FAST SELF LEARNING SPECTRAL CLUSTERING

Aleksandar Trokicić[1], **Branimir Todorović**[1]

[1] University of Niš, Faculty of Sciences and Mathematics, 33 Višegradska Street, 18000 Niš
Email: aleksandar.trokicic@pmf.edu.rs branimirtodorovic@pmf.edu.rs

Abstract

The paper considers the problem of spectral clustering (both constrained and unconstrained) and how to incorporate self-learning into it. We perform clustering iteratively by adding new constraints derived from the clustering results in the previous iterations. To our knowledge, this is the only self-learning algorithm that uses the results of the clustering algorithm directly as constraints. Additionally, it is the only algorithm with linear time and memory complexity that solves this problem. The results of experiments on real data confirm the theoretical findings.

Key words: self-learning, spectral clustering, constrained clustering, multi-layer graph

1 Introduction

The goal of a clustering algorithm is to partition the input data set into groups called clusters based on a similarity measure, so that the similarity of examples within groups is maximized and the similarity of examples from different groups is minimized. Therefore, a clustering problem can be defined as a minimization of a cost function, which penalizes the dissimilarity within clusters and the similarity between different clusters. In this paper, we will consider the problem of spectral clustering, where the cost function can be defined as a relaxation of the normalized cut [5] which only uses the spectrum (eigenvalues and eigenvectors) of a graph matrix to derive clusters.

The spectral clustering algorithm offers several advantages compared to the other clustering algorithms: simple cost function with an analytic solution, easy implementation and the ability to derive arbitrarily shaped clusters. However, spectral clustering has the squared time complexity with respect to the number of examples and it heavily and solely relies on the quality of the similarity measure.

Several approaches have been suggested to mitigate those problems. In order to reduce (linearize) the time complexity of the algorithm, the application of the Nystrom method [3] is suggested. Incorporating oracle knowledge (such as user guide that represents constraints) can reduce reliance on the similarity measure. This has been studied in literature under the definition of constrained spectral clustering [6]. In this paper, we will improve the algirithm from [4] the time complexity of which is a lot lower compared to its competitors. In [4] the set of input examples and the set of constraints are coded as a multi-layer graph and the solution is defined as unconstrained spectral clustering on the multi-layer graph, where a multi-layer graph is a set of graphs with a common vertex set.

However, access to a good oracle knowledge is usually rare. Because of this, methods of its approximation are developed, and therefore those methods will be used to learn additional constraints. Those methods belong to a class of self-learning clustering algorithms. To our knowledge, the only algoirithm that incorporates self-learning with

spectral clustering is found in [7]. It is based on a matrix completion method. However, the time and memory complexity of this algorithm is squared in the number of instances and therefore intractable for large data sets. Therefore, in this paper we will consider the fast self-learning spectral clustering with low time complexity. Our algorithm iteratively learns new constraints and uses them to improve clustering. Based on previous research, this is the first self-learning spectral clustering algorithm with the time complexity linear in the number of items no matter the number of constraints.

2 Algorithm

Spectral clustering belongs to a class of graph clustering algorithms, and therefore the input set of that algorithm is a graph $G = (V, W)$ (or a multi-layer graph) and a number of desired clusters k. Set $V = \{v_1, v_2, \ldots, v_n\}$ is the set of n graph vertices, and the matrix W is the similarity (weight) matrix. The degree matrix is a diagonal matrix such that the degree of vertex v_i is on the i-th place on the diagonal.

Graph clustering represents a vertex partition that satisfies some condition (minimization of the sum of edges within the partition and maximization of the sum of edge weights between different partitions). Spectral clustering satisfies these conditions by finding the orthogonal matrix (*spectral embedding*) $U \in \mathbf{R}^{n\times k}$ that minimizes the $\mathrm{tr}(U^T L U)$, where the matrix L is called normalized Laplacian $L = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$. Cluster assignments are derived from a matrix U via the k-means algorithm on the normalized rows of U.

A multi-layer graph with m layers is a set of m individual graphs $M_G = \{(G_i = (V, E_i, W_i))_{i\leq m}\}$ with the same set of vertices. For each layer a Laplacian matrix L_i and its corresponding spectral embeding $U_i \in \mathbf{R}^{n\times k}$ are computed. According to [1] spectral clustering on a multi layer graph is equivalent to the ordinary spectral clustering where instead of a Laplacian matrix we are using a modified Laplacian matrix $L_{mod} = \sum_{i=1}^{m} L_i - \alpha \sum_{i=1}^{m} U_i U_i^T$ (α is a regularization parameter).

