Active Spectral Clustering

Xiang Wang
Department of Computer Science
University of California, Davis
Davis, CA 95616
xiang@ucdavis.edu

Ian Davidson
Department of Computer Science
University of California, Davis
Davis, CA 95616
davidson@cs.ucdavis.edu

***Abstract*—The technique of spectral clustering is widely used to segment a range of data from graphs to images. Our work marks a natural progression of spectral clustering from the original passive unsupervised formulation to our active semi-supervised formulation. We follow the widely used area of constrained clustering and allow supervision in the form of pairwise relations between two nodes: `Must-Link` and `Cannot-Link`. Unlike most previous constrained clustering work, our constraints are specified incrementally by querying an oracle (domain expert). Since in practice, each query comes with a cost, our goal is to maximally improve the result with as few queries as possible. The advantages of our approach include: 1) it is principled by querying the constraints which maximally reduce the expected error; 2) it can incorporate both hard and soft constraints which are prevalent in practice. We empirically show that our method significantly outperforms the baseline approach, namely constrained spectral clustering with randomly selected constraints, on UCI benchmark data sets.**

***Keywords*-spectral clustering; active learning; constrained clustering**

I. Introduction

Many real-world applications, such as image segmentation, social network analysis and data clustering can be abstracted into a graph partition problem: finding the normalized min-cut (Ncut) of a given graph. Although the Ncut problem is generally intractable, it is well-known that its relaxed form can be solved by spectral clustering. The seminal work by Shi and Malik [1] represents the first incarnation of spectral clustering which was passive and unsupervised. However, in many application domains, considerable domain expertise exists and encoding domain knowledge into clustering algorithms is important if the results are to be novel and actionable. To address this issue, user supervision in the form of pairwise relations between two nodes of the graph have been proposed [2]: `Must-Link` (they belong to the same side of the cut) and `Cannot-Link` (they belong to different sides of the cut) constraints. Constrained clustering, including constrained spectral clustering, has been extensively studied [2] and much previous work [3]–[5] has shown that ML and CL constraints, *when used properly*, can greatly improve the quality of the resultant clustering. Those work represents the second incarnation of spectral clustering: passive and semi-supervised.

In this work we make the natural and important progression to the third incarnation of spectral clustering: active and semi-supervised. In this formulation the constraints are **provided incrementally after querying an oracle** rather than *a priori* in a batch before clustering begins. Active and semi-supervised spectral clustering has many natural problem settings that will see its wide-scale use:

- The constraints can be used as an explicit and elegant way to perform interactive transfer learning. In this setting the graph Laplacian is generated from the source domain and the constraints from the target domain. For example a given set of facial portraits may naturally cluster according to hair-style. The constraints (derived by a domain expert) are used to transfer this structure to a related but more complex domain such as identifying gender.
- The graph Laplacian can sometimes be noisy and inconsistent. This is particularly likely if it is constructed from complex objects such as images that are of poor quality, not completely aligned or taken with different cameras. The oracle then can be used to overcome these issues.
- Some problems in this domain naturally have a sequential aspect. Consider the clustering of a stream of video (Chapter 18, [2]), it makes most sense to specify constraints incrementally rather than all at once.
- Previous research [5] showed that in batch constrained clustering, not all given constraints are equally helpful/informative in terms of improving label purity in the clusters despite the constraints being generated from the **same labels used to measure performance**. We experimentally verify in section IV that incrementally collecting constraints produces better results.

Therefore, it is a natural and important question to ask: instead of passively taking a given set of constraints, which may consist of both helpful and harmful constraints, is it possible for the algorithm to actively query/fetch only the constraints that are expected to be helpful? With the availability of an oracle, we can put constrained spectral

clustering into the context of active learning [6]. Our goal is to maximally improve the quality of the resultant clustering (maximizing performance gain) while making as few queries as possible (minimizing cost). We propose an active learning framework for constrained spectral clustering, or active spectral clustering for short. Our framework consists of two key components:

- A constrained spectral clustering algorithm: This component takes the graph and the constraints that have already been queried as input. It produces a Ncut of the graph which satisfies the constraints we currently have.
- A query strategy: This component evaluates the current cut of the graph as well as the constraints we have, then decides the best constraint to query next, based on the principle of maximally reducing the expected error between our current cluster assignment and the expected groundtruth assignment.

We apply these two components alternatively and the resultant clustering will be refined over iterations and eventually converge to the groundtruth result.

