Active Constrained Clustering by Examining Spectral Eigenvectors

Qianjun Xu[1], Marie desJardins[1], and Kiri L. Wagstaff[2]

[1] University of Maryland Baltimore County, Dept. of CS&EE
1000 Hilltop Circle, Baltimore MD 21250
[2] Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr., Pasadena CA 91109
{qxu1,mariedj}@cs.umbc.edu, kiri.wagstaff@jpl.nasa.gov

Abstract. This work focuses on the active selection of pairwise constraints for spectral clustering. We develop and analyze a technique for Active Constrained Clustering by Examining Spectral eigenvectorS (ACCESS) derived from a similarity matrix. The ACCESS method uses an analysis based on the theoretical properties of spectral decomposition to identify data items that are likely to be located on the boundaries of clusters, and for which providing constraints can resolve ambiguity in the cluster descriptions. Empirical results on three synthetic and five real data sets show that ACCESS significantly outperforms constrained spectral clustering using randomly selected constraints.

1 Introduction

Recently, clustering research has focused on developing methods to incorporate domain knowledge into clustering algorithms, so that the results are tailored to the interests and existing knowledge of the user. For example, pairwise constraints were introduced by Wagstaff *et al.* [1] as a way to use domain-specific information in the form of *must-link* constraints, which specify that two instances must be in the same cluster, and *cannot-link* constraints, which indicate that two instances must be in different clusters. Although it has been repeatedly demonstrated that constraints can improve clustering performance [1–4], these gains often require the user to specify constraints for a significant fraction of the items in the data set. In this paper, we seek to reduce that user burden by *actively* selecting item pairs for constraint labelling, so that the most informative constraints are acquired as quickly as possible.

Active constraint selection has been previously studied by Basu *et al.* for the K-means algorithm [5]. Their method aims to find k neighborhoods to initialize the clusters. However, for data sets that have close boundaries or small overlap areas on the boundaries, which are the focus of this paper, this method does not work well. We instead propose an active constraint selection method that identifies crucial *boundary points* (those near cluster boundaries) with high probability.

The main contribution of this paper is an active constraint selection technique for data sets with close or overlapping boundaries. We refer to this method as Active Constrained Clustering by Examining Spectral eigenvectorS (ACCESS). ACCESS uses a heuristic derived from the theoretical properties of spectral decomposition methods to identify points at or near cluster boundaries with high probability. Providing the clustering algorithm with constraints on such points can help to resolve ambiguity in the cluster descriptions. Our experiments on three synthetic and five real data sets show that ACCESS yields a significant performance improvement over constrained clustering with randomly selected constraints.

2 Background

Spectral clustering. The eigenvectors derived from the data similarity graph have good properties and can be used for clustering; this class of methods is referred to as *spectral clustering* techniques. Given n data points, we can construct a graph $G = (V, E, A)$, where each vertex v_i corresponds to a point p_i, and the edge $e_{i,j}$ between vertices i and j is weighted by their (dis)similarity value, $a_{i,j}$. Any similarity measure can be used; one popular similarity metric is defined as

$$A_{i,j} = exp(\frac{-\delta_{ij}^2}{2\sigma^2}), \tag{1}$$

where δ_{ij} is the Euclidean distance between point i and j and σ is a free scale parameter. Using this definition, the larger the distance δ_{ij}, the smaller the similarity A_{ij}.

The goal of spectral clustering is to find a *minimal cut* of the graph such that the inter-cluster similarities are minimized. However, this objective favors cutting off a small number of isolated points [6]. Previous research explored refined objectives to overcome this drawback, including the *ratio cut* [6] and *normalized cut* [7] criteria. It can be shown that the second smallest eigenvector of the (generalized) graph Laplacian matrix, defined as $L = D - A$, where D is a diagonal matrix with element $d_{ii} = \sum_{j=1}^{n} A_{ij}$, is an approximation of the cluster membership indicator vector and its corresponding eigenvalue gives the optimal cut value [6, 7]. The second smallest eigenvector is used to split the data into two groups.

