Kernel spectral clustering for community detection in complex networks

Rocco Langone, Carlos Alzate and Johan A. K. Suykens
Department of Electrical Engineering ESAT-SCD-SISTA
Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
Email:{rocco.langone, carlos.alzate, johan.suykens}@esat.kuleuven.be

***Abstract*—This paper is related to community detection in complex networks. We show the use of kernel spectral clustering for the analysis of unweighted networks. We employ the primal-dual framework and make use of out-of-sample extension. In the latter the assignement rule for the new nodes is based on a model learned in the training phase. We propose a method to extract from a network a small subgraph representative for its overall community structure. We use a model selection procedure based on the modularity statistic which is novel, because modularity is commonly used only at a training level. We demonstrate the effectiveness of our model on synthetic networks and benchmark data from real networks (power grid network and protein interaction network of yeast). Finally, we compare our model with the Nyström method, showing that our approach is better in terms of quality of the discovered partitions and needs less computation time.**

I. Introduction

In recent years, the study of networks represents a major topic in the scientific community (see for example [1] for a complete overview on the subject). Many complex systems can be described as networks, where the nodes (or vertices) represent some entities between which some relationships exist. Examples include social networks, web graphs, telecommunication networks, biological networks, trade networks. In this framework, a hot topic is the community detection problem or clustering, that consists in to identify groups of nodes within which the connections (or edges) are numerous and between which they are scarce. Spectral clustering methods are a standard technique used for clustering, based on the eigendecomposition of a Laplacian matrix derived from the data. Recently a spectral clustering formulation as a weighted kernel PCA problem with primal and dual representations has been proposed in [2].

The main advantage of this interpretation is the extension of the clustering model to out-of-sample nodes. The clustering model can be trained on a small subset of the whole graph (by solving an affordable eigenvalue problem) and then be applied to the rest of the network in a learning framework. This issue is particularly important when we have to deal with huge complex networks. Moreover, the out-of-sample extension is widely used only in our algorithm in the community detection field. For example, in the paper [3], which presents a state of the art algorithm for clustering, the authors demonstrate an equivalence between kernel k-means and spectral clustering. Even if potentially they could use the out-of sample extension, it seems to us that they work only at a training level (they also never mention the out-of-sample extension explicitly). Moreover in that case the out-of-sample extension is related to the use of the same assignement rule used for training points. In our framework, on the other hand, the out-of-sample extension is based on a predictive model previously learned, and for example can allow us to easily solve the problem of online clustering of huge growing networks. In fact, this task can be accomplished by applying the model on every new node arriving in a data stream. This couldn't be the case for most of the community detection algorithms present in the literature that for every new node have to run again on the whole graph. In this picture, two tasks become crucial:

- the choice of a good criterion to properly select the parameters to feed into the model, like the kernel parameters (if any) and the number of clusters. In fact this allows to obtain a relevant grouping among the data.
- the extraction of a subgraph which is representative of the overall community structure characterizing the entire network.

In order to achieve the first goal here we use a new method proposed in [4]. This criterion is based on Modularity, a quality function introduced in [5]. The Modularity statistic (although with some drawbacks) has been shown to be a meaningful quality function accounting for the presence of a significant community structure in networks. It quantifies the quality of a division of a network into modules. The most common use of the Modularity is a basis for optimization methods for detecting community structure in networks, but only at the training level (like in [6]). In our case, however, we use it as a cluster validity criterion for our model selection purposes. In particular, we consider Modularity to judge the partitions found by the kernel spectral clustering algorithm, which is based on its own optimization problem (briefly described later). Regarding the selection of a small representative subgraph to use as training set, we propose an active selection method based on the greedy optimization of the expansion factor (see [7]), taking inspiration also from fixed-size kernel models [8].

Once the model has been trained on a small subgraph (fed with appropriate parameters), it can be easily extended to unseen nodes in a machine learning framework with a low computational burden. Up to our knowledge, this characteristic is unique in the field of community detection on networks, since most of the best existing algorithms work only at the training level and the eventual out-of-sample extension is not model-based.

The rest of this paper is organized as follows: Section II introduces the problem of community detection. Section III

U.S. Government work not protected by U.S. copyright

summarizes the kernel spectral clustering model. Section IV describes the Modularity-based criterion for model selection, the training and validation set to use in the learning process and the kernel function that has been considered. In section V the active method to select a representative subgraph is presented. Section VI is dedicated to the description of the datasets under investigation. Some simulation results, described in terms of quality of the discovered community structure and computational time, are presented in Section VII: only unweighted networks are taken into account, considering both real and artificial datasets. Moreover, all the experiments are performed in Matlab (for Linux) on an Intel Core Duo Quad with 2.83 GHz. Finally, Section VIII concludes the paper and suggests some future works.

II. Problem statement

A graph (or network) is a mathematical structure used to model pairwise relations between objects from a certain collection. It refers to a set of vertices or nodes and a collection of edges that connect pairs of vertices. A way to represent a graph is the use of a similarity matrix S, which is an $N \times N$ matrix with N equal to the number of vertices in the network. In our case we deal with unweighted graphs and S is called adjacency matrix (in general indicated with the symbol A) and $S_{ij} = 1$ if there is an edge connecting the vertices i and j, otherwise $S_{ij} = 0$. Associated to the similarity matrix there is the degree matrix D, a diagonal matrix with diagonal entries $d_i = \sum_j S_{ij}$ indicating the sum of all the edges connecting node i with the other vertices.