Our contributions are:

1) This is the first principled framework for active spectral clustering. It provides a cost-efficient solution to many real-world applications that need to find the Ncut of a graph: it actively selects the most helpful constraints to query from an oracle and thus minimizes the efforts required from the oracle/domain experts.
2) We propose a ready-to-use active spectral clustering algorithm as a realization of our framework. The proposed algorithm is highly flexible and can deal with both hard and soft constraints. This is of great practical importance because soft constraints are common in many applications, especially when the oracle is not a single human expert, but a group of users who may give inconsistent answers to the same query [7], [8].
3) We address some important implementation issues for our method, such as dealing with outliers and runtime analysis. We also discuss the limitation of our method in its current form as well as future work.
4) We empirically show that our method significantly outperforms the baseline method typically used by practitioners. This is a significant result with practical implications. It means it is far better to incrementally and interactively specify constraints rather than getting them in one large batch.

The rest of the paper is organized as follows: related work is discussed in Section II; we propose our active spectral clustering framework in Section III; it is evaluated empirically in Section IV; implementation issues are discussed in Section V and future directions are discussed in Section VI; we conclude our work in Section VII.

II. Related Work

Active clustering is a special sub-category of active learning algorithms [6]. The difference is that active clustering algorithms query pairwise relations between two nodes instead of the labels of individual nodes. Most existing active clustering methods [3]–[5], [9]–[11] were built upon hard clustering schemes such as K-means clustering and hierarchical clustering. Little attention has been paid to the active learning framework for *spectral clustering*, which is the most popular soft clustering scheme and a solution to many real-world applications.

Xu et al. [12] proposed an active spectral clustering method that examines the eigenvectors of the graph Laplacian to identify boundary points and sparse points, and then queries the oracle for constraints among these ambiguous points. Their work is limited mainly because they explicitly assume that the underlying clusters in the data set are nearly separated and it is the boundary points that cause the inaccuracy in the cluster assignment; it is unclear if the propose method would still work otherwise. Also, since the proposed method is built upon the KKM constrained spectral clustering method [13], it can only incorporate hard constraints, not soft ones.

Active spectral clustering is also related to the area of matrix perturbation analysis for spectral clustering [14], [15], which studies how much the resultant clustering will change when a perturbation is applied to the original graph Laplacian. The results from perturbation analysis can give us an idea of how stable the clustering is and **how many constraints are needed** for the clustering to change significantly. However, the bounds in matrix perturbation theory are typically of the form involving the norm of the matrix and hence do not give directly suggestion on **what constraints we should query**.

III. Our Active Spectral Clustering Framework

In this section we present our active spectral clustering framework. We provide the background knowledge in Section III-A and an overview of our framework in Section III-B. We begin in Section III-C by introducing the first important component of our framework, a constrained spectral clustering algorithm that can handle both hard and soft constraints. Then in Section III-D we describe the second important component of our framework, an active query strategy based on maximum expected error reduction. Important notations used throughout the rest of the paper are listed in Table I. Note that the superscript "*" when attached to a symbol refers to the groundtruth answer typically only available to the oracle.

A. Background and Preliminaries

We have a graph $\mathcal{G}$ with N nodes. A is the associated affinity matrix. A is symmetric and nonnegative. D is the

Table I
TABLE OF NOTATIONS

Symbol	Meaning
A	The affinity matrix
D	The degree matrix
I	The identity matrix
L	The graph Laplacian
$Q^{(t)}$	The constraint matrix at time t
Q^*	The groundtruth (complete) constraint matrix
$\mathbf{u}^{(t)}$	The relaxed cluster indicator vector at time t
$\mathbf{u}^*$	The groundtruth cluster indicator vector

degree matrix of $\mathcal{G}$ where

$$D_{ij} = \begin{cases} \sum_{j=1}^{N} A_{ij} & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}.$$

$L = D - A$ is the graph Laplacian of $\mathcal{G}$.

It is well-known result [1] that spectral clustering finds the normalized minimum cut of $\mathcal{G}$ in its relaxed form:

$$\begin{aligned} \underset{\mathbf{u}\in\mathbb{R}^N}{\arg\min} \quad & \mathbf{u}^T L \mathbf{u}, \\ \text{s.t.} \quad & \mathbf{u}^T D \mathbf{u} = \text{vol}(\mathcal{G}),\ D\mathbf{u} \perp \mathbf{1}, \end{aligned} \tag{1}$$

where $\text{vol}(\mathcal{G}) = \sum_{i=1}^{N} D_{ii}$ and $\mathbf{u}$ is the relaxed cluster indicator vector. The optimal solution to Eq.(1) is the eigenvector associated with the second smallest eigenvalue of L. The actual 2-way cut is given by assigning nodes corresponding to the positive entries in $\mathbf{u}$ to one side and negative nodes to the other.