Recently, researchers have proposed to make use of k eigenvectors simultaneously for the multi-cluster problem [8, 9]. These methods usually use a normalized similarity graph, such as

$$P = D^{-1}A \tag{2}$$

or

$$N = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}. \tag{3}$$

Note that the eigenvectors derived from the P and N matrices are related to the eigenvectors derived from the (generalized) Laplacian matrix. In particular, if λ and x are the solutions of Equation 2, then $1 - \lambda$ and x are the solutions of the

generalized Laplacian matrix [9]. After obtaining k eigenvectors, any clustering technique, such as the K-means algorithm, can be applied to the eigenspace, that is, the space spanned by the largest k eigenvectors. The justification of clustering in the eigenspace can be found in Ng *et al.* [8] and Meila *et al.* [9].

Each row in the P matrix sums to 1. Therefore, we can interpret the entries P_{ij} as the transition probabilities of a random walker moving from point i to point j. The probabilistic interpretation of the normalized similarity matrix gives an intuitive explanation of the constraints, as we will discuss next.

Incorporating constraints in spectral clustering. Kamvar *et al.* developed a technique to incorporate constraints into spectral clustering [10]. We will refer to their method as KKM after the authors' initials. Their work uses a different normalization matrix, as follows:

$$N = (A + d_{max}I - D)/d_{max}, \tag{4}$$

where d_{max} is the largest element in the matrix D and I is the identity matrix. Note that the off-diagonal entries of N are simply the scaled similarity values: $N_{ij} = A_{ij}/d_{max}$ for $i \neq j$. The diagonal entries, however, are computed by $N_{ii} = (d_{max} - d_{ii})/d_{max}$.

Given a must-link constraint (i, j), KKM modifies the corresponding affinities so that $A_{ij} = A_{ji} = 1$; as a result, when N is re-derived from the new similarity matrix, the transition probability between i and j will be greater than or equal to the transition probabilities leading from i or j to any other point. Similarly, a cannot-link constraint (i, j) is incorporated by setting $A_{ij} = A_{ji} = 0$, preventing a direct transition between points i and j.

Note that the use of the transition probability matrix in Equation 4 may cause problems when there are outliers in the data. For example, if point i is isolated from all other data points, then N_{ii} will be much larger than all other entries N_{ij}. Therefore, once a random walk encounters point i, it has a very low probability of leaving it, resulting in a singleton cluster. To overcome this drawback, our method replaces KKM's transition matrix N with the P matrix in Equation 2. We discuss the advantages of using this matrix in Section 5.

Notation. In this paper, we focus on the two-cluster problem, and assume that there are only two clusters, C_1 and C_2. We index the points so that the points in the first cluster appear before the points in the second cluster. We write the similarity matrix $A = (A_{C_1C_1}, A_{C_1C_2}; A_{C_2C_1}, A_{C_2C_2})$, where $A_{C_1C_1}$ and $A_{C_2C_2}$ are the intra-cluster similarity sub-matrices, and $A_{C_1C_2} = A_{C_2C_1}^T$ are the inter-cluster similarity sub-matrices.

3 Active Constraint Selection

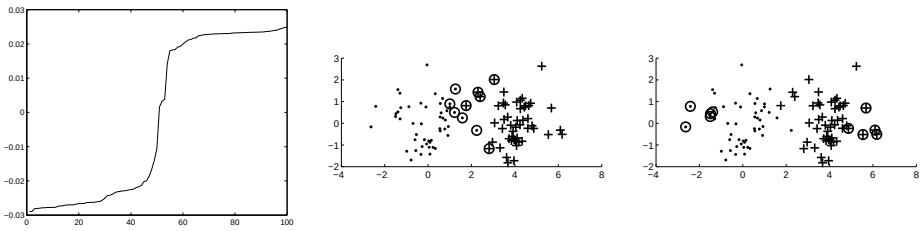
We are interested in clustering problems where the clusters are nearly separated – by which we mean that the boundaries of the clusters are very close, and there may be small overlapping areas. We propose to first analyze the eigenvectors

derived from the data similarity matrix to identify sparse points and boundary points. Then we will query an oracle to give us correct pairwise constraints on these ambiguous points. Incorporating these constraints into spectral clustering improves performance.