The structure of many networks reveals a high degree of organization, where vertices with similar properties are more likely to be linked together and tend to form modules. Discovering these modules that are naturally present within a graph has become a hot topic in the modern science of complex systems and is called community detection. Moreover, once communities have been detected, the roles of the vertices in each community can be further investigated. For example nodes that have a central position in a module are probably responsible for the control and stability of that cluster, while vertices lying on the boundaries between the discovered communities are likely to have a role of intermediary within the graph. The community structure of a graph can also be used to give a compact visualization of large systems. Indeed, one could display only the communities and their interconnections, leading to a more understandable and intuitive description of the graph. This kind of representation is called a supernetwork: each community is a node of this graph and has a link with the other nodes with weight proportional to the number of intercommunity edges of the original network (see for instance Figures 8 and 9) in Section VII).

Nowadays there is a plethora of algorithms performing the community detection task (see for example [9] for an analysis of some of them). Most of them work only at the training level. This means that they use all the graph to extract the partitions, and if new nodes are added to the network the algorithm has to process the new graph including also the old nodes. On the other hand, our approach allows the out-of-sample extension: once our model has been trained on a small representative subgraph selected from the network under investigation, the membership of any other node can be predicted in a straightforward way, without need of any heuristic technique. This could permit to analyze large networks in a reasonable time and efficiently cluster online data stream.

III. Kernel spectral clustering model

A. General picture

The first step to follow in spectral clustering is to build a similarity matrix among the objects to analyze. If we are given some data points, typically the similarity between the points is related to their mutual distance in the space they are embedded in. This similarity matrix can then be considered a weighted graph. The other case (that one we consider) is when the starting data at hand are already a graph (weighted or unweighted). In this case, classical spectral clustering can be directly applied. Nevertheless, in the kernel spectral clustering model described next, a graph over the starting network needs to be built up in order to describe the similarity among the nodes in the kernel-based framework.

Given a graph, several properties of it can be explained through spectral graph theory, which is the study of the eigenspectrum of graph Laplacian matrices [11], [10]. Typical graph Laplacians are: the unnormalized Laplacian defined as $L = D - S$, the symmetric normalized Laplacian $L_{\text{SYM}} = D^{-1/2}LD^{-1/2} = I_N - D^{-1/2}SD^{-1/2}$ and the non-symmetric normalized Laplacian $L_{\text{RW}} = D^{-1}L = I_N - D^{-1}S$ denoted L_{RW} because it is related to a random walk on the graph. In the latter case, the clustering problem can be interpreted as finding a partition of the graph in such a way that the random walker remains most of the time in the same cluster with few jumps to other clusters, minimizing the probability of transitions between clusters. The stochastic transition matrix P of a random walker on a graph can be obtained by normalizing the similarity matrix S associated to the graph such that its rows sum to 1. The ij-th entry of P represents the probability of moving from node i to node j in one step of the process. This transition matrix can be defined as $P = D^{-1}S$. The corresponding eigenvalue problem becomes $Pr = \xi r$, and as we show afterward it can be viewed as the dual problem of a constrained optimization problem typical of least squares support vector machines (LS-SVM)[8].

B. Primal-dual formulation

The kernel spectral clustering model is described by a primal-dual formulation. The parameters of the model are estimated in the training phase using training data, some (if any) hyper-parameters are tuned in a validation stage and finally the model is tested on a test set. In our case the data that we deal with are the N nodes of the graph under investigation. Each training node x_i is a row of the adjacency matrix S, that is $x_i = S(i,:) \in \mathbb{R}^N$ (since matrix S is symmetric, considering the columns is equivalent). For more details, also regarding validation data, see Section IV-B. Before continuing

it is worthwhile to point out that for large networks the number of nodes (and then the dimension of each data point x_i) can be very big (nowadays network data can contain millions of nodes). However, this does not represent a problem because usually the networks at hand are sparse and can be easily stored in the memory of a PC. If the graph cannot be fitted completely into the main memory, it can be stored in the hard disk and the memberships, in the test phase, can be predicted just by uploading into the memory the related nodes and calculating their out-of-sample extension. In other words, there is no need to store in the memory the whole graph and the related kernel matrix. Moreover the sparsity allows to reduce the computation time needed for the evaluation of the kernel matrix. Given N_{tr} training nodes $\mathcal{D} = \{x_i\}_{i=1}^{N_{\text{tr}}}, x_i \in \mathbb{R}^N$ and the number of communities k, the primal problem of spectral clustering via weighted kernel PCA is formulated as follows [2]:

$$\min_{w^{(l)},e^{(l)},b_l} \frac{1}{2}\sum_{l=1}^{k-1} w^{(l)^T} w^{(l)} - \frac{1}{2N}\sum_{l=1}^{k-1} \gamma_l e^{(l)^T} D_\Omega^{-1} e^{(l)} \quad (1)$$

$$\text{such that } e^{(l)} = \Phi w^{(l)} + b_l 1_{N_{\text{tr}}} \quad (2)$$

where $e^{(l)} = [e_1^{(l)}, \ldots, e_{N_{\text{tr}}}^{(l)}]^T$ are the projections, $l = 1, \ldots, k-1$ indicates the number of score variables needed to encode the k clusters to find, $D_\Omega^{-1} \in \mathbb{R}^{N_{\text{tr}} \times N_{\text{tr}}}$ is the inverse of the degree matrix associated to the kernel matrix Ω (as explained later), Φ is the $N_{\text{tr}} \times d_h$ feature matrix $\Phi = [\varphi(x_1)^T; \ldots; \varphi(x_{N_{\text{tr}}})^T]$ and $\gamma_l \in \mathbb{R}^+$ are regularization constants. The clustering model is expressed by:

$$e_i^{(l)} = w^{(l)^T} \varphi(x_i) + b_l, i = 1, \ldots, N_{\text{tr}} \quad (3)$$

where $\varphi : \mathbb{R}^N \to \mathbb{R}^{d_h}$ is the mapping to a high-dimensional feature space, b_l are bias terms, $l = 1, \ldots, k-1$. The projections $e_i^{(l)}$ represent the latent variables of a set of $k-1$ binary clustering indicators given by $\text{sign}(e_i^{(l)})$ which can be combined to form the final groups in an encoding/decoding scheme. The dual problem related to this primal formulation is:

$$D_\Omega^{-1} M_D \Omega \alpha^{(l)} = \lambda_l \alpha^{(l)} \quad (4)$$

where Ω is the kernel matrix with ij-th entry $\Omega_{ij} = K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$, D_Ω is the diagonal matrix with diagonal entries $d_i^\Omega = \sum_j \Omega_{ij}$, M_D is a centering matrix defined as $M_D = I_{N_{\text{tr}}} - (1/1_{N_{\text{tr}}}^T D_\Omega^{-1} 1_{N_{\text{tr}}})(1_{Ntr} 1_{N_{\text{tr}}}^T D_\Omega^{-1})$, the $\alpha^{(l)}$ are dual variables. The kernel function $K : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ plays the role of the similarity function of the graph. Now, the dual problem is related to the random walk model and represents the weighted kernel PCA formulation of it used in our simulations (for a complete derivation see [2]).

C. Encoding/decoding scheme

In the ideal case of k well separated clusters and properly chosen kernel parameters, the matrix $D^{-1} M_D \Omega$ has $k-1$ piecewise constant eigenvectors with eigenvalue 1 (see for example [11]). In the eigenvector space every cluster $\mathcal{A}_p$, $p = 1, \ldots, k$ is a point and is represented with a unique codeword $c_p \in \{-1, 1\}^{k-1}$. The codebook $\mathcal{CB} = \{c_p\}_{p=1}^k$ can then be obtained in the training process from the rows of the binarized projected variables matrix for training data $[\text{sign}(e^{(1)}), \ldots, \text{sign}(e^{(k)})]$. An effect of the centering matrix M_D defined in the last section is the fact that the eigenvectors have zero mean. This is important for encoding since the optimal threshold for binarizing the eigenvectors is automatically determined. Moreover this leads to the fact that, unlike classic spectral clustering, the constant eigenvector is not present. Taking into account that the first eigenvector $\alpha^{(1)}$ already provides a binary clustering then number of score variables needed to encode k clusters is $k-1$. The decoding scheme consists of comparing the cluster indicators obtained in the validation/test stage with the codebook and selecting the nearest codeword in terms of Hamming distance. This scheme corresponds to the ECOC decoding procedure and it is used for out-of-sample extension. In particular, the proposed extension is based on the score variables which correspond to the projections of the mapped out-of-sample points onto the eigenvectors found in the training stage. The cluster indicators can be obtained by binarizing the score variables for out-of-sample points as follows:

$$\text{sign}(e_{\text{test}}^{(l)}) = \text{sign}(\Omega_{\text{test}} \alpha^{(l)} + b_l 1_{\text{Ntest}}) \quad (5)$$

with $l = 1, \ldots, k-1$. Ω_{test} is the $N_{\text{test}} \times N_{\text{tr}}$ kernel matrix evaluated using the test points with entries $\Omega_{\text{test,ri}} = K(x_r^{\text{test}}, x_i)$, $r = 1, \ldots, N_{\text{test}}$, $i = 1, \ldots, N_{\text{tr}}$. This natural extension to out-of-sample points corresponds to the main advantage of the kernel spectral clustering framework. In this way, the clustering model can be trained, validated and tested in an unsupervised learning scheme.

Finally, a new notion regarding the out-of-sample points has been introduced in [12]. One of the KKT condition related to the optimization problem (1) links the projections for training data $e^{(l)}$ with eigenvectors $\alpha^{(l)}$. This relationship can be used to compute the out-of-sample eigenvectors $\alpha_{\text{test}}^{(l)} \in \mathbb{R}^{N_{test}}$, by:

$$\alpha_{\text{test}}^{(l)} = \frac{1}{\lambda^{(l)}} (D_{\text{test}}^{-1} e_{\text{test}}^{(l)}), l = 1, \ldots, k-1, \quad (6)$$

where $D_{\text{test}}^{-1} = \text{diag}(\frac{1}{\text{deg}(x_1^{\text{test}}), \ldots, \text{deg}(x_N^{\text{test}})}) \in \mathbb{R}^{N_{test} \times N_{test}}$ is the inverse degree matrix for test data and deg(x) $= \sum_{j=1}^{N_{\text{tr}}} K(x, x_j)$ extends the concept of degree to out-of-sample data. In Section V we will show, for one of the networks under investigation in this paper, how the degree distribution for the out-of-sample data is similar to the real degree distribution. This is a further demonstration that the model can generalize well on new nodes.

IV. Model selection criterion by means of Modularity evaluation

A. Modularity

Often people use heuristics to select the tuning parameters present in their models. Since model selection is a crucial point, here we use a systematic way to do it properly, described in [4]. The method is based on a validation. We train the kernel spectral clustering model described in the previous section

with different number of clusters. In the validation step the obtained groupings are judged depending on Modularity: the one (or more) partition with the highest value of Modularity is selected.