We shall now review constrained spectral clustering on graphs. In this case, input is a graph $G = (V, W)$ and a set of constraints $\mathcal{PC}$. A constraint will be written in the following form $e = \{v_i, v_j, t, type\}$. A number $t \in (0, 1)$ is the constraint weight. A type $type \in \{ML, CL\}$ of constraint is either *must link* (vertices must be in the same cluster) or *cannot link* (vertices cannot be in the same cluster).

A solution (from [4]) for the problem of constrained spectral clustering is an unconstrained spectral clustering on a three-layer graph $M_G = \{(G_i = (V, E_i, W_i))_{i\leq 3}$ with different adjacency matrices. In the first layer we have an input graph $W_1 = W$, the second layer is *must link* and the third one is *cannot link* layer:

$$W_2(i,j) = \begin{cases} t, & \text{if } \{v_i, v_j, 1, type = ML\} \in \mathcal{PC} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

$$W_3(i,j) = \begin{cases} 0, & \text{if } \{v_i, v_j, 1, type = CL\} \in \mathcal{PC} \\ t, & \text{otherwise} \end{cases} \tag{2}$$

The constrained problem is defined as unconstrained on M_G. However, in order for time complexity to be linear in n, the Nystrom [3] method is applied.

2.1 Self-learning spectral clustering

In this section we will explain our formulation of self-learning spectral clustering, and show the algorithm we have created to solve this problem.

There are two cases that are presented in this research, the case of an unconstrained spectral clustering (input is a graph $G = (V, W)$) and the case of a constrained spectral clustering (input is a three layer graph that encodes the constraints M_G). If, in the unconstrained case, we encode the input as a multi layer graph, only the first layer would not be empty so M_G would indicate the same clustering as G. Therefore, in the following text, when we discuss an input graph we are referring to a three layer graph $M_G = \{(G_i = (V, E_i, W_i))_{i \leq 3}$.

Simply formulated, a self-learning algorithm learns a set of additional information (such as constraints) that improves the learning of a target task. We will use results of a clustering as additional constraints, and our algorithm will be iterative. In contrast to the algorithm from [7] which uses the results of a clustering to find a constraint matrix of low rank, our algorithm will use the results of a clustering as additional layers in the multi layer graph without any additional assumptions.

In our case the first layer is derived from the input set, the second and the third layers are derived from the set of constraints, and additional layers will be self learned and derived from the clustering results. Therefore, the first three layers should be more trustworthy than the following layers. So, we will change the modified Laplacian slightly: $L_{mod} = \sum_{i=1}^{m} \alpha_i (L_i - \alpha U_i U_i^T)$, where $\alpha_i \in (0, 1)$ are additional weights that will determine the trustworthiness of the layer.

A discussion of the graph layers follows: Initially, we have 3 layers: the input graph, the must link layer, and the cannot link layer. Then we perform a clustering and, for every pair of vertices (v_i, v_j) from the same cluster, we put $W_{4ij} = 1$ and otherwise 0.

Now spectral clustering is performed again on the graph containing only the first 3 layers and the results are similarly added into the multi layer graph as a 5th layer, and so forth. This process is iteratively performed t times, and the final clustering is the solution of the clustering the entire multi layer graph. The time complexity of this algorithm (with the application of the Nystrom method) is $O(t(nk^2 + t_{km}))$, and the memory complexity is $O(tnl)$. To our knowledge this is the only self-learning spectral clustering that achieves the linear time complexity.