Properties of eigenvectors. Our active constraint selection method is based on the following properties of the eigenvectors. Interested readers are referred to the citations for proofs of the theoretical results that we use here.

1. In the ideal case, if the k clusters are well separated, so that only the intra-cluster similarity sub-matrices have nonzero entries, we will obtain k *piecewise constant* eigenvectors—in other words, items from the same cluster will have the same values in the eigenvector, and the clusters can be easily recognized [11, 12].
2. If the clusters are nearly separated (i.e., the dense clusters are loosely connected by a few *bridges* (edges) between them), then the first k eigenvectors will be approximately piecewise constant. This claim has previously been shown by applying matrix perturbation theory to the ideal case [13]. The values in the eigenvectors of points adjacent to these bridges will be pulled towards each other.
3. If the graph is connected, then the identity vector $\mathbf{1}$ is the smallest eigenvector of the Laplacian matrix, and the corresponding eigenvalue is 0. All other eigenvectors are orthogonal to $\mathbf{1}$, which implies that there are both positive and negative (and possibly zero) values in each eigenvector. This can be easily shown by the definition of the Laplacian matrix. This fact motivates a simple heuristic to partition the data: items with positive values in the eigenvector can be put into one cluster, and items with negative values in the eigenvector can be put into the other cluster [11].
4. It has been proved [6] that the second smallest eigenvector of the Laplacian matrix gives the optimal ratio cut cost for splitting the data set into two groups. By inference, the third smallest eigenvector gives the optimal ratio cut cost for further splitting the first two groups. A similar result has been derived for the generalized eigenvectors of the Laplacian matrix for the normalized cut criterion[7]. In summary, the sorted eigenvalues indicate the estimate of cut cost in order, and the different eigenvectors correspond to different splitting strategies.

Close and distant boundary points. For the scenario we are interested in, the items located on the cluster boundaries are the objectives of our active constraint selection, since they are far from the cluster centers and may be interspersed with boundary points of the other clusters. If we can impose constraints to strengthen the similarity between boundary points and members of their clusters, while weakening their similarity to points from other clusters, the clusters themselves will be more clearly apparent in the similarity matrix. We distinguish boundaries between clusters from the outer boundaries of clusters by calling the former *close boundaries*, and the latter *distant boundaries*. Our method aims to find both types.



(a)The 2nd largest eigenvector. (b)Close boundary points. (c)Distant boundary points.

Fig. 1. An illustration of the active constraint selection.

According to statement 2 above, for the data sets we are interested in, the eigenvector will be approximately piecewise constant. In this case, the values in the eigenvector will be bimodally distributed and centered on v and $-w$ (where v and w are positive numbers), with some variances. The close boundary points, *i.e.*, items adjacent to the bridges, are likely to have eigenvector values towards the opposite center. In addition, we hypothesize that the items with eigenvector values far from 0 will be on the distant boundaries (see Figure 1).

Figure 1(a) shows the sorted second largest eigenvector of two Gaussian distributed clusters (represented by + and $\cdot$) and the close and distant boundary points identified from this eigenvector (represented by o). The 10 points with eigenvector values closest to 0 are shown in Figure 1(b); they indeed appear to be located on the close boundaries. In Figure 1(c), the points with largest positive (negative) values have been identified as distant boundary points.

Since we only consider clustering problems with two clusters, we expect that the largest two eigenvectors of the P matrix will be most useful for splitting the data. However, whether or not these eigenvectors are appropriate for this purpose depends on the true data distribution and on the value of the σ parameter, as we show next.

Sparse Points. Figure 2 shows the Ellipses data set. It has two important characteristics: (1) the two clusters have very close boundaries and there is a small group of overlapped items; and (2) there are two small groups of '+' data—the three circled items located at the bottom left corner and one located at the top right corner in Figure 2—that are far away from the main group of '+' data. We call these *sparse points*, and we now examine their effects on the eigenvectors.

