functions 2007.

Results and discussion

Experiments have been divided in three main groups: i) *preliminary tests*, aimed at running *DAN* on few and relatively simple synthetic datasets, ii) *proper tests*, aimed at running *DAN* on more complex datasets, and iii) *comparisons*, aimed at assessing the behavior of *DAN* with reference to *k−Means* and spectral clustering.

Almost all datasets used for experiments (except for Iris) are synthetic and have been generated according to the following algorithm:

Inputs: number of samples (n), dimension in the Euclidean space (N), number of clusters (k), and radius of a cluster (r)

1. For each cluster $j = 1, 2, \ldots, k$, choose a random position c_j in the normalized Euclidean space;
2. Equally subdivide samples among clusters and randomly spread them around each position c_j, with a distance from c_j in $[0, r]$.

Preliminary tests

A first group of 4 synthetic datasets, called *TS/1* (i.e., Testing Set 1) hereinafter, has been generated. Their main characteristics are summarized in Table 1. Figure 1 shows the datasets with 3 and 10 clusters, together with the optimal solutions achieved by *DAN*.

Table 1 Features of datasets used for preliminary tests (*TS/1*)

Group	*Dim*	N_s	N_c	μ_r	σ_r
	2D	1897	5	0.4	0.3
	3D	1683	3	0.09	0.04
	3D	1500	10	0.42	0.22
	4D	1680	6	0.62	0.45

***Dim*, N_s, and N_c denote the dimension of datasets, the number of samples, and the intrinsic number of clusters. Moreover, μ_r and σ_r denote the average radium and the variance of samples.**

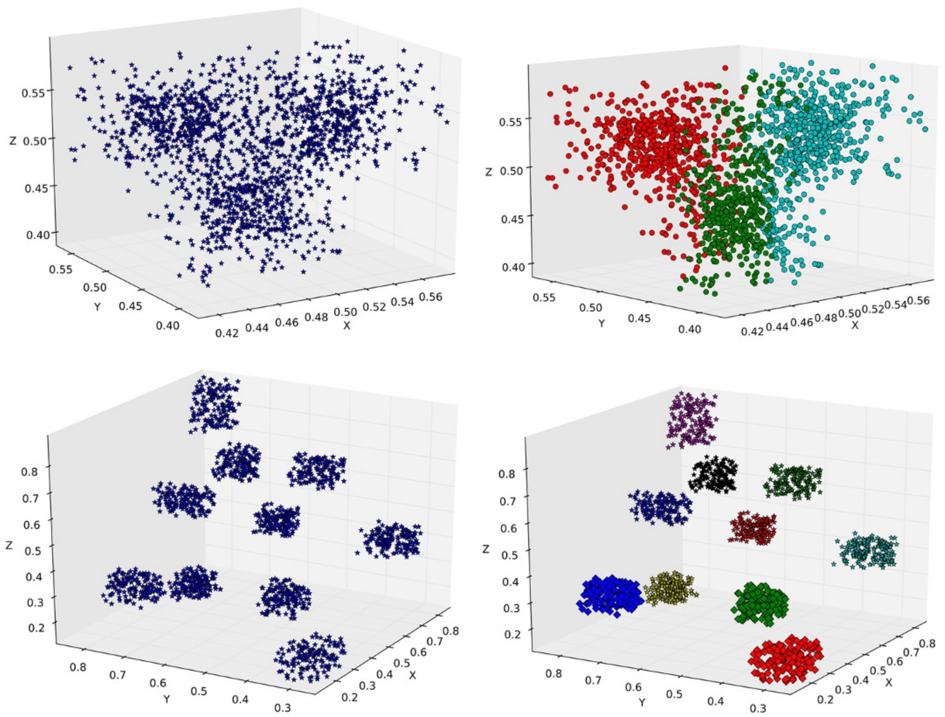

Figure 1 Second and third datasets of *TS/1*, together with the solutions achieved by *DAN* using $\log_{10}(\lambda) = 3$ (each cluster has been colored with a different color).