In practice, the graph Laplacian L is often generated with noise and/or from a biased sample of the underlying data distribution. As a result, the clustering found by (unconstrained) spectral clustering, $\mathbf{u}$, will differ from the groundtruth cluster assignment, $\mathbf{u}^*$. In order to find a $\mathbf{u}$ that better approximates the groundtruth result, a popular approach is to incorporate constraint information into spectral clustering. Formally, let $\mathscr{A}$ be a constrained spectral clustering algorithm:

$$\mathbf{u} \leftarrow \mathscr{A}(L, Q).$$

Now the resultant clustering $\mathbf{u}$ is decided by both the graph Laplacian L and a constraint matrix Q. In its generalized form, $Q \in \mathbb{R}^{N\times N}$ is a symmetric matrix that encodes pairwise relations between the nodes of $\mathcal{G}$ as follows:

- $Q_{ij} > 0$ means that node i and j should be assigned to the same cluster;
- $Q_{ij} < 0$ means that node i and j should be assigned to different clusters;
- $Q_{ij} = 0$ means that the relation between node i and j is unknown.

For hard constraints, we have $Q_{ij} = 1$ for `Must-Links` and $Q_{ij} = -1$ for `Cannot-Link`s. For soft constraints, the magnitude of Q_{ij} indicates how confident we are about that constraint. Previous work on constrained spectral clustering assumes the constraint matrix Q is provided *a priori*, i.e. the selection of known entries in Q is independent from the algorithm $\mathscr{A}$ and we have no control over the selection process.

B. An Overview of Our Framework

In this work, we make the assumption that there is an oracle who has access to the groundtruth constraint matrix $Q^* = \mathbf{u}^*\mathbf{u}^{*T}$, where $\mathbf{u}^* \in \mathbb{R}^N$ is the groundtruth cluster assignment. We assume that we can actively query an oracle about the value of any entry in Q^*, one entry at a time. Q is the matrix that contains all the constraints we have queried so far (0 for unknown entries). Note that the nonzero entries in Q is always a proper subset of the nonzero entries in Q^*. Our goal is to minimize the difference between $\mathbf{u} = \mathscr{A}(L, Q)$ and $\mathbf{u}^*$ using as few queries as possible.

We propose an iterative process to incrementally query the oracle about the constraint that can maximally reduce the expected error in our current result. We start with an empty constraint matrix $Q^{(0)}$ with all 0 entries, then we compute the current clustering using a constrained spectral clustering algorithm $\mathscr{A}$:

$$\mathbf{u}^{(0)} \leftarrow \mathscr{A}(L, Q^{(0)}).$$

Note that $\mathbf{u}^{(0)}$ should be the same as the clustering found by the unconstrained spectral clustering algorithm as we have no constraint so far. Then assume at iteration t we already have

$$\mathbf{u}^{(t)} \leftarrow \mathscr{A}(L, Q^{(t)}).$$

A query strategy $\mathscr{Q}$ will evaluate the current clustering $\mathbf{u}^{(t)}$ and the current constraints $Q^{(t)}$ to decide what is the next entry in Q^* we should query from the oracle. Let

$$(i, j) \leftarrow \mathscr{Q}(\mathbf{u}^{(t)}, Q^{(t)}),$$

we update $Q^{(t)}$ to $Q^{(t+1)}$ by filling in $Q_{ij}^{(t)}$ and $Q_{ji}^{(t)}$ with the value of Q_{ij}^* (since the constraint matrix is symmetric). Then we update the clustering by

$$\mathbf{u}^{(t+1)} \leftarrow \mathscr{A}(L, Q^{(t+1)}).$$

We repeat this iteration until certain stopping criteria is met.

Our framework has two key components: the constrained clustering algorithm $\mathscr{A}$ and the query strategy $\mathscr{Q}$. Next we will discuss their realization in detail, respectively.

C. The Constrained Spectral Clustering Algorithm

The first key component of our framework is a constrained spectral clustering algorithm $\mathscr{A}$. $\mathscr{A}$ takes the graph Laplacian L and a constraint matrix Q as input and outputs a (relaxed) cluster indicator vector $\mathbf{u}$. In general, our framework has no restriction on the realization of $\mathscr{A}$ as long as it satisfies the following property:

Property 1 (Convergence): As the constraint matrix Q approaches Q^*, the output of the constrained clustering algorithm, $\mathbf{u}$, will converge to the groundtruth cluster assignment $\mathbf{u}^*$:

$$\lim_{Q \to Q^*} \mathbf{u} = \mathbf{u}^*.$$

This property is to ensure that our active learning framework will converge to the groundtruth cluster assignment as more constraints are revealed by the oracle. As trivial as this property may seem like, it is not automatically guaranteed by all constrained clustering algorithms. For example, some constrained K-means clustering algorithms are sensitive to the order in which the constraints are enforced and thus may not converge to the groundtruth clustering after all.