Modularity is a quality function of a graph introduced in [5]. It is based on the idea that a random graph is not expected to have a cluster structure, so the possible existence of clusters can be revealed by the comparison between the actual density of edges and the density one would expect to have in the graph if the vertices were attached randomly, regardless of community structure (this characterizes a particular null model). Modularity can be either positive or negative, with positive high values indicating the possible presence of a strong community structure. It can be written as follows:

$$Q = \frac{1}{2m}\sum_{ij}(S_{ij} - F_{ij})\delta_{ij} \tag{7}$$

with $i, j \in \mathcal{A}_p$. The sum runs over all pairs of vertices, S as before is the similarity matrix, m indicates the sum of all the weights, and F_{ij} represent the expected number of edges between vertices i and j in the null model. The Kronecker delta δ_{ij} function yields 1 if vertices i and j are in the same community and 0 otherwise. Since the standard null model of Modularity imposes that the expected degree sequence matches the actual degree sequence of the graph, the Modularity can be written as: $Q = \frac{1}{2m}\sum_{ij}(S_{ij} - \frac{d_i d_j}{2m})\delta_{ij}$, where we indicate with $d_i = \sum_j S_{ij}$ the degree of the vertex i. Then, after some linear algebra calculations [13], it can be shown that the problem of maximizing the Q-measure in order to find the optimal partition is given by:

$$\max_X [\text{tr}(X^T M X)] \text{ such that } X^T X = D^M. \tag{8}$$

Here $M = S - \frac{1}{2m}dd^T$ is the Modularity matrix or Q-Laplacian, $d = [d_1, \ldots, d_N]$ indicates the vector of the degrees of each node, $D^M \in \mathbb{R}^{k\times k}$ is a diagonal matrix with diagonal entry $D^M_{ii} = |C_i|$ where $|C_i|$ is the number of nodes in cluster C_i, and X represents the cluster indicator matrix.

B. Proposed algorithm

The model selection algorithm can briefly be expressed in the following way:

Algorithm MS Modularity-based model selection algorithm

Input: training set, validation set stage I, validation set stage II, positive (semi-) definite kernel function $K(x_i, x_j)$
Output: selected number of clusters k and (if any) kernel parameters

1) compute cluster indicator matrix X from the cluster results of the different models, obtained using the training set and the validation set I stage in the learning process,
2) compute the Modularity matrix $B = S - \frac{dd^T}{2m}$, where S refer to the validation set used in the II stage of the validation process
3) compute the Modularity $M = \frac{1}{2m}\text{tr}(X^T B X)$,
4) select the model (i.e. k and the kernel parameters) corresponding to the partition(s) which gives the highest Modularity value.

The training set, validation set and the two stages of the validation process have the following meaning. The training set is the matrix given as input to the kernel spectral clustering model during the training phase (see 1). The validation process can be divided into two stages:

1) stage I: the cluster memberships for the validation set (data not belonging to the training set) are predicted by the model based on eq. (5);
2) stage II: the quality of the predicted memberships are judged by means of Modularity criterion.

In these two stages the validation sets involve the same data (the nodes of the graphs under study) but represented in different ways. In stage I some rows of the adjacency matrix are considered: this is called an adjacency list. In stage II the validation set is a square matrix (a kind of validation adjacency matrix), because this is needed in order to calculate the related Modularity. See Figure 1 for further clarification.

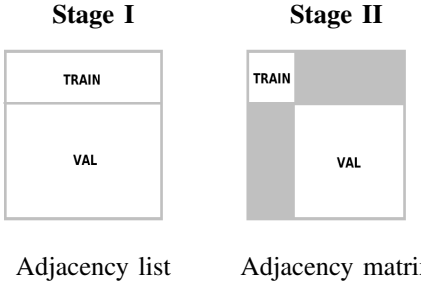


Fig. 1. Example of training and validation set used for unweighted graphs. In this case the first 25% of the total nodes form the training set and the remaining 75% the validation set. In the first stage of the validation process the graph is represented as an adjacency list while in the second stage (consisting of evaluating the quality of the predicted partitions by means of Modularity criterion) it is given as an adjacency matrix.

C. The community kernel

In dealing with unweighted networks a recently proposed kernel function particularly suited for the study of unweighted networks, the community kernel [14] is used to build up the similarity matrix of the graph. This kernel function does not have any parameter to tune and the similarity Ω_{ij} between two nodes i and j is defined as the number of edges connecting the common neighbors of these two nodes: $\Omega_{ij} = \sum_{k,l\in\mathcal{N}_{ij}} A_{kl}$. Here $\mathcal{N}_{ij}$ is the set of the common neighbors of nodes i and j, A indicates the adjacency matrix of the graph, Ω is the kernel matrix. As a consequence, even if two nodes are not directly connected to each other, if they share many common neighbors their similarity Ω_{ij} will be set to a large value. Moreover, in [14] it is empirically observed that this kernel matrix is positive definite, a fundamental requirement in order to use the community kernel in the LS-SVM framework. We use a matlab implementation consisting of one loop and the computation time for calculating the kernel matrix is directly proportional to the number of the training nodes and the sparsity of the dataset. Moreover, since every step of the

loop is independent from the previous, the code is suitable for parallelization on several CPU or on a GPU, with a potential increase in speed of a factor 50 or more (see for example *http://wiki.accelereyes.com/wiki/index.php/JACKET*). Finally, also a C++ implementation could lead to a significant improvement in terms of speed: for example the matlab implementation of the Louvain [6] method takes about 16 s to analyze the artificial network with 3 000 nodes considered in this paper, while the C++ implementation allows it to process, according to its authors, a network with 2 million nodes in a couple of minutes.