The distance from these sparse points to the center of the cluster to which they belong is larger than the distance between the boundaries of two clusters. Therefore, for small values of σ, it is possible that the largest eigenvectors will treat these small groups of '+' data items as a separate cluster. This is exactly what we see using the similarity matrix with $\sigma = 0.2$. Figure 3(a) shows the second largest eigenvector of the Ellipses data set. The anomalous points are exactly those sparse points in Figure 2. Fortunately, the third largest eigenvector (Figure 3(b)) roughly corresponds to the groupings for the remaining data. From

Figure 3(b) we can see that most of the data in the first cluster (indexed from 1 to 80) have positive values, while most data in the second cluster have negative values. Several items violate this structure. These items have either values near zero or large negative values, and therefore can be identified by our method. We can interpret the eigenvectors as follows: the first eigenvector gives cut 1 in Figure 2, the second eigenvector gives cut 2, and the third eigenvector gives cut 3. The third largest eigenvector will be automatically selected by ACCESS to identify the close and distant boundary points.

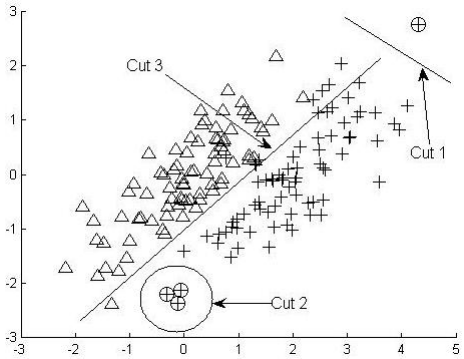
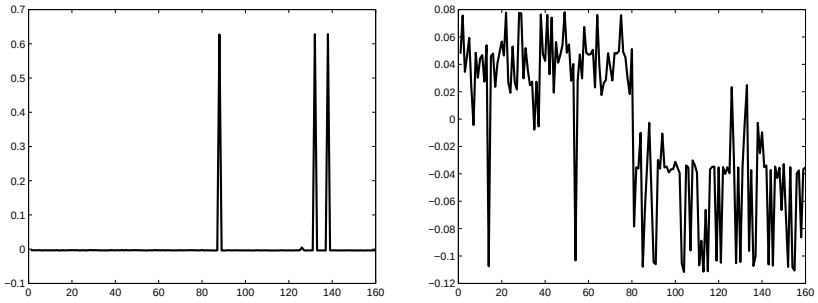


Fig. 2. Ellipses data set.



(a)The 2nd largest eigenvector. (b)The 3rd largest eigenvector.

Fig. 3. The eigenvectors of Ellipses data set.

In summary, for two-cluster problems, our active constraint selection method will identify two types of informative points: (1) the sparse points, identified by the first m eigenvectors (where m depends on how many sparse subclusters are

found in the data set), and (2) the close and distant boundary points identified by the $(m+1)$st eigenvector. These $m+1$ eigenvectors are used to construct the eigenspace matrix in step 7 of the ACCESS algorithm, given below. In the next section, we explain how we identify these points.

Implementation of active constraint selection. Active constraint selection starts with the largest eigenvector. Each eigenvector may produce one of two outcomes. If it identifies one or more sparse points (defined as points whose deviation from the mean value in the eigenvector are greater than three standard deviations), then the next eigenvector will be further examined. Alternatively, if it does not identify any sparse points, then we use this $(m+1)$st eigenvector to identify the close and distant boundary points, and we ignore the remaining eigenvectors.

The close and distant boundary points are identified as follows. Each data point has an associated p_{close}-*value* and $p_{distant}$-*value* when considered as a close or distant boundary point, respectively. The p_{close}-value is inversely proportional to its distance from 0, while the $p_{distant}$-value is proportional to its distance from 0. The detailed computation is as follows (ϵ is a small constant):

1: for the $(m+1)st$ eigenvector e
2: $max_{pos} = \max\{e_i \mid e_i >= 0\}$
3: $max_{neg} = \max\{abs(e_i) \mid e_i < 0\}$
4: **for** e_i **do**
5: **if** $e_i >= 0$ **then**
6: $p^i_{close} = (max_{pos} - e_i + \epsilon)/(max_{pos} + \epsilon)$
7: $p^i_{distant} = (e_i + \epsilon)/(max_{pos} + \epsilon)$
8: **else**
9: $p^i_{close} = (max_{neg} - abs(e_i) + \epsilon/(max_{neg} + \epsilon))$
10: $p^i_{distant} = (abs(e_i) + \epsilon)/(max_{neg} + \epsilon)$
11: **end if**
12: **end for**

