INFORMS TutORials in Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Machine Learning and Data Mining with Combinatorial Optimization Algorithms

Dorit S. Hochbaum

To cite this entry: Dorit S. Hochbaum. Machine Learning and Data Mining with Combinatorial Optimization Algorithms. *In* INFORMS TutORials in Operations Research. Published online: 19 Oct 2018; 109-129.
https://doi.org/10.1287/educ.2018.0179

https://pubsonline.informs.org/series/educ/

TUTORIALS IN OPERATIONS RESEARCH
 | ISBN 978-0-9906153-2-3
https://doi.org/10.1287/educ.2018.0179

Machine Learning and Data Mining with Combinatorial Optimization Algorithms

Dorit S. Hochbaum
Department of Industrial Engineering and Operations Research, University of California, Berkeley, Berkeley, California 94720
Contact: hochbaum@ieor.berkeley.edu, https://orcid.org/0000-0002-2498-0512 (DSH)

Abstract Binary classification is a fundamental machine learning task defined as correctly assigning new objects to one of two groups based on a set of training objects. Driven by the practical importance of binary classification, numerous machine learning techniques have been developed and refined over the last three decades. Among the most popular techniques are artificial neural networks, decision trees, ensemble methods, logistic regression, and support vector machines. We present here machine learning and pattern recognition algorithms that, unlike the commonly used techniques, are based on combinatorial optimization and make use of information on pairwise relations between the objects of the data set, whether training objects or not. These algorithms solve the respective problems optimally and efficiently, in contrast to the primarily heuristic approaches currently used for intractable problem models in pattern recognition and machine learning. The algorithms described solve efficiently the classification problem as a network flow problem on a graph. The technical tools used in the algorithm are the parametric cut procedure and a process called *sparse computation* that computes only the pairwise similarities that are "relevant." Sparse computation enables the scalability of any algorithm that uses pairwise similarities. We present evidence on the effectiveness of the approaches, measured in terms of accuracy and running time, in pattern recognition, image segmentation, and general data mining.

Keywords supervised machine learning • data mining • binary classification • parametric cut • supervised normalized cut

1. Introduction

In a generic data mining or classification scenario, there are objects with associated feature vectors that are to be classified, or clustered, such that in each group of objects in the same cluster, all objects share some similarity or pattern. The dominant approaches for such machine learning tasks fall most often in the realm of artificial intelligence rules or continuous optimization of intractable problems. We present here combinatorial algorithms for machine learning, data mining, and image segmentation that, unlike the majority of existing machine learning methods, utilize pairwise similarities in addition to feature vectors. We provide empirical evidence that the use of pairwise similarities enhances the quality of data mining and classification and, at the same time, can be done efficiently while scaling well for very-large-scale data sets.

The algorithms described solve efficiently the classification problem as a network flow problem on a graph. The technical tools used in the algorithm are the parametric cut procedure and a process called *sparse computation* that limits the computation of the similarities to only those that are relevant. This is crucial, because the number of pairwise similarities grows at a quadratic rate in the size of the data sets. Sparse computation is the process that enables the

scalability of any algorithm that uses pairwise similarities. It is demonstrated empirically that sparse computation enables the scalability of similarity-based algorithms to very-large-scale data sets while maintaining high levels of accuracy.

The focus here is on a clustering model where a cluster is a group of objects that is as dissimilar as possible from the complement while having as much similarity as possible within the cluster. These two objectives are combined as either a ratio or with linear weights. This problem is a variant of an intractable problem known as *normalized cut*, for which heuristic methods are often used, primarily in image segmentation. The variant problem and the polynomial time algorithm for solving it are called the Hochbaum's normalized cut (HNC).

The combinatorial algorithm(s) described here form a unique paradigm for general data mining and machine learning, in which there are training data in the form of labeled objects. The optimization model and the algorithm under such a setup are referred to as the supervised normalized cut (SNC). SNC is distinctly different from other machine learning algorithms, not only in that it finds an optimal solution to a well-defined optimization problem but also in that the relationships between the unlabeled objects are crucial to the determination of the clusters. The standard approach of machine learning is to generate a mapping of feature vectors to a classification that fits very well the training objects. This mapping is then applied to the unlabeled objects to determine their classification. The mapping is therefore exclusively based on the information provided by the training data yet ignores altogether any relationships between unlabeled objects. Suppose there is a group of very similar unlabeled objects so that different subsets in the group are close to differently classified labeled objects. Most machine learning algorithms would split such a group and assign the various subsets of objects to different labels. By contrast, SNC weighs the similarities between the objects in the group, the intrasimilarity, versus their similarities to training objects, and would assign the entire group with high intrasimilarity the same label. This similarity between unlabeled objects is a useful source of additional information about the data and is shown to contribute to the accuracy performance of machine learning methods. SNC and related combinatorial algorithms for data mining as well as experimental results are described in Section 6.