V. Selecting a representative subgraph

Sampling a subgraph representative of the community structure of whole network under study is a crucial task in our model, since it allows a meaningful out-of-sample extension to nodes not present in the training set. Simply taking a random sample of nodes can lead to very different results in several runs, since the quality of the selected subgraph can have a huge variability. Theoretically a sampled subgraph can also be a set of disconnected parts, causing bad results in the predicted memberships of the test nodes. Also selecting a subgraph in such a way that it follows the same degree distribution or betweenness centrality distribution of the whole graph can produce samples that are not representative of the community structure of the larger network. Recently a new quality function describing the representativeness of the sample respect to the community structure of the whole graph has been introduced in [7]. This quality function is called expansion factor (EF) and is defined as $\frac{|N(\mathcal{G})|}{|\mathcal{G}|}$, where G indicates a subgraph, $N(\mathcal{G})$ its neighborhood, i.e. the remaining part of the network to which $\mathcal{G}$ is connected, and $||$ is the cardinality of a set. The idea is that by selecting a subgraph for which the expansion factor is maximum, one samples a representative subgraph. Roughly speaking, by including in $\mathcal{G}$ nodes that best contribute to the expansion factor, we are taking nodes that are more connected to the rest of the network than to $\mathcal{G}$. These nodes are then very likely to belong to clusters not yet represented in the subgraph, allowing us to produce a sample which is a condensed representation of the community structure of the whole network.

Here we propose a greedy strategy for the optimization of the expansion factor EF, that can be summarized as follows:

Algorithm EF Subset selection maximizing expansion factor

Input: network of N nodes $\mathcal{V} = \{n_i\}_{i=1}^{N}$ (represented as the adjacency matrix $A \in \mathbb{R}^{N \times N}$), size of subgraph m
Output: active set of m selected nodes

1) select randomly an initial subgraph $\mathcal{G} = \{n_j\}_{j=1}^{m} \subset \mathcal{V}$
2) **repeat**
3) compute $EF(\mathcal{G})$
4) randomly pick two nodes as $n_* \in \mathcal{V}$ and $n_+ \in \mathcal{V} - \mathcal{G}$
5) let $\{\mathcal{W} = \mathcal{V} - \{n_*\}\} \cup \{n_+\}$
6) **if** $EF(\mathcal{W}) > EF(\mathcal{G})$
7) swap($\{n_+\}$,$\{n_*\}$)
8) **end if**
9) **until** change in EF value is too small (according to a threshold ϵ)
10) **return** $\mathcal{G}$

The selection of the active subset can take from a few to several minutes or hours depending on the size N of the entire network and its density, the chosen size m for the active subset and the threshold ϵ.

Figure 2 depicts a typical example of the greedy optimization of the EF performed by the algorithm (in this case the artificial network formed by 3 000 nodes and described later is considered). Moreover, in Figure 4 is shown how the active sampling technique produces better sample than a random sampling. To see this, we compare the ARI index [15] between the partitions predicted by the kernel spectral clustering model and the true memberships, by using the training set selected randomly or actively. In Figure 3 we can see that the degree and betweenness centrality distribution of the active set is quite different from those one of the whole graph. All these empirical observations are in agreement with what has been discussed in [7]. On the other hand, if we compare the degree distribution associated to the full kernel matrix Ω and the out-of-sample degree distribution related to the test kernel matrix Ω_{test} (see Section III-C), they are quite similar (see Fig. 5). This implies that the active selected subset is meaningful and allows our model to correctly generalize to unseen nodes, as it is also shown in Table II.

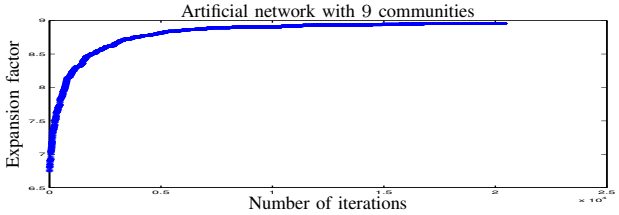


Fig. 2. Example of the greedy optimization of the expansion factor EF.

VI. Description of the datasets

Real and artificial datasets are investigated. The software provided by Fortunato related to the paper [16] is used to produce the unweighted synthetic graphs, while the real datasets that are studied here are now classic real-world networks present in the literature. A short description of the data is furnished in Table I.

Network	Nodes	Edges	Sparsity (%)
Art_9C	3 000	22 904	99.49
Art_13C	10 000	76 789	99.85
Art_22C	50 000	383 220	99.97
Yeast_pro	2 114	4 480	99.90
Power grid	4 941	6 594	99.95

TABLE I
SUMMARY OF SOME PROPERTIES OF THE GRAPHS ANALYZED IN THIS PAPER. SPARSITY REFERS TO THE ADJACENCY MATRIX ASSOCIATED TO EACH GRAPH AND INDICATES THE PERCENTAGE OF ZERO ENTRIES WITH RESPECT THE TOTAL NUMBER OF ELEMENTS OF THE MATRIX.

A. Synthetic networks

Three graphs are investigated:

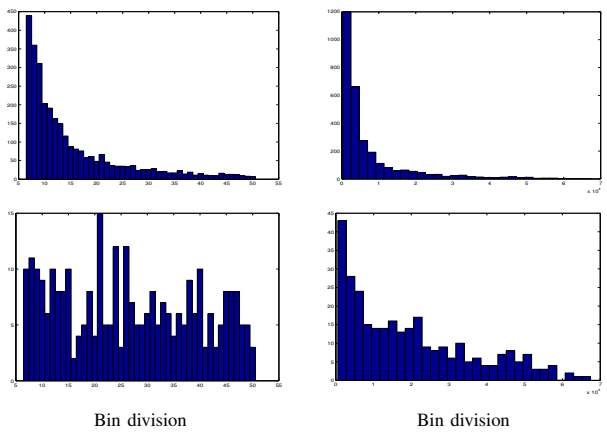


Fig. 3. Degree and betweeness centrality distribution (number of nodes) of the entire starting synthetic network with 9 communities (top left and top right) and of a typical active set selected from the algorithm EF (bottom left and bottom right). We can notice that the representativeness of the set in terms of community structure can not be related to its representativeness in terms of degree distribution.