Multiresolution analysis has been performed varying the value of λ according to Equation 4. A logarithm scaling has been used for λ, as we experimentally found that small changes had a negligible impact on the corresponding algorithm for community detection. In particular, for each dataset, we calculated the adjacency matrix for all values of λ such that $\log_{10}(\lambda) = 0, 1, 2, 3, 4$. It is worth pointing out that the maximum value of $\log_{10}(\lambda)$ is expected to depend on the cardinality of the dataset in hand –the greater the cardinality, the greater the value of $\log_{10}(\lambda)$. However, for most datasets, a value of $\log_{10}(\lambda) = 4$, i.e., $\lambda = 10,000$, appears to be large enough to include all relevant information by means of multiresolution analysis. Table 2 shows the results of multiresolution analysis for preliminary tests.

As for the capability of identifying the optimal or suboptimal solutions[a] by means of multiresolution analysis, we observed the following pattern to occur: the optimal number of communities is robust with respect to the values of $\log_{10}(\lambda)$, as highlighted in Table 2.

Table 2 Results of multiresolution analysis achieved during preliminary tests

Group	N_c	Number of Clusters				
	5	2	3	**5**	**5**	**5**
	3	**3**	**3**	**3**	**3**	103
	10	2	3	**10**	**10**	151
	6	2	4	**6**	**6**	37
		0	1	2	3	4
				$\log_{10}(\lambda)$		

The number of communities is reported, calculated for $\log_{10}(\lambda) = 0, 1, 2, 3, 4$. Optimal values are highlighted in bold.

Table 3 Characteristics of datasets used for proper tests (*TS/2*), listed out according to the group they belong to

Group	*Dim*	N_s	N_c	μ_r	σ_r
	3D	350	5	0.35	0.19
	3D	2000	20	0.44	0.2
	3D	5000	30	0.51	0.24
	4D	535	4	0.64	0.46
	8D	1680	6	0.86	0.62
	12D	930	8	1.22	0.88
Iris	4D	150	3	0.49	0.26

***Dim*, N_s, and N_c denote the dimension of datasets, the number of samples, and the intrinsic number of clusters. Moreover, μ_r and σ_r denote the average radium and the variance of samples.**

Our hypothesis was that this recurrent pattern could be considered as a decision rule for identifying the optimal number of communities (and hence of λ).

Proper Tests (*TS/2*)

We generated a second group of datasets, characterized by an increasing complexity with respect to *TS/1*. This second group of datasets is denoted as *TS/2* (i.e., Testing Set 2) hereinafter. We run *DAN* also on these new datasets, with the goal of verifying the validity of the pattern identified during preliminary tests. Moreover, we performed experiments using *Iris*, a well-known multivariate real dataset available at the UCI ML repository 2010. Iris contains 50 samples (described by 4 attributes) belonging to 3 species of Iris: setosa, virginica and versicolor. Table 3 summarizes the main characteristics of *TS/2* and *Iris*. The corresponding results, obtained with *DAN*, are shown in Table 4.

Looking at these results, we still observe the pattern identified by preliminary tests. Furthermore, one may note that a correlation often exists between the cardinality of the dataset in hand and the order of magnitude of its optimal λ (typically, the former and the latter have the same order of magnitude). It is also interesting to note that in some datasets of *TS/1* (i.e., 2nd, 3rd and 4th) and of *TS/2* (i.e., 4th, 5th and 6th) the optimal λ precedes a rapid increase in the number of communities. As a final note, we found no significant correlation between the optimal λ and the weighted-modularity parameter, notwithstanding the fact that this parameter is typically important to assess the performance of the adopted community detection algorithm.