Our method chooses sets of boundary points S_{close} and $S_{distant}$ such that following condition is satisfied: $\{p^i_{close} >= p^j_{close}, \forall i, j, i \in S_{close}, j \notin S_{close}\}$ and $\{p^i_{distant} >= p^j_{distant}, \forall i, j, i \in S_{distant}, j \notin S_{distant}\}$. Given q, the number of points to query, ACCESS selects s sparse points, $2(q-s)/3$ close boundary points and $(q-s)/3$ distant boundary points.

Algorithm. The pseudo-code for the ACCESS algorithm is given in Figure 4. There are two parameters: q, the number of items to query, and σ, the scale parameter in Equation 1. Note that our main contribution is in step 3, active constraint selection.

4 Experiments and Results

Data sets. We implemented experiments on three synthetic and five real data sets. The Sphere data set is generated by Gaussian distributions with mean $(0, 0)$

1: Derive matrix A and matrix $P = D^{-1}A$.
2: Compute the eigenvalues and eigenvectors of P.
3: Actively pick q data points by examining the eigenvectors and query the oracle for labels or pairwise constraints.
4: Impose must-link constraint pairs (i, j) by assigning $A_{ij} = A_{ji} = 1$.
5: Impose cannot-link constraint pairs (i, j) by assigning $A_{ij} = A_{ji} = 0$.
6: Reconstruct matrix P'.
7: Identify the largest m' eigenvectors that have sparse points.
8: Pick the largest $m' + 1$ largest eigenvectors of P', and construct the eigenspace matrix $X = (x^1, x^2, \cdots, x^{m'+1})$.
9: Row normalize X to length 1.
10: Perform K-means clustering on the rows of X to identify two clusters.
11: Assign data point i to cluster c if row $X_{i.}$ is assigned to cluster c.

Fig. 4. The ACCESS algorithm.

and $(3, 0)$, and covariance matrix $(1, 0; 0, 1)$. The Ellipses and Test data sets are shown in Figure 2 and Figure 5. The Iris and Soybean data sets are from the UCI Machine Learning Repository [14]. For these data sets, we derive the similarity matrix from the Euclidean distances as in Equation 1. The text data sets are from the 20 Newsgroups collection. We preprocess the data as described by Basu *et al.* [5], then use cosine similarity values. Let $NN_{20}(p)$ be the set of 20 nearest neighbors to point p. We set $A_{i,j}$ of the similarity matrix to zero if $p_i \notin NN_{20}(j)$ and $p_j \notin NN_{20}(i)$. The value 20 was selected based on the method reported by Kamvar *et al.* [10]. Key properties of each data set are shown in Table 1.

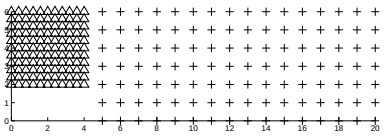


Fig. 5. Tester data set.

Table 1. Real data sets.

data	cluster 1	cluster 2	num
Iris	Versicolour	Virginica	100
Soybean	brown-spot	frog-eye-leaf-spot	183
Text1	alt.atheism	rec.sport.baseball	200
Text2	alt.atheism	sci.space	200
Text3	rec.sport.baseball	sci.space	200

Parameter selection. The σ parameter in Equation 1 significantly affects clustering performance. Ng *et al.* [8] proposed a parameter selection criterion based on the observation that a good σ parameter will yield a partition with small distortion (*i.e.*, small mean squared error). In our implementation, we use a small σ value, since this yields a sparse similarity matrix, which tends to produce good spectral clustering results. In addition, we automatically identify eigenvectors that will isolate small groups of data (Figure 3(b)) and use $m + 1$ eigenvectors for clustering.