The collection of clustering models under HNC is more general than the type of model suggested by normalized cut. It allows one to use two different sets of similarity weights, one set to measure intracluster similarity and a second set to measure the intercluster similarity. The first set quantifies the similarity within the cluster, and the second set quantifies the dissimilarity between clusters. It also allows one to include, in addition to the similarity and dissimilarity weights, object weights that can serve as a form of priors or likelihoods of belonging to specific classes.

The HNC criterion has been successfully applied in specific contexts. These include image segmentation (Hochbaum et al. [21]), evaluation of drug effectiveness (Hochbaum et al. [20]), video tracking (Fishbain et al. [13]), enhancement of the capabilities of low-resolution nuclear detectors (Yang et al. [33]), and recently also for identification and tracking neurons in calcium imaging movies (Spaen et al. [29]). In the work on enhancing the capabilities of plastic scintillators (Yang et al. [23]), SNC was compared with several known data mining algorithms. The major conclusions of this study were that SNC and Support Vector Machine (SVM) are by far the most successful among the several machine learning methods in the comparison set, in terms of accuracy, with SNC slightly more accurate than SVM and significantly faster than SVM. The drug evaluation study in Hochbaum et al. [20] used a collection of machine learning techniques containing methods previously used for the task of drug evaluation, with similar conclusions: namely, SNC and SVM proved to be the leaders in terms of accuracy, where SNC was substantially more efficient than SVM. For neuron identification in calcium imaging movies, commonplace in neuroscience, the HNC criterion has been a top performer in the Neurofinder benchmark and has provided superior performance compared with matrix factorization techniques (see Spaen et al. [29]).

1.1. About Image Segmentation and the Use of Similarities

Our use of similarities in clustering is motivated by image segmentation. Image segmentation is fundamental in computer vision (Shapiro and Stockman [26]). It is used in numerous applications, such as in medical imaging (Dhawan [10], Hosseini et al. [22], Pham et al. [24], Roobottom et al. [25]), and it is also of independent interest in clustering (Coleman and Andrews [8], Pappas [23], Shi and Malik [28], Tolliver and Miller [30], Wu and Leahy [31], Xing and Jordan [32]). The image segmentation problem is to delineate, or segment, a salient feature in an image. As such, this is a bipartition problem with the goal of separating the foreground from the background (see Section 5 for illustrations). It is not obvious how to construct a quantitative measure for optimizing the quality of a segmentation. The common belief is that the normalized cut (NC) criterion (Shi and Malik [28]) is a good model for achieving high-quality image segmentation.

The use of pairwise similarities between adjacent pixels is commonplace in image segmentation. An image is formalized as an undirected weighted graph $G = (V, E)$ where each pixel in the image is represented as a node in the graph. A pair of pixels are said to be *neighbors* if they are adjacent to each other. The common neighborhoods used in image segmentation are the *4-neighbor* and *8-neighbor* relations. In the 4-neighbor relation, a pixel is a neighbor of the two vertically adjacent pixels and two horizontally adjacent pixels. The 8-neighbor relation adds also the four diagonally adjacent pixels. Every pair of adjacent neighbors $i, j \in V$ is associated with an edge $[i, j] \in E$. Each edge $[i, j] \in E$ has a weight $w_{ij} \geq 0$ representing the similarity between pixel nodes i and j. It follows that the graphs representing images are very sparse, of maximum node degree equal to 4 or 8, for the *4-neighbor* or *8-neighbor* relations, respectively. This is the main reason why up until now the similarity-based algorithms have been used in image segmentation yet scarcely used in machine learning in general. That is because, for a general data set, without any specific information concerning the adjacency structure, a complete graph has to be constructed. Such complete graphs become prohibitively large even for moderately sized data sets.