Table 4 Results of multiresolution analysis on the selected datasets during proper tests, listed out according to the group they belong to

Group	N_c	Pattern	Number of Clusters				
	5	✓	3	**5**	**5**	8	84
	20	✓	3	4	16	**20**	**21**
	30	✓	4	5	21	**30**	**30**
	4	✓	2	**4**	**4**	105	181
	6	✓	2	4	**6**	**6**	1186
	8	✓	3	5	**8**	**8**	875
Iris	3	✓	**3**	**3**	10	82	147
			0	1	2	3	4
						$\log_{10}(\lambda)$	

The number of communities is reported, calculated for $\log_{10}(\lambda) = 0, 1, 2, 3, 4$. Optimal values are reported in bold. The patterns observed on synthetic datasets (and reported in the table for the sake of completeness), allows to easily compute the expected optimal number of communities also for *Iris*.

Comparison: *DAN* vs. *k*-Means and spectral clustering

Experimental results obtained with the proposed method have been compared with those obtained by running two clustering algorithms: the k*—Means* and the spectral clustering. For the sake of readability, let us preliminarily spend few words on these algorithms.

K-*means*

As centroid-based clustering is one of the most acknowledged clustering strategies, the $k-$*Means* algorithm (e.g., 1998), which belongs to this family, has been selected as one of the comparative tools. For the sake of completeness, let us briefly summarize it:

1. Randomly place k centroids in the given metric space;
2. Assign each sample to the closest centroid, thus identifying tentative clusters;
3. Compute the Center of Mass (CM) of each cluster;
4. IF CMs and centroids (nearly) coincide THEN STOP;
5. Let CMs become the new centroids;
6. REPEAT from STEP 2.

The evaluation function of $k-$ *Means*, called *distortion* and usually denoted as J, is computed according to the formula:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n_j} \left| s_i^{(j)} - c_j \right|^2 \tag{6}$$

where n_j is the number of samples that belong to the j-th cluster, $s_i^{(j)}$ is the i-th sample belonging to j-th cluster, and c_j its centroid. Note that different outputs of the algorithm can be compared in terms of distortion only after fixing k –i.e., the number of clusters. In fact, comparisons performed over different values of k are not feasible, as the more k increases the lower the distortion is. For this reason, the use of k*—Means* entails a main issue: how to identify the optimal number k of centroids (see 2004).

Spectral clustering

Spectral clustering 2007 algorithms use the spectrum of the similarity matrix to identify relevant clusters (the generic element of a similarity matrix measures the similarity between the corresponding data). These methods allow to perform dimensionality reduction, so that clustering can be enforced along fewer dimensions. Similarity matrices can be generated in different ways –e.g., ϵ-neighborhood graph, k-nearest neighbor graphs and fully connected graph. The main tools for spectral clustering are graph Laplacian matrices. In particular, in this work we used the unnormalized graph Laplacian matrix defined as:

$$L = D - W \tag{7}$$

where D is the degree matrix (i.e., a diagonal matrix with the degrees $d_1, \ldots, d_n$ on the diagonal) and W is the adjacency (or similarity) matrix of the similarity graph. The following algorithm has been used to perform unnormalized spectral clustering:

1. Generate the fully connected similarity graph and let W be its adjacency matrix;
2. Compute the unnormalized Laplacian L;
3. Compute the first k eigenvectors $u_1, \ldots, u_k$ of L;
4. Let $U \in \Re^k$ be the matrix containing the eigenvectors $u_1, \ldots, u_k$ as columns;

5. For $i = 1, \ldots, n$, let $y_i \in \Re^k$ be the vector corresponding to the i-th row of U;
6. Cluster the points $(y_i)_{i=1,\ldots,n}$ in $\Re^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$.

Notably, also in this case the number k of cluster is required as input.

Comparative results

We run the k-Means algorithm (using the Euclidean metric) and the spectral clustering algorithm on the selected datasets –with the goal of getting new insights on the results of the partitioning procedure defined in *DAN*. Both algorithms used for comparison purposes have been run using the optimal values of k identified by means of multiresolution analysis. The comparison has been performed considering the distortion J computed for each solution. Figure 2 reports comparative results and clearly shows that, in around 72.2 percent of the cases, *DAN* achieves the best result. These results highlight the validity of the proposed framework, also considering that *DAN* computes partitions without any a priori knowledge about the datasets, as the optimal (or suboptimal) number of clusters is typically found by applying the previously described pattern. Although $k-Means$ is faster than *DAN*, it is important to stress that its results, at each attempt, depend tightly on the initial position of the k centroids. Hence, in absence of a strategy for identifying the initial disposal of centroids, $k-Means$ should be (and it is in fact) run several times –the solution with the smaller distortion being selected as optimal. The spectral clustering algorithm showed its effectiveness many times, although bad solutions have been computed with datasets 2 and 3 of *TS/2*, characterized by 20 and 30 clusters, respectively.