There are a number of possible candidates for $\mathscr{A}$ [13], [16], [17] in the literature. In this work we implement the framework using the constrained spectral clustering algorithm we proposed in a recent work [18]. The main advantage of this approach is that it not only satisfies the convergence property but also is flexible enough to incorporate both hard and soft constraints. Its objective function is as follows:

$$\begin{aligned} \arg\min_{\mathbf{u} \in \mathbb{R}^N} \quad & \mathbf{u}^T L \mathbf{u}, \\ \text{s.t.} \quad & \mathbf{u}^T D \mathbf{u} = \text{vol}(\mathcal{G}),\ \mathbf{u}^T Q \mathbf{u} \geq \alpha. \end{aligned} \tag{2}$$

By comparing Eq.(2) to Eq.(1), we can see that a new term $\mathbf{u}^T Q \mathbf{u} \geq \alpha$ is added to the original formulation of spectral clustering. Recall Q is the constraint matrix and $\mathbf{u}$ is the cluster assignment vector, then

$$\mathbf{u}^T Q \mathbf{u} = \sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{u}_i \mathbf{u}_j Q_{ij}$$

can be considered as a measure (in relaxed form) of how well the pairwise relations as implied by cluster assignment $\mathbf{u}$ conform to those as demanded by the constraint matrix Q: the larger the value is, the more consistent they are. Hence the term $\mathbf{u}^T Q \mathbf{u} \geq \alpha$ essentially lower bounds how well the constraints in Q are satisfied (in its relaxed form) by the cluster assignment $\mathbf{u}$.

Eq.(2) is intractable in general. However, we can use the Karush-Kuhn-Tucker Theorem [19] to find a sub-optimal solution (please see the original paper [18] for detailed derivation). Intuitively speaking, recall that the solution to the unconstrained spectral clustering problem (Eq.(1)) is provided by the eigenvalue problem:

$$L\mathbf{u} = \lambda \mathbf{u}.$$

Similarly, the solution to our constrained spectral clustering problem (Eq.(2)) is provided by the following *generalized* eigenvalue problem

$$L\mathbf{u} = \lambda (Q - \alpha I)\mathbf{u}, \tag{3}$$

where I is an $N \times N$ identity matrix and $\alpha < \lambda_{max}$, λ_{max} to be the largest eigenvalue of Q. As we have shown in [18], all generalized eigenvectors of Eq.(3) associated with positive eigenvalues[1] will satisfy the constraint $\mathbf{u}^T Q \mathbf{u} \geq \alpha$; and among those the one that minimizes $\mathbf{u}^T L \mathbf{u}$ would be a sub-optimal solution to Eq.(2). Note that α is a lower-bound on how well the constraints in Q are satisfied: larger α implies the resultant clustering conforms more strictly to the given constraints. When α is sufficiently large, there will only be one generalized eigenvector associated with nonnegative eigenvalue and the eigenvector will be used as the solution to Eq.(2).

D. The Query Strategy

The second key component of our framework is the query strategy $\mathscr{Q}$. Our strategy evaluates the current cluster assignment $\mathbf{u}$ and the constraints in Q and decides what is the best entry of Q^* to query next. The principle it uses is maximum expected error reduction, which means that for all unknown pairwise relations between two nodes, we compute the *expected* error between our current estimation of that value and its groundtruth value, and we pick the pair of nodes with largest expected error and query the oracle for their relation.

Formally, let $P_{ij}^{(t)}$ be our estimation of the pairwise relation between node i and j at time t. A straightforward way to compute $P_{ij}^{(t)}$ from the cluster assignment vector $\mathbf{u}^{(t)}$ is:

$$P_{ij}^{(t)} = \mathbf{u}_i^{(t)} \mathbf{u}_j^{(t)}. \tag{4}$$

Let $\mathsf{d} \in \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ be a distance function that measures the error between our current estimation and the groundtruth value:

$$\mathsf{d}(P_{ij}^{(t)}, Q_{ij}^*) = (P_{ij}^{(t)} - Q_{ij}^*)^2.$$

Since Q_{ij}^* remains unknown until after we actually query it, we cannot compute the error $\mathsf{d}(P_{ij}^{(t)}, Q_{ij}^*)$ directly. Instead, we compute the mathematical expectation of the error over the two possible answers from the oracle:

$$\begin{aligned} \mathsf{E}(\mathsf{d}(P_{ij}^{(t)}, Q_{ij}^*)) = \ & \mathsf{d}(P_{ij}^{(t)}, 1)\mathsf{Pr}(Q_{ij}^* = 1) + \\ & \mathsf{d}(P_{ij}^{(t)}, -1)\mathsf{Pr}(Q_{ij}^* = -1). \end{aligned}$$

Now the question becomes how we can estimate $\mathsf{Pr}(Q_{ij}^* = 1)$ and $\mathsf{Pr}(Q_{ij}^* = -1)$ based on the information we already have. Recall that we assumed $Q^* = \mathbf{u}^* \mathbf{u}^{*T}$, thus Q^* is a rank-one matrix. If we treat the current constraint matrix $Q^{(t)}$ as an approximation to Q^* with missing values, then it is a standard approach to use the rank-one approximation of $Q^{(t)}$ to recover the unknown entries in Q^* [20]. Let $\bar{\mathbf{u}}^{(t)}$ be the largest singular vector of $Q^{(t)}$, then

$$\bar{Q}^t = \bar{\mathbf{u}}^{(t)} \bar{\mathbf{u}}^{(t)T}$$

[1]The generalized eigenvector associated with eigenvalue 0 may or may not satisfy the constraint $\mathbf{u}^T Q \mathbf{u} \geq \alpha$, but it is excluded either way since it is a trivial solution that assigns all the nodes to the same side of the cut.

