Community detection using Kernel Spectral Clustering with memory

This content has been downloaded from IOPscience. Please scroll down to see the full text.

2013 J. Phys.: Conf. Ser. 410 012100
(http://iopscience.iop.org/1742-6596/410/1/012100)

View the table of contents for this issue, or go to the journal homepage for more

Download details:

IP Address: 52.27.132.29
This content was downloaded on 08/07/2015 at 10:09

Please note that terms and conditions apply.
Community detection using Kernel Spectral Clustering with memory

Rocco Langone and Johan A.K. Suykens
Kasteelpark Arenberg 10, 3001, Leuven (Belgium)
E-mail: {rocco.langone, johan.suykens}@esat.kuleuven.be

Abstract. This work is related to the problem of community detection in dynamic scenarios, which for instance arises in the segmentation of moving objects, clustering of telephone traffic data, time-series micro-array data etc. A desirable feature of a clustering model which has to capture the evolution of communities over time is the temporal smoothness between clusters in successive time-steps. In this way the model is able to track the long-term trend and in the same time it smooths out short-term variation due to noise. We use the Kernel Spectral Clustering with Memory effect (MKSC) which allows to predict cluster memberships of new nodes via out-of-sample extension and has a proper model selection scheme. It is based on a constrained optimization formulation typical of Least Squares Support Vector Machines (LSSVM), where the objective function is designed to explicitly incorporate temporal smoothness as a valid prior knowledge. The latter, in fact, allows the model to cluster the current data well and to be consistent with the recent history. Here we propose a generalization of the MKSC model with an arbitrary memory, not only one time-step in the past. The experiments conducted on toy problems confirm our expectations: the more memory we add to the model, the smoother over time are the clustering results. We also compare with the Evolutionary Spectral Clustering (ESC) algorithm which is a state-of-the art method, and we obtain comparable or better results.

1. Introduction
Community detection of evolving networks recently received much attention in the science community [2, 3, 4, 5, 6]. We aim to cluster networks whose community structure change over time and we wish to obtain a meaningful clustering result over the whole time range taken into account. To this purpose we use the kernel spectral clustering with memory effect (MKSC) model, which is based on a Least Squares Support Vector Machines (LSSVM) [7] formulation. We incorporate the temporal smoothness [8] between consecutive partitioning as a prior knowledge at the primal level, resulting in the dual problem to a set of linear systems. Moreover, the out-of-sample extension is a key feature of our model: we can predict the membership of a new node thanks to the model learned during the training phase. Finally, a systematic model selection scheme is used in order to carefully tune all the necessary parameters.

This paper is organized as follows. In Section 2 we summarize the basic MKSC model (i.e. with one snapshot of memory) and we describe how we can add more memory, which represent the new contribution present in this work. Section 3 presents some simulation results on 4 toy problems. We compare the results obtained by the MKSC model (considering different amounts of memory) with the kernel spectral clustering (KSC, see [9]) applied separately on each snapshot and the evolutionary spectral clustering (ESC) algorithm [10], which is a state-of-the art method for clustering evolving graphs. Finally in Section 4 we draw some conclusions.
2. MKSC with arbitrary memory
The primal problem of the MKSC model has been stated as follows [11]:
\[
\min_{\mathbf{w}^{(l)}, \mathbf{e}^{(l)}, b_l} \frac{1}{2} \sum_{l=1}^{k-1} \mathbf{w}^{(l)T} \mathbf{w}^{(l)} - \frac{\gamma}{2N} \sum_{l=1}^{k-1} \mathbf{e}^{(l)T} D_{\text{Mem}}^{-1} \mathbf{e}^{(l)} - \nu \sum_{l=1}^{k-1} \mathbf{w}^{(l)T} \mathbf{w}_{\text{old}}^{(l)}
\]
\[
\text{such that } \mathbf{e}^{(l)} = \Phi \mathbf{w}^{(l)} + b_l \mathbf{1}_N,
\]
where \(\mathbf{e}^{(l)} = [e_1^{(l)}, \ldots, e_N^{(l)}]^T\) are the projections, \(l = 1, \ldots, k - 1\) indicates the score variables needed to encode the \(k\) clusters to find, \(D_{\text{Mem}}^{-1} \in \mathbb{R}^{N \times N}\) is a weighting matrix, \(\Phi\) is the \(N \times d\) feature matrix \(\Phi = [\varphi(x_1)^T; \ldots; \varphi(x_N)^T]\), \(\nu\) and \(\gamma\) are regularization constant. Here \(w_{\text{old}}^{(l)} = \sum_{i=1}^{M} w_{\text{prev},i}^{(l)} = \sum_{i=1}^{M} \Phi_{\text{prev},i}^{(l)} \mathbf{e}^{(l)}\) represents the \(l\)th binary models developed for the \(M\) previous snapshots of the evolving network, not only one snapshot, and the MKSC model in the original MKSC model proposed in [11]. The term \(w_{\text{old}}^{(l)} w_{\text{old}}^{(l)}\) describes the correlation between the actual expansion and the previous model, which we want to minimize. In this way we are able to introduce temporal smoothness in our formulation, such that the current partition does not deviate too much from the recent history. In particular, the more memory we add (i.e. the more \(M\) increases) the smoother will be the clustering results. The clustering model is expressed by:
\[
e^{(l)} = w^{(l)T} \varphi(x_i) + b_l, i = 1, \ldots, N.
\]
where \(\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^{dN}\) 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 \(\text{sign}(e_i^{(l)})\). The binary indicators are combined to form a codebook \(CB = \{e_p\}_{p=1}^{k}\), where each codeword is a binary string of length \(k - 1\) representing a cluster.