Evaluation. The Rand index [15] is often used as an evaluation of the clustering result. The Rand index measures the agreement of two partitions, P_1 and P_2. Given a data set with n points, there are $n(n-1)/2$ pairs of decisions: for each pair of items, each partition either assigns them to the same cluster or to different clusters. Let a and b be the number of pairs for which the two partitions agree by assigning them to the same cluster or to different clusters, respectively. The Rand index (RI) is then defined as:

$$RI(P_1, P_2) = \frac{a+b}{n(n-1)/2}. \tag{5}$$

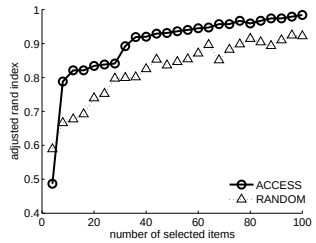
In other words, the RI computes the percentage of agreements among all pairs of decisions.

One problem with the Rand index is that its expected value for two random partitions is not a constant. The adjusted Rand index (ARI) [16] has been proposed to overcome this shortcoming. The expected value for two random partitions with a fixed number of clusters for each partition and a fixed number of instances for each cluster is zero. Let n_{ij} be the number of items that appear in cluster i in P_1 and in cluster j in P_2. ARI is computed as:

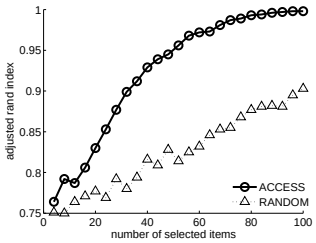
$$ARI(P_1, P_2) = \frac{R - E[R]}{M[R] - E[R]},\ R = \sum_{ij} \binom{n_{ij}}{2} \tag{6}$$

where $E[R] = \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right] / \binom{n}{2}$ is the expected value of R and $M[R] = \frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right]$ is the maximum possible value for R. Note that, the ARI is usually smaller than the RI. We use the ARI for evaluating the expect of our clustering result with the a priori assigned class labels.

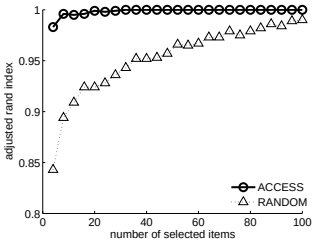
Results and analysis. The baseline for our experiments is constrained clustering with randomly selected constraints. The items are randomly selected, and constraints for each pair of selected items are derived from their true class labels. We compute the transitive closure of the must-link and cannot-link constraints as in Wagstaff *et al.* [1]. Results are averaged over 100 runs. In the results shown in Figures 6(a) to 6(h), the x axis is the number of items selected, and the y axis is the adjusted Rand index. Note that the only difference between the baseline and our method is which items are selected for querying.



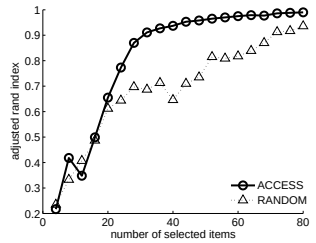
(a) Ellipses ($\sigma = 0.2$).



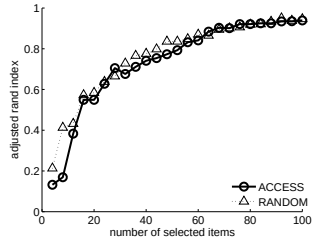
(b) Spheres ($\sigma = 0.5$).



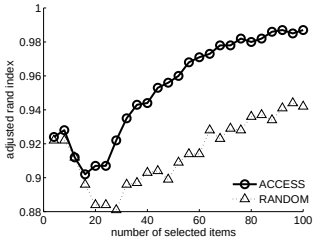
(c) Tester ($\sigma = 0.2$).



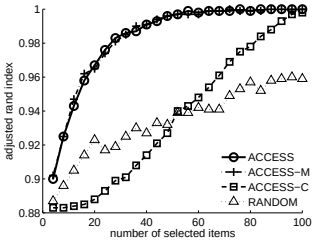
(d) Iris ($\sigma = 0.2$).



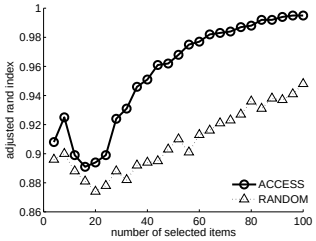
(e) Soybean ($\sigma = 0.5$).