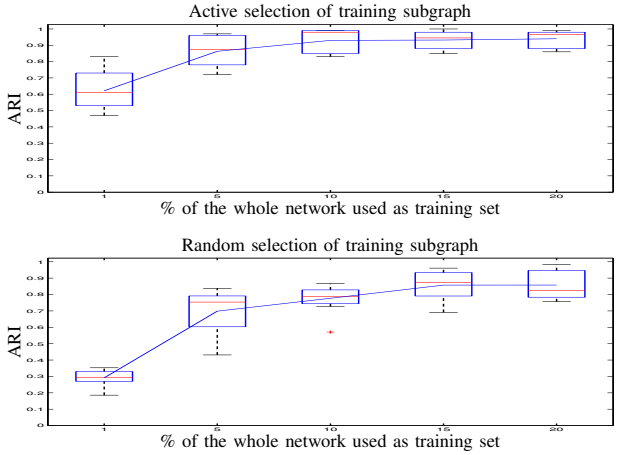


Fig. 4. Results related to the analysis of the artificial network with 9 communities described in section VI-A

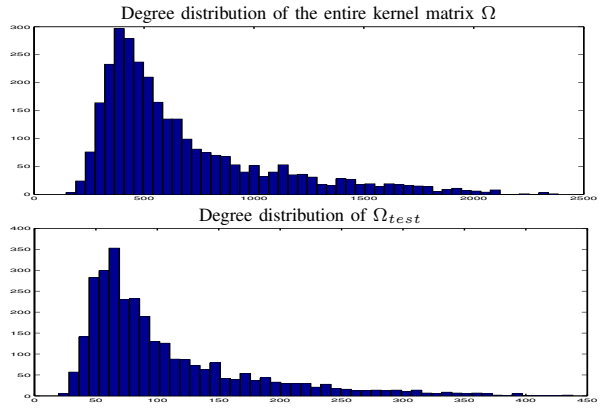


Fig. 5. The degree distribution related to the full kernel matrix (of size $3\,000 \times 3\,000$) describing the similarity between the nodes of the network Art_9C is pictured in the first row. In the second row the degree distribution associated to the test kernel matrix (of size $3\,000 \times 300$) is shown. The two distributions look like quite similar, meaning that the model trained on the active selected subset can generalize well in the test phase.

- Art_9C: a benchmark network with $3\,000$ nodes and $22\,904$ edges formed by 9 communities.
- Art_13C: an artificial network with $10\,000$ nodes and $76\,789$ edges formed by 13 communities.
- Art_22C: a synthetic graph with $50\,000$ nodes and $383\,220$ edges formed by 22 communities.

B. Real networks

The graphs under investigation are:

- Yeast_pro: interaction network data for yeast formed by $2\,114$ nodes and $4\,480$ edges. As explained in [17] proteins can have direct or indirect interactions with one another. Indirect interaction refers to being a member of the same functional module (e.g., transcription initiation complex, ribosome) but without directly binding to one another. In contrast, direct interaction, refers to two amino acid chains that bind to each other. Obviously, many of these interactions reflect the dynamic state of the cell and are present or absent depending on the particular environment or developmental status of the cell.
- Power grid: the network of Western USA power grid [18] formed by $4\,941$ nodes and $6\,594$ edges. The vertices represent generators, transformers and substations, and edges represent high voltage transmission lines between them.

VII. Simulation results

A. General overview

In the case of the synthetic networks, in order to compare the memberships predicted by the kernel spectral clustering model with the true memberships the ARI index is used. On the other hand, in the case of the real datasets, to assess the quality of the clustering produced by the model, the Modularity of the predicted partition is calculated. Finally, the computation time associated to the analysis of the graphs under study refers to the test stage, since in comparison the time needed to find the parameters of the model during the training phase (calculating the kernel matrix and solving the related eigenvalue problem) is negligible. Moreover, the training has to be done just once: in fact, each time we need to calculate the membership of a new node in a network, we just need to extend the trained model on it. All the results, related to the test and (if any) validation stage are summarized in Table II and Figures 6, 7. For the power grid network, the model selection procedure performed here gives us another indication regarding the number of cluster in which to divide the network compared to the model selection performed in [4]. The suggested divisions are both good in terms of their quality. For instance all the partitions with Modularity value above 0.51 can be taken into account. In this paper we take into consideration the new result (16 communities), since here the average Modularity achieves a clearer maximum on a wider range of possible choices. In Table III also the results obtained using the Nyström method are shown: only the synthetic graphs are considered, in order to make a more objective comparison with our model.

B. Comparison with the Nyström method

The Nyström method is a technique for finding numerical approximations of eigenfunctions. It has been proposed in [19] to reduce the computational burden in spectral clustering eigenvalue problems. In fact, it allows one to extrapolate the complete grouping solution using only a small subset extracted randomly from the whole dataset to partition. From Table III we can notice that the kernel spectral clustering model on average performs better, in terms of ARI index, than the Nyström method for all the graphs but the network Art_22C. In this case, however, the Nyström method needs at least 5 000 randomly selected nodes in the initial working subset in order to perform the clustering task. On the other hand less nodes are required by the active selection technique to obtain a good subgraph. These observations require additional investigations to be properly understood. Finally, our algorithm is faster and in our opinion this is because the major bottleneck of the Nyström method is that two matrix inversion operations are involved. This is not the case for the kernel spectral clustering model.