is the optimal rank-one approximation to $Q^{(t)}$ in terms of Frobenius norm [21]. Then we can compute $\mathsf{Pr}(Q^*_{ij} = 1)$ and $\mathsf{Pr}(Q^*_{ij} = -1)$ as follows:

$$\begin{aligned}\mathsf{Pr}(Q^*_{ij} = 1) &= \mathsf{Pr}(Q^*_{ij} = 1|Q^{(t)}) \\ &= \frac{1 + \min\{1, \max\{-1, \bar{Q}^t_{ij}\}\}}{2}, \\ \mathsf{Pr}(Q^*_{ij} = -1) &= 1 - \mathsf{Pr}(Q^*_{ij} = 1).\end{aligned}$$

Finally, we query the entry that has the maximum expected error:

$$\mathscr{Q}(\mathbf{u}^{(t)}, Q^{(t)}) = \underset{\{(i,j)|Q^{(t)}_{ij}=0\}}{\arg\max} \mathsf{E}(\mathsf{d}(P^{(t)}_{ij}, Q^*_{ij})). \quad (5)$$

IV. Empirical Results

To show the effectiveness of our approach, we evaluated its performance on several UCI benchmark data sets [22]. Our goal is to show that our framework can achieve better performance with a smaller number of actively selected constraints, as compared to a randomly selected constraint set. This effectively tests our active selection approach against the batch approach using the same number of constraints.

We compared our method (`active`) to a baseline method (`random`). Both methods used the exact same implementation of the constrained spectral clustering algorithm. The only difference was that the constraints used by `active` were actively selected using our query strategy, whereas the constraints used by `random` were randomly selected.

We used both hard and soft constraints in our experiments. For hard constraints, we chose five different data sets with groundtruth labels, namely Hepatitis, Iris, Wine, Glass, and Ionosphere. We performed 2-way partition on all data sets. We removed the SETOSA class from the Iris data set, which is the class that is known to be well-separately from the other two. For the same reason we removed Class 1 from the Wine data set. We also removed data instances with missing values. The statistics of the data sets after preprocessing are listed in Table II. For each data set, we computed the affinity matrix A using the RBF kernel (the edge weight between two nodes is the similarity between those two data instances).

For soft constraints, we chose a subset of the 20 Newsgroup data, as shown in Table III. We randomly sampled about 350 documents from 6 groups. At the highest level, those groups can be divided into two topics: computer (comp) and recreation (rec). To generate soft constraints, if two articles are from different topics, we set the corresponding entry in Q^* to -1; if two articles are from the same topic but different groups, we set the corresponding entry to 0.5; if they are from the same group, we set the entry to 1. The affinity matrix A was generated from the similarity matrix based on inner-product (the edge weight between two nodes is the number of words those two articles have in common).

Table II
THE UCI BENCHMARKS

Identifier	#Instances	#Attributes
Hepatitis	80	19
Iris	100	4
Wine	119	13
Glass	214	9
Ionosphere	351	34

Table III
THE NEWSGROUP DATA

Group	Label	#Instances
3	comp.os.ms-windows.misc	53
4	comp.sys.ibm.pc.hardware	60
5	comp.sys.mac.hardware	59
9	rec.motorcycles	65
10	rec.sport.baseball	64
11	rec.sport.hockey	51

On all data sets, we started with no constraint and queried one constraint at a time from the oracle. Note that we did not use the transitive or entailment properties [5] to deduce more constraints based on the existing ones, which is impossible to do when the constraints are soft. To evaluate the accuracy of the resultant clustering at each time step, we used Rand index [23]. For each data set, we made up to $2N$ queries, where N was the size of the data set. There is only one parameter in our method, which is α for the constrained spectral clustering algorithm (see Eq.(3)). Throughout all experiments, we simply set it to $\lambda_{max}/2$, where λ_{max} is the largest eigenvalue of Q. In this way, we guarantee the existence of at least one feasible solution, while requiring that a reasonable amount constraints must be satisfied.