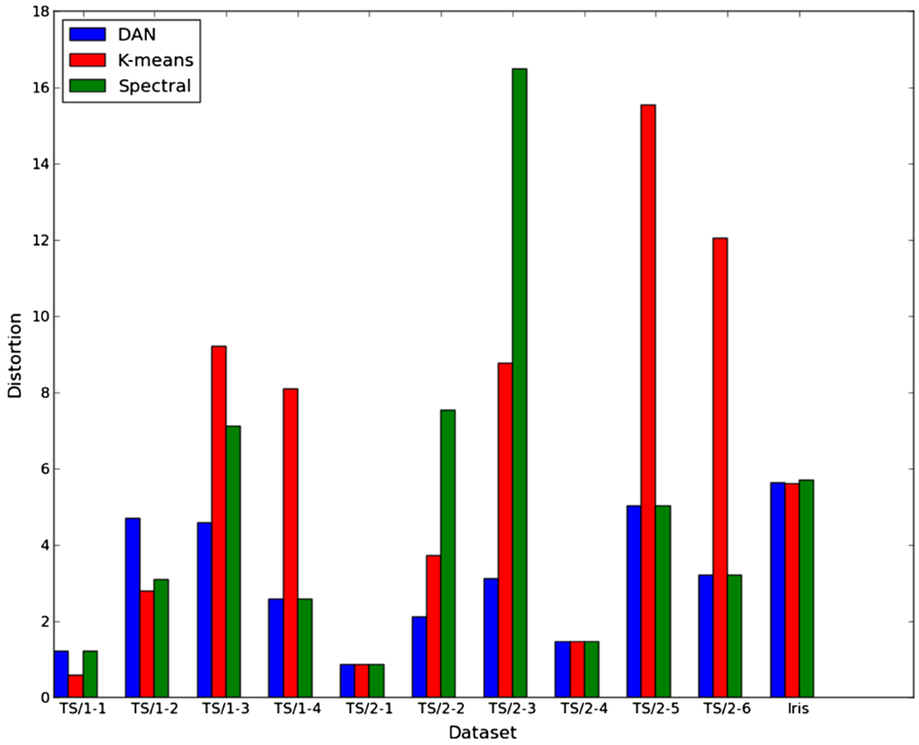

Figure 2 Comparison, in terms of distortion, among solutions achieved by *DAN*, blue bars, $k-Means$, red bars and spectral clustering, green bars (the lesser the better).

Conclusions

In this paper, a method for clustering multidimensional datasets has been described, able to find the most appropriate number of clusters also in absence of a priori knowledge. We have shown that community detection can be effectively used also for data clustering tasks, provided that datasets are viewed as complex networks. The proposed method, called *DAN*, makes use of transformations between metric spaces and enforces multiresolution analysis. A comparative assessment with other well-known clustering algorithms (i.e., $k-Means$ and spectral clustering) has also been performed, showing that *DAN* often computes better results.

As for future work, we are planning to test *DAN* with other relevant datasets, in a comparative setting. Furthermore, we are planning to study to which extent one can rely on the decision pattern described in the paper, assessing its statistical significance over a large number of datasets.

Endnote

[a] As pointed out by Arenas et al. 2008, it may not appropriate to speak of correct vs. incorrect solutions for multiresolution analysis. In a context of community detection we deem more appropriate to speak of optimal or suboptimal solutions (see also 2011 for more information on this issue).