It has long been observed that a minimum cut in a graph with edge similarity weights tends to create a bipartition that has one side very small in size, containing a singleton in extreme cases (Wu and Leahy [31]). This is so because the number of edges between a single node and the rest of the graph tends to be much smaller than between two comparably sized sets. This phenomenon is particularly notable in grid graphs such as the 4-neighbor image pixels graph and in complete graphs, where the number of edges between two sets in a balanced partition is an order of magnitude larger than the number of edges adjacent to a singleton. To correct for such unbalanced bipartitions, Shi and Malik [28], in the context of image segmentation, proposed the *normalized cut* as an alternative criterion to the minimum cut, where the value of the cut is "normalized" by the size of the set or by the sum of similarities within the set. However, Normalized Cut is an NP-hard problem and as such cannot be used even for practical-sized images. Several heuristics and approximation algorithms have been employed for solving the normalized cut problem [Dhillon et al. [11, 12], Shi and Malik [28], Tolliver and Miller [30], Xing and Jordan [32]. The most frequently used method is the spectral method that finds the Fiedler eigenvector (Shi and Malik [28]) and that provides a heuristic solution for the normalized cut problem. Section 3 provides an evaluation of how the spectral method compares to combinatorial relaxations of the normalized cut problem. It is shown that the spectral method is a continuous relaxation of normalized cut, whereas a discrete relaxation leads to combinatorial algorithms based on flow problems. Empirical evidence is provided in Hochbaum et al. [21] that demonstrates that the performance of the combinatorial algorithms for image segmentation problems dominates the performance of the spectral method along several dimensions: in terms of better approximation of the respective normalized cut objective, in terms of the visual quality of the segmented images, and in terms of running time.

2. Notations, Problem Definitions, and Preliminaries

Let $G = (V, E)$ be an undirected graph with edge weights w_{ij} associated with each edge $[i, j] \in E$. We note that all the combinatorial algorithms to be described are applicable also for directed, asymmetric, graphs, but in all applications studied here, the similarities are symmetric, and hence the respective graph is undirected. A bipartition of a graph is called a *cut*, $(S, \bar{S}) = \{[i, j] | i \in S, j \in \bar{S}\}$, where $\bar{S} = V \backslash S$. The *capacity of a cut* $(S,\bar{S})$ is the sum of weights of edges with one node in S and the other in $\bar{S}$: $C(S, \bar{S}) = \sum_{i \in S, j \in \bar{S}, [i,j] \in E} w_{ij}$. More generally, for a pair of sets $A, B \subseteq V$, we define $C(A, B) = \sum_{i \in A, j \in B, [i,j] \in E} w_{ij}$. In particular, the *capacity of a set*, $S \subset V$, is the sum of edge weights within the set $S, C(S, S) = \sum_{i,j \in S, [i,j] \in E} w_{ij}$. The *weighted degree* of node i is the sum of weights of edges incident with $i, d_i = \sum_{j \,|\, [i,j] \in E} w_{ij}$.

In the context of classification, the nodes of the graph correspond to data points, each of which may be associated with a feature vector. The edge weights w_{ij} quantify the similarity between the respective feature vectors associated with nodes i and j. Higher similarity is associated with higher weights.

A common form of similarity function is the *Gaussian similarity*. For two objects i and j associated with the feature vectors $v^{(i)}$ and $v^{(j)}$ let $\|v^{(i)} - v^{(j)}\|$ be the Euclidean distance between i and j. The Gaussian similarity between i and j is

$$w_{ij} = \exp\left(-\frac{\left\|v^{(i)} - v^{(j)}\right\|^2}{2\epsilon^2}\right), \tag{1}$$

where parameter ϵ represents a scaling factor. The Gaussian similarity function is often used in image segmentation and spectral clustering, where each object is associated with a scalar value.

In addition to the similarity weights, a node i may have also an arbitrary nonnegative weight associated with it, q_i. For a set of nodes $S \subseteq V$, $q(S) = \sum_{i \in S} q_i$.

For a graph on n nodes with edge (or arc) weights w_{ij}, the matrix $W = (w_{ij})$ and the diagonal matrix D with $D_{ii} = d_i = \sum_{j | [i,j] \in E} w_{ij}$ are $n \times n$ matrices. The matrix $\mathcal{L} = D - W$ is called the *Laplacian* of the graph.

The problem of NC is formulated as

$$\mathrm{NC}(S^*) = \min_{\emptyset \subset S \subset V} \frac{C(S,\bar{S})}{\sum_{i \in S} d_i} + \frac{C(S,\bar{S})}{\sum_{i \in \bar{S}} d_i}. \tag{2}$$

Sharon et al. [27] referred to the following problem as "normalized cut" and stated that it is NP-hard:

$$\min_{\emptyset \subset S \subset V} \frac{C(S,\bar{S})}{C(S, S)}.$$

The solution to this problem is a subset S that is as dissimilar as possible to $\bar{S}$ and that also has similarity within, measured by $C(S, S)$, as large as possible. This problem is related to the normalized cut problem, as discussed in Lemma 1, but it is polynomial time solvable as shown in Hochbaum [16]. This problem is referred to as HNC:

$$\mathrm{HNC}(S^*) = \min_{\emptyset \subset S \subset V} \frac{C(S, \bar{S})}{C(S, S)}. \tag{3}$$