C. Complexity

Finally, a further analysis related to the computation time required by the kernel spectral clustering model is shown in Figure 10: the computational complexity seems to be $O(N^2)$, where N indicates the number of nodes of the whole graph. However, it mainly depends on the number of training nodes. For other networks it can happen that less nodes are needed in the training subgraph in order to obtain a good clustering in the test stage, thus reducing the computational complexity. This performance can be considered already good and competitive with many of the best existing algorithms for community detection (see for example section 3.3 in [20]).

Synthetic Network	**Size training set (nodes)**	**ARI**	**CPU time (s)**
Art_9C	300	0.99	8.28
Art_13C	1 000	0.98	187.03
Art_22C	2 500	0.71	9778.57
Real Network	**Size training set (nodes)**	**Modularity**	**CPU time (s)**
Yeast_pro	1 057	0.39	4.93
Power grid	988	0.54	21.90

TABLE II
RESULTS OF THE KERNEL SPECTRAL CLUSTERING MODEL.

Synthetic Network	**Size training set (nodes)**	**ARI**	**CPU time (s)**
Art_9C	300	0.85 ± 0.10	7.82
Art_13C	1 000	0.87 ± 0.07	457.29
Art_22C	5 000	0.79 ± 0.06	14 889.68

TABLE III
RESULTS OF THE NYSTRÖM METHOD USING 10 RANDOMIZATIONS OF THE INITIAL WORKING SUBSET. THE AVERAGE VALUE AND THE ASSOCIATED STANDARD DEVIATION ARE SHOWN. THE CPU TIME IS RELATED TO A SINGLE RUN.

D. Final comments

As a general comment, we can say that the results shown in Table II are good. Indeed, the quality of the partitions predicted by the kernel spectral clustering model is meaningful

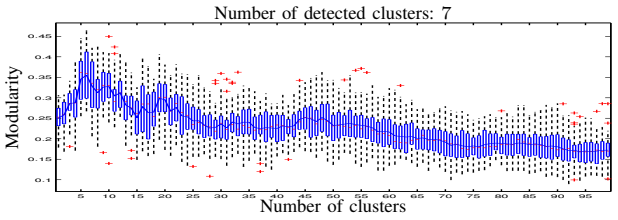


Fig. 6. Validation procedure for the protein interaction network.

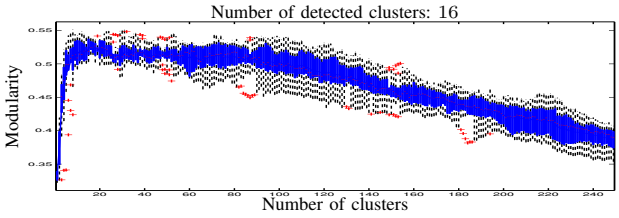


Fig. 7. Validation procedure for the power grid network.

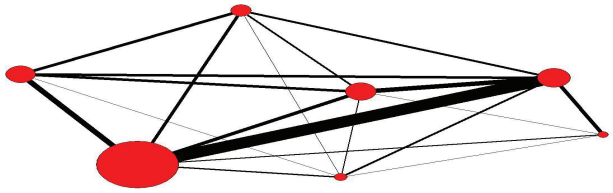


Fig. 8. Partition discovered by the kernel spectral clustering model for the network of interacting proteins of yeast. Every circle represents a cluster with size related to the number of nodes belonging to it. The position of the circles is not relevant. The edges are the links between nodes belonging to different communities, with thickness proportional to the number of these intercommunity edges. The nodes and edges in each detected community, for simplicity, are not shown. The Modularity corresponding to this partition is 0.39, meaning that this community structure can be considered meaningful. Finally, the figure has been made by using the software for large network analysis `Pajek` (see *http://pajek.imfm.si/doku.php*).

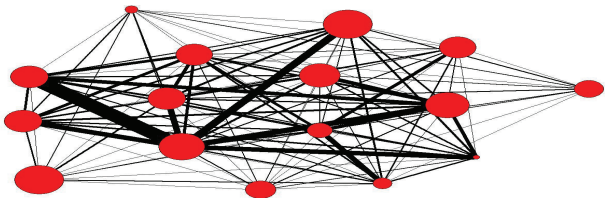


Fig. 9. Community structure of the Western USA power grid discovered by the kernel spectral clustering model. The comments made for Figure 8 are still valid here. The Modularity statistic related to this depicted partition is 0.54.

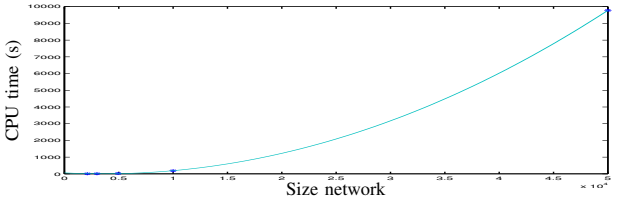


Fig. 10. Computational complexity of the kernel spectral clustering model for the networks analyzed in this paper.

according to the related values of the ARI index or the Modularity statistic. Moreover we can see that on average our model performs better than the Nyström method for all the synthetic graphs but the network Art_22C. Probably the structure of this graph is such that a random subsampling of the nodes is the best strategy for selecting a good working set. However, extra research work is needed to understand this last issue. Finally, the computation time is reasonable and can be potentially improved by a factor 50 or more (see end of section IV-C). Furthermore, it is lower than the Nyström technique for the largest graphs.