3 Experimental results

We evaluate the performance of our algorithm on real world data sets: Postures and Covtype [1] and Orange[2](only attributes without missing data) data set. The adjacency matrix is formed using a RBF kernel $k(x, y) = \exp(-\frac{||x-y||^2}{2})$. Constraints are derived from a random subset A of input feature vectors for which we assume a complete knowledge. If items $v_i, v_j \in A$ belong to the same class then we add $\{v_i, v_j, t, type = ML\}$ to the $\mathcal{PC}$, otherwise we add $\{v_i, v_j, t, type = CL\}$ to the $\mathcal{PC}$. We evaluated the results using a Rand Index $RI = \frac{tp+tn}{tp+fp+tn+fn}$ where tp, fp, tn, fn stand for *true positive*, *false positive*, *true negative* and *false negative*, respectively. We compare our algorithm **F**ast **S**elf-**L**earning - **S**pectral **C**lustering) with the algorithm **C**onstrained **Sp**ectral - **M**ulti**L**ayer) as well as the **S**pectral **l**earning algorithm from [7] (which is an extension of an algorithm from [2]) and present the result in the following tables. Note that the

[1] http://archive.ics.uci.edu/ml

[2] http://www.causality.inf.ethz.ch/activelearning.php

Table 1: Performance comparison of algorithms **csp-ml**, **fsl-sc** and **sl** on the following real world data sets: Postures, Orange and Covtype. We report an average rand index and its variance.

Data set	Known Label Percentage	csp-ml	fsl-sc	sl
Postures	*No constraints*	0.6907 ± 0.0095	0.6927 ± 0.0031	0.6907 ± 0.0095
	6	0.6920 ± 0.0127	0.6969 ± 0.0113	0.2565 ± 0.0019
	10	0.7004 ± 0.0106	0.7130 ± 0.0079	0.3530 ± 0.002
	15	0.7099 ± 0.0120	0.7298 ± 0.0007	0.4250 ± 0.0038
Orange	*No constraints*	0.4940 ± 0.0018	0.4959 ± 0.0002	0.4940 ± 0.0018
	6	0.4962 ± 0.0033	0.4990 ± 0.0030	0.4950 ± 0.0081
	10	0.5073 ± 0.0059	0.5158 ± 0.0030	0.5061 ± 0.0092
	15	0.5202 ± 0.0065	0.5322 ± 0.0013	0.5294 ± 0.0154
Covtype	*No constraints*	0.4807 ± 0.0137	0.4983 ± 0.0082	0.4807 ± 0.0137
	6	0.4968 ± 0.0029	0.5002 ± 0.0031	0.4565 ± 0.0644
	10	0.5137 ± 0.0178	0.5311 ± 0.0211	0.4909 ± 0.0519
	15	0.5367 ± 0.0199	0.5672 ± 0.0191	0.5351 ± 0.0521

main algorithm from [7] is prohibitive on thess data sets because of its memory requirement. We show that self-learning new constraints improves the results. New constraints in the algorithm **fsl-sc** are learned in 10 iterations. Results are presented in *table (1)*.

4 Conclusions

We implemented a new algorithm for self-learning spectral clustering, which can be used for both the unconstrained and constrained spectral clustering. Firstly note that the performance is improved as we add more constraints, which is one of the requirements of constrained clustering. Additionally, our algorithm outperformed its competitors across all data sets. We deduce that the largest margin between our algorithm and its competitors is achieved with a larger number of constraints. That means that our algorithm is able to better utilize the constraints compared to other algorithms. Furthermore, even in the case with no constraints we note that self-learning improved the results. Finally, our algorithm also has a smaller variance.

To our knowledge this is the only self-learning spectral clustering algorithm with linear time and memory complexity.

References

[1] Dong, X., Frossard, P., Vandergheynst, P., Nefedov, N.: Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds. IEEE Transactions on Signal Processing 62(4), 905–918 (2014)

[2] Kamvar, S.D., Klein, D., Manning, C.D.: Spectral learning. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence. pp. 561–566 (2003)

[3] Kumar, S., Mohri, M., Talwalkar, A.: Sampling techniques for the Nyström method. In: Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics. pp. 304–311 (2009)

[4] Trokicić, A., Todorović, B.: Constrained spectral clustering via multilayer graph embeddings on a grassmann manifold. International Journal of Applied Mathematics and Computer Science 29(1), 125–137 (2019)

[5] Von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17(4), 395–416 (2007)

[6] Wang, X., Qian, B., Davidson, I.: On constrained spectral clustering and its applications. Data Mining and Knowledge Discovery 28(1), 1–30 (2014)

[7] Wang, X., Wang, J., Qian, B., Wang, F., Davidson, I.: Self-taught spectral clustering via constraint augmentation. In: Proceedings of the 2014 SIAM International Conference on Data Mining. pp. 416–424. SIAM (2014)