The solution set S^* is referred to here as a *source* set, and its complement is called a *sink* set. The optimization problem HNC, (3), was shown in Hochbaum [16] to be a *monotone integer program*, and as such, it is solvable in polynomial time using a (parametric) minimum cut procedure on an associated graph (Hochbaum [14]). That algorithm can solve more general problems than can HNC. For instance, the similarity weights used in the numerator, the intrasimilarity, can be different from the similarity weights used in the denominator, the intersimilarity. This problem of HNC (3) is equivalent to $\min_{\emptyset \subset S \subset V} \left[(C(S, \bar{S}))/\left(\sum_{i \in S} d_i\right)\right]$, as

proved in Lemma 1. In this equivalent formulation, *any* node weights can be used to replace the weighted degrees of the nodes. For any node weights, the problem is called q-HNC, defined in (6). One variant of node weights, employing *entropy*, was used in Hochbaum et al. [21] for segmentation of images. In that context the entropy variant proved to be highly effective. Furthermore, the solution to HNC was shown in Hochbaum et al. [21] to provide a much better approximation to the objective of NC than the solution provided by the spectral method, with a better visual (subjective) quality. A sketch of some of these empirical results is given in Section 5. Additional discussion on variants of HNC and how they compare with the spectral method used to solve heuristically the normalized cut is provided in Section 3.

To illustrate the algorithmic technique used, we provide a sketch of the algorithm of Hochbaum [16] for solving (3). The algorithms for solving the other variants are given in Hochbaum [18]. The method solves *parametrically* the linearized problem HNC_λ for all parameter values λ,

$$\text{HNC}_\lambda = \min_{\emptyset \subset S \subset V} C\left(S, \bar{S}\right) - \lambda C(S, S). \tag{4}$$

The optimal solution to the ratio problem HNC corresponds to one specific optimal value of the parameter λ^*. It is well known that for a sequence of values $\lambda_1 < \lambda_2 < \cdots < \lambda_k$ and S_i denoting the respective source set associated with λ_i, the source sets are nested: $S_1 \subseteq S_2 \subseteq \cdots \subseteq S_k$. The value of λ^* is selected as the largest for which the objective is nonpositive: $\min_{\emptyset \subset S \subset V} C(S, \bar{S}) - \lambda^* C(S, S) \leq 0$. A biproduct of this solution method is that it finds *all* "breakpoints" of the values of λ and the associated bipartitions, one of which is the desired optimal solution. The number of breakpoints is guaranteed to be relatively "small." Thus the minimum ratio solution corresponds to a specific linear weighting of the two criteria in the objective. It is often the case, however, that the optimal weighting λ^* of the ratio solution is not necessarily the best in terms of the quality of the resulting cluster, and a source set associated with a nonoptimal value of λ is a better cluster. After all, any arbitrary scalar multiplication of the numerator changes the value of the optimal parameter and potentially the respective bipartition solution. It is therefore more effective to consider a "good" weighting of the two criteria instead of solving for the ratio problem. Here, this weighting value of λ is one of the parameters to be selected when implementing HNC, in its linearized version, as a classification method.

2.1. Overview

We first describe the type of clustering algorithms common in image segmentation. These include the use of the spectral method, which is shown in Section 3 to be a (continuous) relaxation of the normalized cut problem. In particular, we show how HNC and variants are a form of discrete relaxation of the normalized cut problem. In Section 4, we describe the combinatorial algorithm that solves a generalization of HNC. The algorithm for HNC is a special case of that method. Next we provide a sample of experimental results for image segmentation in Section 5. Section 6 contains the use of the combinatorial algorithm as a general machine learning technique, with a sample of empirical results. Section 7 sketches the methodology of sparse computation that permits the scaling of any similarity-based algorithms. Several concluding remarks are provided in Section 8.

3. Combinatorial Algorithms in Image Segmentation and the Spectral Method

As noted earlier, the normalized cut model, defined in (2), is well known in applications for image segmentation, and the model of HNC, defined in (3), is somewhat related to it. The normalized cut model is an intractable problem, but it has been solved heuristically by the *spectral method* in which one computes the Fiedler eigenvector of a certain matrix. Because the spectral method is commonly used in image segmentation, we provide here an overview of the method and how HNC relates to it and compares with it.