The results are shown in Fig. 1 and we can observe that:

- In most cases, our active method converged to the groundtruth clustering after a small number of queries.
- As a contrast, the baseline method often did not even show performance gain with a randomly selected constraint set of the same size.
- In most cases, the performance of our active method consistently increased as more constraints had been queried. This suggested that the active query process utilized constraints in a helpful way.
- Our active approach outperformed the random approach in average by a large margin. In many cases, our active approach even outperformed the best-of-luck result from the random approach.
- There are also some dataset-specific observations. For example, the performance of our active method on the Iris data set was not as good when only a very small number of constraints had been queried. We contribute this to the existence of contextual outliers, as we discovered in our previous work [24], the influence of which misled the initial active query process. However, our active method recovered quickly after

more constraints were queried. We also noticed that on the 20NG data set, the performance of the random method struggled hopelessly. This is because the data set contains six well-separated sub-clusters (each corresponding to a sub-topic). Randomly queried constraints did not work as effectively as the active method to help identify the two more generalized topics, which lead to the groundtruth 2-partition we are looking for.

The above observations can be explained by noting that our actively set of constraints complement each other whilst the randomly selected constraints may very well be contradictory. These results for the baseline approach are consistent with earlier work [5] which showed the randomly chosen constraint sets often **hurt** performance of the underlying algorithm when measured by the Rand index.

In our experiments, we also noticed that there were cases where our query strategy found more than one constraints with the same largest expected error. In this case, we used a randomized tie-breaking step to pick one of them to query. As a result, although our query strategy is designed to be deterministic, its output on certain data sets could vary over many trials. However, for reasonably large data sets, the variation between different trials appeared to be insignificant.

The data sets and Matlab codes used in our experiments are publicly available. Please contact the authors for information.

V. Implementation Issues

A. Outliers

Our query strategy $\mathscr{Q}$ is an instance-based strategy. As a result, the existence of outliers may cause a large number of additional queries. For example, imagine we have a graph with one outlying node. Without constraints, the spectral clustering algorithm may identify the outlying node as one cluster and the rest of the graph as another (especially when the majority of the graph is a relatively well-connected component). According to our query strategy, the outlying node will have the largest expected error, because the resultant cluster assignment will be entirely different depending on whether or not this node is a true outlier, or it is actually a cluster by itself. Our query strategy will keep querying the pairwise relations between this outlying node and all the other nodes in the rest of the graph until it reaches a conclusion.

On the one hand, we need to point out that this kind of intensive queries on a *key* node is necessary without prior knowledge on the existence of outliers or the underlying distribution of the data. On the hand other, if we do have prior information, e.g. the minimum size of the potential clusters, we could remove obvious outliers during the preprocessing step. This can help avoid initiating our method with a completely wrong clustering, which inevitably would take much more queries to converge to the groundtruth result.

We also found out that normalizing our estimation of the relation between node i and j, which is $P_{ij}^{(t)}$ as shown in Eq.(4), can help reduce the influence of potential outliers in the graph. This can avoid a relatively large entry in $\mathbf{u}^{(t)}$ from dominating the query process. Specifically, we have:

$$P_{ij}^{(t)} = \begin{cases} 1 & \text{if } \mathbf{u}_i^{(t)}\mathbf{u}_j^{(t)} > 1 \\ -1 & \text{if } \mathbf{u}_i^{(t)}\mathbf{u}_j^{(t)} < -1 \\ \mathbf{u}_i^{(t)}\mathbf{u}_j^{(t)} & \text{otherwise} \end{cases}.$$