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

Both authors had the first insight about using complex networks to perform clustering tasks. They investigated the properties of the corresponding models and have devised suitable experiments to assess them. Marco A. Javarone has performed all experiments, using Python as programming language. Both authors read and approved the final manuscript.

Acknowledgements

Many thanks to Alessandro Chessa and to Vincenzo De Leo (both from Linkalab). The former for his wealth of ideas about complex networks and the latter for the support given to install and run their Complex Network Library.

Received: 14 October 2012 Accepted: 15 January 2013 Published: 13 March 2013

References

Albert, R, Barabasi A: **Statistical Mechanics of Complex Networks.** *Rev Mod Phys* 2002, **74:**47–97.

Alsabti, K: **An efficient k-means clustering algorithm.** In *Proceedings of IPPS/SPDP Workshop on High Performance Data Mining*; 1998.

Arenas, A, Fernandez A, Gomez S: **Analysis of the structure of complex networks at different resolution levels.** *New Journal of Physics* 2008, **10**(5):053039.

Blondel, VD, Guillaume JL, Lambiotte R, Lefebvre E: **Fast unfolding of communities in large network.** *Journal of Statistical Mechanics: Theory and Experiment* 2008. **P10008**.

Eick, C, Zeidat N, Zhao Z: **Supervised Clustering – Algorithms and Benefits.** In *Proc. of ICTAI*; 2004.

Frank, A, Asuncion A: **UCI Machine Learning Repository.** 2010. [http://archive.ics.uci.edu/ml].

Gudkov, V, Montealegre V, Nussinov S, Nussinov Z: **Community detection in complex networks by dynamical simplex evolution.** *Phys Rev E* 2008, **78:**016113.

Guimer, R, Danon L, Diaz-Guilera A, Giralt F, Arenas A: **Self-similar community structure in a network of human interactions.** *Phys Rev E Stat Nonlin Soft Matter Phys* 2003, **68:**065103.

Jain, AK: **Data clustering: 50 years beyond K-means.** *Pattern Recognition Letters* 2010, **31**(8):651–666.

Li, Z, Hu Y, Xu B, Di Z, Fan Y: **Detecting the optimal number of communities in complex networks.** *Physica A: Statistical Mehcanics and Its Applications* 2011, **391:**1770–1776.

Mark, HH, Yu B: **Model Selection and the Principle of Minimum Description Length.** *Journal of the American Statistical Association* 1998, **96:**746–774.

Mucha, P, Richardson T, Macon K, Porter M, Onnela J: **Community Structure in Time-Dependent, Multiscale, and Multiplex Networks.** *Science* 2010, **328**(5980):876–878.

Newman, MEJ, Girvan M: **Finding and evaluating community structure in networks.** *Phys Rev* 2004, **69:**026113.

Newman, M: *Networks: An Introduction*. New York, NY, USA: Oxford University Press, Inc.; 2010.

Sporns, O, Chialvo DR, Kaiser M, Hilgetag C: **Organization, development and function of complex brain networks.** *Trend in Cognitive Sciences* 2004, **8**(9).

Heimo, T, Kaski K, Kumpula JM, Saramaki J: **Detecting modules in dense weighted networks with the Potts method.** *Journal of Statistical Mechanics: Theory and Experiment* 2008, **08:**08007.

Tibshirani, R, Walther G, Hastie T: **Estimating the number of clusters in a data set via the gap statistic.** *Journal of the Royal Statistical Society - Series B: Statistical Methodology* 2001, **63**(2):411–423.

Toivonen, R, Kivela M, Saramaki J, Viinikainen M, Vanhatalo M, Sams M: **Networks of Emotion Concepts.** *PLoS ONE* 2012, **7**(1):e28883.

von Luxburg, U: **A Tutorial on Spectral Clustering.** *Statistics and Computing* 2007, **17**(4):395–416.

doi:10.1186/2194-3206-1-5
Cite this article as: Armano and Javarone: Clustering datasets by complex networks analysis. *Complex Adaptive Systems Modeling* 2013 **1**:5.