The out-of-sample extension to new nodes is done by an Error-Correcting Output Codes (ECOC) decoding procedure.

The dual model is:
\[
\left(D_{\text{Mem}}^{-1} M_{\text{Mem}} \Omega - \frac{I}{\gamma} \alpha^{(l)}\right) = -\nu D_{\text{Mem}}^{-1} M_{\text{Mem}} \Omega_{\text{new-old}} \alpha^{(l)} = \nu D_{\text{Mem}}^{-1} M_{\text{Mem}} \sum_{i=1}^{M} \Omega_{\text{new-prev},i} \alpha^{(l)}_{\text{prev},i}.
\]
where \(\Omega\) is the kernel matrix with \(ij\)-th entry \(\Omega_{ij} = K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)\), \(\Omega_{\text{new-old}}\) captures the similarity between the nodes of the current graph and the nodes of each of the previous \(M\) snapshots, \(\alpha^{(l)}\) are dual variables, \(K : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+\) is the kernel function, \(M_{\text{Mem}}\) is the centering matrix equal to \(M_{\text{Mem}} = I_N - \frac{1}{N_{\text{Mem}}} 1_N 1_N^T\).

3. Simulation Results
We conducted experiments on the following toy problems, related to both binary and multiway clustering:

- **two moving Gaussians**: from time-steps 1 to 10 two Gaussian clouds move towards each other until they overlap almost completely. This situation can be also described by considering the points of the 2 Gaussians as nodes of a weighted network where the weights of the edges change over time.
- **three moving Gaussians**: three merging Gaussian clouds.
- **switching network**: we build up 9 snapshots of a network of 1000 nodes formed by 2 communities. At each time-step some nodes switch their membership between the two clusters (the software described in [12] is used to generate this benchmark).
- **expanding/contracting network**: a network with 5 communities experiences over-time 24 expansion events and 16 contractions of its communities.

For the two moving Gaussians data-set, in Fig. 1 we show how increasing the memory of the MKSC model can give smoother and better clustering results over time, measured in terms of Normalized Mutual Information (NMI) between consecutive partitioning and the smoothed ARI ([11]). We also compare with the Evolutionary Spectral
Clustering (ESC) algorithm [10] which is a state-of-the-art method for community detection of evolving graphs. We obtain comparable or better performances, depending on the amount of memory added to our model. For what concerns the switching network (see Fig. 2), the NMI plot shows, as we expect, that the more memory we add the more similar are the clustering results over time. Moreover it seems that one snapshot of memory is enough to have good performances ($MKSC_{M1}$ performs better of all the other models). Finally in Table 1 we summarize the results obtained on all the data-sets, showing also the memory requirement and the computation time.

![Figure 1](image1.png)

**Figure 1.** First row: performance of MKSC with 1, 2 and 3 snapshots of memory ($MKSC_{M1}$, $MKSC_{M2}$, $MKSC_{M3}$). KSC and ESC in the two moving Gaussian experiment in terms of the the NMI between consecutive partitions (left) and the smoothed ARI index [11] (right). The smoothed ARI plot and the NMI trend tell us that, as expected, the models with temporal smoothness are more able than KSC to produce clustering results which are more similar to the ground truth and also more consistent and smooth over time. Moreover the more memory we add to the MKSC model, the better the results. In the second row we show the true partitioning (left) and the prediction of $MKSC_{M3}$ (right), where only snapshots 4, 6 and 9 are depicted.