B. Time Complexity

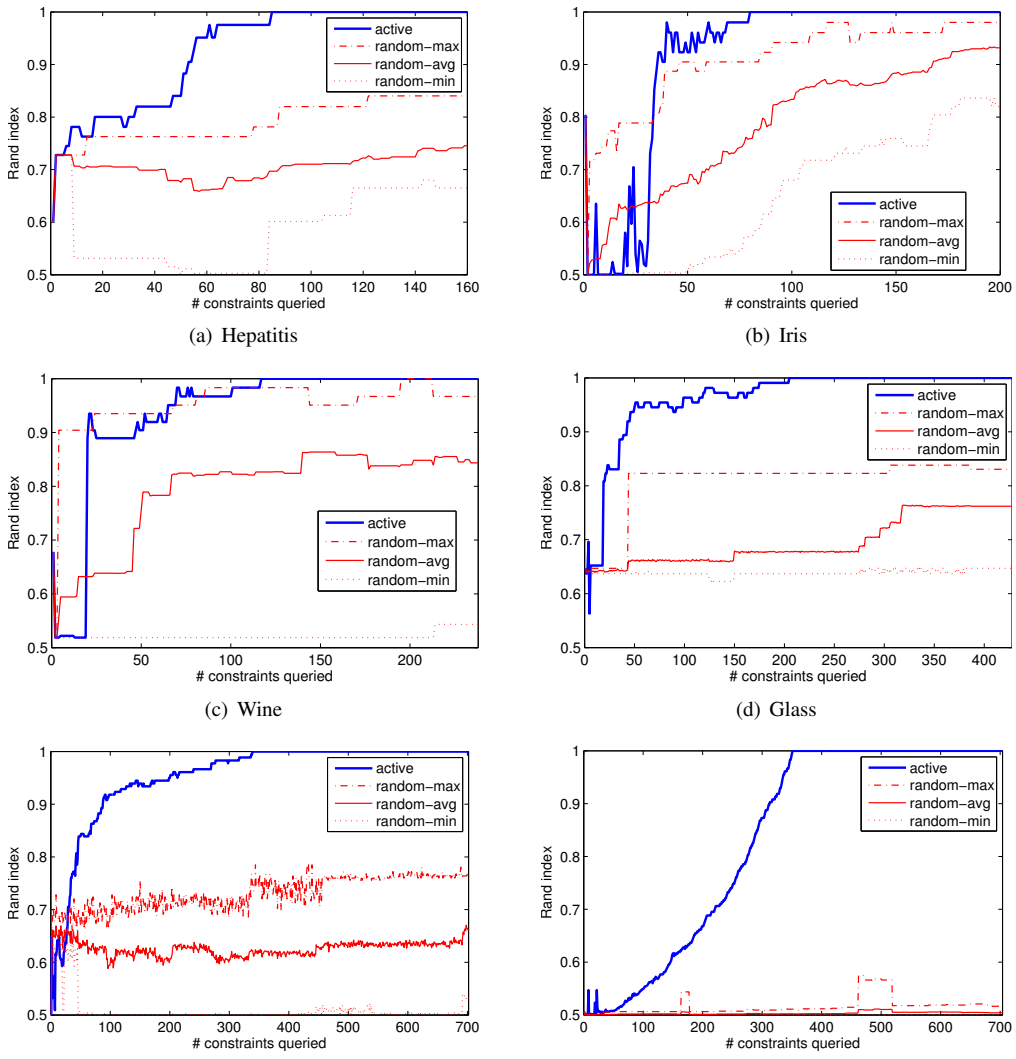
Our active learning method is an iterative process. The time complexity for each iteration is constant. Within the iteration, there are two main steps, the constrained spectral clustering process, and the query process. The runtime of the constrained spectral clustering algorithm we introduce is dominated by that of solving a generalized eigenvalue problem on an $N \times N$ matrix; the runtime of the query process is dominated by computing the rank-one approximation to an $N \times N$ matrix, which takes no longer than solving the eigenvalue problem. Therefore, the overall time complexity of our method is equal to the number of iterations times the time complexity of solving an eigenvalue problem on an $N \times N$ matrix, depending on which solver you choose to use, but $\mathcal{O}(N^2)$ at least. In other words, the runtime of our method is mainly decided by 1) the size of the data set; 2) the number of iterations/queries.

Note that the time complexity of our method does *not* increase with the number of constraints we have queried or the number of constraints we query at each time. Thus in practice we can choose to query more than one constraint during each iteration. However, under the assumption that each query comes with a cost, this is essentially a tradeoff between the runtime and the cost of querying the oracle.

C. Stopping Criterion

A common consideration when implementing active learning algorithms is when to stop querying, because either 1) the result has converged and will no longer change with more constraints, or 2) the result is "good enough" thus further queries are no longer worth the cost. It is possible to find such a criterion when the learning task itself is supervised/semi-supervised and there is some kind of auxiliary information to measure the quality and/or stability of the result. However, it is less likely to find such a measure that works for unsupervised learning (clustering) in general, due to the absolute absence of groundtruth information. Moreover, as Burr Settles stated in his survey on active learning [6]:

> "... in my own experience, the real stopping criterion for practical applications is based on eco-



(a) Hepatitis (b) Iris

(c) Wine (d) Glass

(e) Ionosphere (f) Newsgroup

Figure 1. Results on six UCI data sets, with comparison between our method `active` and the baseline method `random`. Y-axis is Rand index, X-axis is the number of constraints queried. The maximum number of queries is $2N$, where N is the size of the corresponding data set. For the `random` method, the max/average/min performance over 10 runs (each with a randomly generated constraint set) are reported, respectively.

nomic or other external factors, which likely come well before an intrinsic learner-decided threshold."

Therefore, from the practical perspective, we suggest to make as many queries as possible/affordable, and our method will consistently improve the result unless it has already converged to the groundtruth.

VI. Limitations and Future Work

Our query strategy as described in Section III-D assumes that Q^* is, or can be well approximated by, a rank-one matrix. Only with this assumption, we can estimate the expected error using the rank-one approximation technique. However, we notice that there are real-world application scenarios where Q^* may have a higher rank. For example, the Q^* may be derived from a K-way partition of the data set ($K > 2$), then the rank of Q^* would be $K - 1$. Or there might be one-sided oracles who only provide `Must-Link` or Cannot-Link constraints, but not both. To deal with these scenarios, we need to adopt more sophisticated method to estimate the expected error between our current cluster assignment and the groundtruth result.