(f) Text1.



(g) Text2.



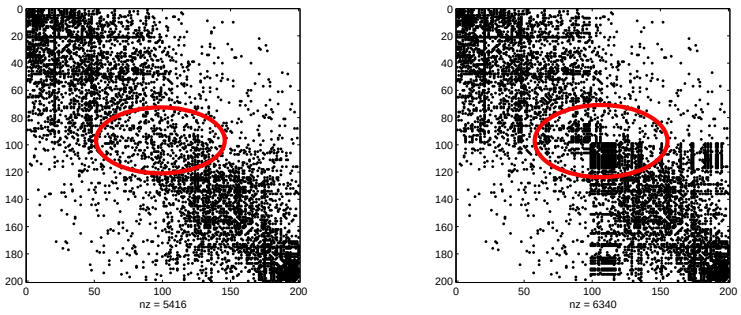
(h) Text3.

Fig. 6. Performance plots.

ACCESS yields better performance, with fewer queries, than randomly selecting constraints, on all data sets except Soybean (for which the performance of ACCESS and random selection is approximately equal). To further understand why our method selects good constraints, we examine the similarity matrix for the Text2 data set, before (Figure 7 (a)) and after (Figure 7 (b)) imposing constraints derived from 50 actively selected items (636 must-link and 589 cannot-link constraints). Rows and columns correspond to the item indices. A dot at position (i, j) means that the similarity value $A_{i,j}$ is positive. We first obtain the second largest eigenvector of the similarity matrix (before imposing any constraints), and then sort the matrix according to the ascending order of this eigenvector. Both plots have the same item ordering. From Figure 7 (a), we can see that the items at the cluster boundaries (i.e., at the intersection of the two diagonal blocks) are mixed together. After imposing the constraints, they are more clearly distinguished (Figure 7 (b)).

We also did comparative experiments when imposing only must-link or only cannot-link constraints. The results show that the must-link constraints improve clustering performance more than the cannot-link constraints. For some of the data sets (Figure 6(g)), imposing only must-link constraints achieves the same performance as imposing both types of constraints. In Figure 6(g), the curve labeled 'ACCESS-M' shows the result of imposing only actively selected must-link constraints, while the 'ACCESS-C' curve illustrates the result of imposing only actively selected cannot-link constraints.

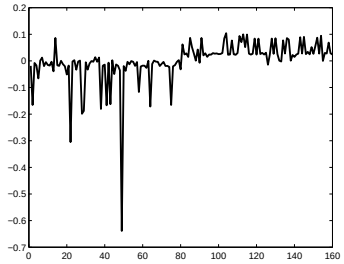
Figure 6(e) shows a case where our method is less effective. Further examining the Soybean data set, there are large overlapping areas between the two clusters. In this case, our method performs comparably to randomly selected constraints.



(a) Before. (b) After.

Fig. 7. The similarity matrix of the Text2 data set before and after imposing constraints.

Fig. 8. The sixth largest eigenvector of the Ellipses data set derived using the N matrix in Equation 4.



5 Discussion and Related Work

There are two primary reasons that we used the P matrix given in Equation 2, rather than the N matrix used by KKM in Equation 4. First, we have previously showed that using randomly selected constraints, the KKM method sometimes performs worse (and was not seen to perform better) than the P-based method [17]. Second, and more importantly, due to the disadvantages discussed in Section 2, the eigenvectors derived from the N matrix fail to identify close and distant boundary points. Because of the complicated distribution of clusters, the N matrix often yields eigenvectors in which several sparse points have extremely large values, while all other points have values near 0. For example, for the N matrix of the Ellipses data set, ACCESS identifies sparse points in the first five eigenvectors. The sixth eigenvector is plotted in Figure 8. However, even for this eigenvector, our method for identifying sparse points returns new points. As a result, it is difficult to use the N matrix to identify the boundary points in the data set.