![Figure 2](image2.png)

**Figure 2.** Performance of MKSC with 1, 2 and 3 snapshots of memory ($MKSC_{M1}$, $MKSC_{M2}$, $MKSC_{M3}$). KSC and ESC for the switching network data-set in terms of the the NMI between consecutive partitions, the smoothed ARI index and the smoothed Modularity [11]. The NMI plot shows, as we expect, that the more memory we add the more consistent are the clustering results over time. Moreover $MKSC_{M1}$ performs better of all the other models in terms of smoothed ARI and smoothed Modularity.

4. Conclusions

In this paper we presented an extension of our previous work introduced in [11]. In the latter we proposed the kernel spectral clustering with memory effect (MKSC), a clustering model in the Least Squares Support Vector Machines (LS-SVM) framework with primal-dual formulation and out-of-sample extension. The MKSC model is useful to cluster evolving networks (represented as a sequence of graphs), where a desirable feature of the clustering results over time is their consistency and smooth evolution. This temporal smoothness has been introduced in the primal formulation via a memory term of one snapshot in the past. In this work we explored the consequence of increasing this memory to more than one snapshot. We tested the new model on 4 toy problems. We observed that in some cases we can obtain clusters of better quality and more consistent over time by adding more memory to the basic MKSC model.
Table 1. Summary of the results for the 4 data-sets (top row refers to the Gaussians and bottom row to the synthetic networks). For each evaluation measure the values represent an average over time (i.e. the mean value per snapshot). Regarding the MKSC model, between parenthesis we indicate with which amount of memory we obtained the best results ($M_1$, $M_2$, or $M_3$).

| EVALUATION MEASURE | MKSC | ESC |
|---------------------|------|-----|
| Smoothed ARI        | 0.88 | 0.87|
| NMI                 | 0.86 | 0.85|
| CPU time (s)        | 0.53 | 0.39|
| Memory requirement (s) | 0.29 | 2.92|
| Smoothed Modularity | 0.36 | 0.31|
| NMI                 | 0.73 | 0.73|
| CPU time (s)        | 52.23| 5.56|
| Memory requirement (s) | 4.31 | 3.06|

| EVALUATION MEASURE | MKSC | ESC |
|---------------------|------|-----|
| Smoothed ARI        | 0.99 | 0.91|
| Smoothed Modularity | 0.36 | 0.31|
| NMI                 | 0.73 | 0.73|
| CPU time (s)        | 51.83| 5.08|
| Memory requirement (s) | 2.57 | 2.79|

Acknowledgements
This work was supported by Research Council KUL: ERC AdG A-DATADRIVE-B, GOA/1105 Antimicrobics, GOA/1009 MaNet, CoE EF/05/006 Optimization in Engineering(OPTEC), IOF-SCORE54CHEM, several PhD/postdoc fellow grants, Flemish Government/FWO. PhD/postdoc grants projects: G0226.06 (cooperative systems and optimization), G0321.06 (Tensorcs), G.0302.07 (SVM/Kernel), G.0320.08 (convex MPC), G.0558.08 (Robust MHE), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine) G.0377.12 (structured models) research communities (WOG:ICCoS, ANMMM, MLDM); G.0302.07 (SVM/Kernel), G.0320.08 (convex MPC), G.0558.08 (Robust MHE), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine) G.0377.12 (structured models) research communities.

References
[1] Fortunato S 2010 Physics Reports 486 75–174
[2] Mucha P J, Richardson T, Macon K, Porter M A and Onnela J P 2010 Science 328 876–878
[3] Lin Y, Chi Y, Zhu S, Sundaram H and Tseng B L 2009 ACM Trans. Knowl. Discov. Data 3
[4] Bóta A, Krész M and Pluhár A 2011 Acta Cybernetica 20 35–52
[5] Asur S, Parthasarathy S and Ucar D 2009 ACM Trans. Knowl. Discov. Data 3 16:1–16:36
[6] Pallá G, Iszl Barábi A, Vicsek T and Hungary B 2007 Nature 446 79–81
[7] Suykens J A K, Van Gestel T, De Brabanter J, De Moor B and Vandewalle J 2002 Least Squares Support Vector Machines (World Scientific, Singapore)
[8] Chakrabarti D, Kumar R and Tomkins A 2006 Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining KDD ’06 (New York, NY, USA: ACM) pp 554–560 ISBN 1-59593-339-5
[9] Alzate C and Suykens J A K 2010 IEEE Transactions on Pattern Analysis and Machine Intelligence 32 335–347
[10] Chi Y, Song X, Zhou D, Hino K and Tseng B L 2007 KDD pp 153–162
[11] Langone R, Alzate C and Suykens J A K 2012 Submitted
[12] Greene D, Doyle D and Cunningham P 2010 Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining ASONAM ’10 (Washington, DC, USA: IEEE Computer Society) pp 176–183