Another natural extension for our method is K-way partition where $K > 2$. The spectral formulation of clustering is for cutting a graph and many applications of the approach only require a $K = 2$ clustering. Although it is possible to extend the formulation to K-way partition, some important theoretical properties will be lost after the extension, e.g. the result is no longer deterministic and it is difficult to interpret the result as a normalized min-cut. To modify our current algorithm for K-way partition, we need to modify

the constrained spectral clustering algorithm $\mathscr{A}$ to support K-way partition. Common practice is to, instead of only looking at one eigenvector, look at the top-K eigenvectors all together and perform K-means clustering on the rows of the $N \times K$ matrix [25]. Note that now the output of $\mathscr{A}(L, Q)$ would become the $N \times N$ matrix P that directly encodes the pairwise relations between nodes, since a single indicator vector $\mathbf{u}$ cannot encode K-way partition for $K > 2$. Then we need to modify the query strategy to deal with a Q^* whose rank is now $K - 1$, as we mentioned above.

VII. Conclusion

In this work, we proposed an active learning framework for spectral clustering. Its goal is to maximally improve the performance of a given constrained spectral clustering algorithm by using as few constraints as possible. We designed a query strategy that incrementally and iteratively picks the constraint with the largest expected error among all unknown constraints and then retrieves the groundtruth value for that constraint from an oracle. Our framework is not only principled, but also high flexible to work with both hard and soft constraints that may occur in real-world applications. We used several UCI benchmark data sets to validate the advantage of our approach, by comparing to the baseline method with randomly selected constraint set. Empirical results showed that our method can find the groundtruth cluster assignment by only using a small constraint set, and it outperformed the baseline method of specifying the same number of constraints as a batch by a large margin.

Acknowledgment

The authors gratefully acknowledge the support of this research from the NSF (IIS-0801528) and ONR (N000140910712 P00001).

References

[1] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[2] S. Basu, I. Davidson, and K. Wagstaff, Eds., *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Chapman & Hall/CRC, 2008.

[3] D. Klein, S. D. Kamvar, and C. D. Manning, "From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering," in *ICML*, 2002, pp. 307–314.

[4] S. Basu, A. Banerjee, and R. J. Mooney, "Active semi-supervision for pairwise constrained clustering," in *SDM*, 2004.

[5] I. Davidson, K. Wagstaff, and S. Basu, "Measuring constraint-set utility for partitional clustering algorithms," in *PKDD*, 2006, pp. 115–126.

[6] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.

[7] "Flickr: The commons." [Online]. Available: http://www.flickr.com/commons/

[8] "Galaxy zoo." [Online]. Available: http://www.galaxyzoo.org/

[9] P. K. Mallapragada, R. Jin, and A. K. Jain, "Active query selection for semi-supervised clustering," in *ICPR*, 2008, pp. 1–4.

[10] D. Greene and P. Cunningham, "Constraint selection by committee: An ensemble approach to identifying informative constraints for semi-supervised clustering," in *ECML*, 2007, pp. 140–151.

[11] R. Huang, W. Lam, and Z. Zhang, "Active learning of constraints for semi-supervised text clustering," in *SDM*, 2007.

[12] Q. Xu, M. desJardins, and K. Wagstaff, "Active constrained clustering by examining spectral eigenvectors," in *Discovery Science*, 2005, pp. 294–307.

[13] S. D. Kamvar, D. Klein, and C. D. Manning, "Spectral learning," in *IJCAI*, 2003, pp. 561–566.

[14] L. Huang, D. Yan, M. I. Jordan, and N. Taft, "Spectral clustering with perturbed data," in *NIPS*, 2008, pp. 705–712.

[15] G. W. Stewart and J.-g. Sun, *Matrix Perturbation Theory*. Academic Press, Inc., 1990.

[16] Z. Lu and M. Á. Carreira-Perpiñán, "Constrained spectral clustering through affinity propagation," in *CVPR*, 2008.

[17] T. Coleman, J. Saunderson, and A. Wirth, "Spectral clustering with inconsistent advice," in *ICML*, 2008, pp. 152–159.

[18] X. Wang and I. Davidson, "Flexible constrained spectral clustering," in *KDD*, 2010, pp. 563–572.

[19] H. Kuhn and A. Tucker, "Nonlinear programming," *ACM SIGMAP Bulletin*, pp. 6–18, 1982.

[20] M. Brand, "Fast online svd revisions for lightweight recommender systems," in *SDM*, 2003.

[21] R. Horn and C. Johnson, *Matrix analysis*. Cambridge Univ. Press, 1990.

[22] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[23] K. Wagstaff and C. Cardie, "Clustering with instance-level constraints," in *ICML*, 2000, pp. 1103–1110.

[24] X. Wang and I. Davidson, "Discovering contexts and contextual outliers using random walks in graphs," in *ICDM*, 2009, pp. 1034–1039.

[25] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.