Recently, there has been some work on active constrained clustering in general. Basu *et al.* implemented an active constraint selection for their Pairwise Constrained K-means algorithm [5]. Their method has two phases. The first phase, Explore, selects an k-neighborhood of must-linked points using the k-centers heuristic. This k-neighborhood is used to initialize the cluster centroids. When queries are allowed, the Consolidate phase is invoked to randomly select a point and query the user about its relation to the known neighborhoods until a must-link is obtained. The authors proved that at least one point can be obtained for each cluster in at most $k\binom{k}{2}$ queries. It implies that 2-neighborhoods can be obtained after querying four items using their method. After that, the authors suggest invoking the Consolidate phase as early as possible, to randomly select items for querying, because the randomly selected samples capture the underlying data distribution and can produce a better estimate of centroids. Their method is tailored to the K-means algorithm, and the purpose of active selection is to get a good estimate of the cluster centroids. When applying their

method to spectral clustering with a large number of selected items (>4 items for 2-cluster problems), the performance should be similar to that of randomly selected items because of the Consolidate phase. An empirical comparison of ACCESS and PCK-means is described in another paper [18].

Klein *et al.* developed a cluster-level active querying technique for hierarchical clustering, which works on data sets that exhibit local proximity structure – locally a point has the same cluster membership as its closest neighbors, while globally, a subcluster has different cluster memberships from its closest neighboring subclusters [2]. These active techniques do not work well in our scenario, where the boundaries are very close. In contrast, our method can identify the points close to the boundaries of clusters.

6 Conclusions and Future Work

In this paper, we described ACCESS, an active constrained spectral clustering method. The actively selected constraints significantly improve clustering performance over randomly selected constraints for data sets that have close boundaries and overlapping regions.

The constraints selected by our method are located on the boundaries of the clusters. It is likely that they could also improve the performance of other clustering methods such as K-means and hierarchical clustering. We are working on applying these constraints to these clustering methods and comparing the performances of different active selection methods.

Our current method focuses on two-cluster problems. We believe that the same idea can be generalized to multiple-cluster problems as well, by identifying the boundary points of one cluster and splitting these points, then recursively splitting the remaining data items.

7 Acknowledgements

We thank Matthew Gaston, Eric Eaton, Blazej Bulka and Priyang Rathod for helpful discussions on developing this technology. We also thank the anonymous reviewers for their comments on improving this paper. This research is supported by NSF grant 0325329.

References

1. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: Proceedings of the 18th International Conference of Machine Learning. (2001) 577–584
2. Klein, D., Kamvar, S.D., Manning, C.D.: From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In: Proceedings of the Nineteenth International Conference on Machine Learning. (2002) 307–314

3. Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning distance functions using equivalence relations. In: Proceedings of the Twentieth International Conference on Machine Learning. (2003) 11–18
4. Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering. In: Proceedings of the Twenty-First International Conference on Machine Learning. (2004) 81–88
5. Basu, S., Banerjee, A., Mooney, R.J.: Active semi-supervision for pairwise constrained clustering. In: Proceedings of the SIAM International Conference on Data Mining. (2004) 333–344
6. Hagen, L., Kahng, A.B.: New spectral methods for ratio cut partitioning and clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **11** (1992) 1074–1085
7. Shi, J., Malik, J.: Normalized cuts and image segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. (1997) 731–737
8. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems 14. (2002) 849–856
9. Meilă, M., Shi, J.: A random walks view of spectral segmentation. In: Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics. (2001)
10. Kamvar, S.D., Klein, D., Manning, C.D.: Spectral learning. In: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence. (2003) 561–566
11. Fiedler, M.: A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. Czechoslovak Mathematical Journal **25** (1975) 619–627
12. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2001) 269–274
13. Ding, C.H., He, X., Zha, H.: A spectral method to separate disconnected and nearly-disconnected web graph components. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2001)
14. Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
15. Rand, W.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association **66** (1971) 846–850
16. Hubert, Arabie: Comparing partitions. Journal of Classification **2** (1985) 193–218
17. Xu, Q., desJardins, M., Wagstaff, K.L.: Constrained spectral clustering under a local proximity structure assumption. In: Proceedings of the 18th International FLAIRS Conference. (2005)
18. Xu, Q., desJardins, M., Wagstaff, K.L.: Active constraint selection for clustering. (Submitted to ICDM 2005)