1) Synthetic networks: The agreement between the true memberships and the partitions predicted by the kernel spectral clustering model is good for all the cases. Moreover, the results are obtained by using a small training subgraph, selected to be representative of the overall community structure characterizing the entire network.

2) Yeast_pro: Unlike the analysis performed in [17], we consider both direct and indirect interactions, for a total of $2\,114$ nodes instead of $1\,870$. We found 7 clusters, with the largest component containing about the 60% of the linked proteins (and all the proteins with indirect interactions). This outcome is shown in Figure 8 (only the directly interacting proteins are considered). The community structure found by our algorithm seems quite attractive for its simplicity, but needs further investigation in order to assess a meaningful biological interpretation of the discovered modules.

3) Power grid: The main difference with the analysis performed in [4] is that now we have a representative training set of the whole graph and if in future more nodes will be added to the network, to infer their membership we just have to perform the out-of-sample extension based on this training set. Moreover, here we test the ability of our model on the test set represented by the entire network, obtaining a rather good result in terms of the Modularity of the discovered partition. The latter is depicted in Figure 9.

VIII. Conclusions

In this paper we applied the kernel spectral clustering model described in [2] and the model selection procedure proposed in [4] in order to analyze large unweighted networks, real and artificial. Moreover, we proposed a method to draw from the whole network a subgraph representative of the community structure of the entire network, based on the greedy optimization of the expansion factor [7]. This actively selected subgraph is then used as training set in the learning process of the kernel machine.

The results obtained, summarized in Table II, are good both in terms of the quality of the discovered partitions and the computation time needed to perform the analysis. Future challenges may relate to on-line learning and very large scale problems.

Acknowledgements

Research supported by:Research Council KUL: GOA/11/05 Ambiorics, GOA/10/09 MaNet , CoE EF/05/006 Optimization in Engineering (OPTEC) en PFV/10/002 (OPTEC), IOF-SCORES4CHEM, several PhD/postdoc & fellow grants; Flemish Government:FWO: PhD/postdoc grants, projects: G0226.06 (cooperative systems and optimization), G0321.06 (Tensors), G.0302.07 (SVM/Kernel), G.0320.08 (convex MPC), G.0558.08 (Robust MHE), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine) research communities (WOG: ICCoS, ANMMM, MLDM); G.0377.09 (Mechatronics MPC); IWT: PhD Grants, Eureka-Flite+, SBO LeCoPro, SBO Climaqs, SBO POM, O&O-Dsquare;Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, Dynamical systems, control and optimization, 2007-2011);IBBT;EU: ERNSI; FP7-HD-MPC (INFSO-ICT-223854), COST intelliCIS, FP7-EMBOCON (ICT-248940), FP7-SADCO (MC ITN-264735), ERC HIGHWIND (259 166);Contract Research: AMINAL;Other:Helmholtz: viCERP,ACCM Rocco Langone is a doctoral student at the K.U. Leuven, Belgium. Carlos Alzate is a postdoctoral fellow of the Research Foundation - Flanders (FWO). Johan Suykens is a professor at the K.U.Leuven, Belgium. The scientific responsibility is assumed by its authors.

References

[1] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[2] C. Alzate and J. A. K. Suykens, "Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 335–347, February 2010.

[3] I. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors a multilevel approach," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 11, pp. 1944 –1957, nov. 2007.

[4] R. Langone, C. Alzate, and J. A. K. Suykens, "Modularity-based model selection for kernel spectral clustering," in *Proc. of the International Joint Conference on Neural Networks (IJCNN 2011)*, 2011, pp. 1849–1856.

[5] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Natl. Acad. Sci. USA*, vol. 103, no. 23, pp. 8577–8582, 2006.

[6] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[7] A. S. Maiya and T. Y. Berger-Wolf, "Sampling community structure," in *Proc. 19th ACM Intl. Conference on the World Wide Web (WWW '10)*, 2010.

[8] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.

[9] J. Leskovec, K. J. Lang, and M. W. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proc. of WWW 2010)*, 2010, pp. 631–640.

[10] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, pp. 849–856.

[11] F. R. K. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[12] C. Alzate and J. A. K. Suykens, "Out-of-sample eigenvectors in kernel spectral clustering," in *Proc. of the International Joint Conference on Neural Networks (IJCNN 2011)*, 2011, pp. 2349–2356.

[13] J. Q. Jiang, A. W. Dress, and G. Yang, "A spectral clustering-based framework for detecting community structures in complex networks," *Applied Mathematics Letters*, vol. 22, no. 9, pp. 1479–1482, 2009.

[14] Y. Kang and S. Choi, "Kernel PCA for community detection," in *Business Intelligence Conference*, 2009.

[15] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 1, no. 2, pp. 193–218, 1985.

[16] A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," *Phys. Rev. E*, vol. 80, no. 1, p. 016118, Jul 2009.

[17] A.-L. Barabasi, S. P. Jeong, H. Mason, and Z. N. Oltvai, "Lethality and centrality in protein networks," *Nature*, vol. 6833, no. 411, pp. 41–42, 2001.

[18] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, no. 393, pp. 440–442, 1998.

[19] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the Nyström method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 214–225, Feb. 2004.

[20] Z. Lv, D. Chen, M. Shang, and Y. Fu, "Detecting overlapping communities of weighted networks via a local algorithm," *Physica A*, no. 389, pp. 4177–4187, 2010.