We call a generalization of normalized cut, introduced in Hochbaum [18], in which the node weights are any nonnegative quantities q_i, the *quantity-normalized cut*, or q-NC:

$$\text{q-NC}(S^*) = \min_{\emptyset \subset S \subset V} \frac{C(S, \bar{S})}{q(S)} + \frac{C(S, \bar{S})}{q(\bar{S})}. \tag{5}$$

A relaxation of this problem, which omits the second term in the objective value, is referred to as q-HNC:

$$\text{q-HNC}(S^*) = \min_{\emptyset \subset S \subset V} \frac{C(S, \bar{S})}{q(S)}. \tag{6}$$

We next show that $\text{HNC}(S^*) = \min_{\emptyset \subset S \subset V} [(C(S, \bar{S}))/(C(S, S))]$ is equivalent to q-HNC where the node weights $q_i = d_i$.

Lemma 1 (Hochbaum [16]). *The sets of optimal solutions to* $\min_{S \subset V} [(C(S, \bar{S}))/(C(S, S))]$ *and* $\min_{S \subset V} [(C(S, \bar{S}))/(d(S))]$ *are identical.*

Proof. Employing arithmetic,

$$\frac{C(S, \bar{S})}{C(S, S)} = \frac{C(S, \bar{S})}{d(S) - C(S, \bar{S})} = \frac{1}{\frac{d(S)}{C(S, \bar{S})} - 1}.$$

Therefore, minimizing this ratio is equivalent to maximizing $(d(S))/(C(S, S))$, which in turn is equivalent to minimizing the reciprocal quantity $(C(S, \bar{S}))/(d(S))$, which is the first term in (2), as claimed. □

Recall that the $n \times n$ matrix $\mathcal{L} = D - W$ is the *Laplacian* of the graph for $W = (w_{ij})$, and D is the diagonal matrix with $D_{ii} = d_i$. For a real vector $\mathbf{y} \in R^n$ and a diagonal matrix Q with $Q_{ii} = q_i$, we introduce the *Rayleigh problem*,

$$\min_{\mathbf{y}^T Q \mathbf{1} = 0, y_i \in \{-b, 1\}} \frac{\mathbf{y}^T \mathcal{L} \mathbf{y}}{\mathbf{y}^T Q \mathbf{y}}.$$

The following lemma, proved in Hochbaum [18], states that for $Q = D$ or general Q, the Rayleigh problem is equivalent to the normalized cut or the quantity-normalized cut, respectively. A special case of this lemma for $Q = D$ was proved in Shi and Malik [28].

Lemma 2 (Hochbaum [18]). *For a diagonal matrix* Q *with* $Q_{ii} = q_i \geq 0$,

$$\min_{\mathbf{y}^T Q \mathbf{1} = 0, y_i \in \{-b, 1\}} \frac{\mathbf{y}^T \mathcal{L} \mathbf{y}}{\mathbf{y}^T Q \mathbf{y}} = \min_{\emptyset \neq S \subset V} \frac{C(S, S)}{q(S)} + \frac{C(\bar{S}, \bar{S})}{q(\bar{S})}.$$

Proof. Let the variables y_i be binary with values $-b$ or 1, defined as follows:

$$y_i = \begin{cases} 1 & \text{if } i \in S, \\ -b & \text{if } i \in \bar{S}. \end{cases}$$

We first note that $\mathbf{y}^T Q \mathbf{y} = q(S) + b^2 q(\bar{S})$. Second, the orthogonality constraint, $\mathbf{y}^T Q \mathbf{1} = 0$, is equivalent to $b = (q(S))/(q(\bar{S}))$. The latter is a form of a *balance* requirement on the two parts of the bipartition, which is why we refer to this constraint as the "balance constraint":

$$\begin{aligned} \mathbf{y}^T \mathcal{L} \mathbf{y} &= \mathbf{y}^T D \mathbf{y} - \mathbf{y}^T W \mathbf{y} \\ &= \textstyle\sum_{i \in S} d_i + b^2 \sum_{i \in \bar{S}} d_i - [C(S, S) - 2bC(S, \bar{S}) + b^2 C(\bar{S}, \bar{S})] \\ &= C(S, S) + C(S, \bar{S}) + b^2 C(S, \bar{S}) + b^2 C(\bar{S}, \bar{S}) \\ &\quad - [C(S, S) - 2bC(S, \bar{S}) + b^2 C(\bar{S}, \bar{S})] \\ &= (1 + b^2 + 2b) C(S, \bar{S}) = (1 + b)^2 C(S, \bar{S}). \end{aligned}$$

Therefore,

$$\min_{\mathbf{y}^T Q\mathbf{1}=0, y_i\in\{-b,1\}} \frac{\mathbf{y}^T \mathscr{L}\mathbf{y}}{\mathbf{y}^T Q\mathbf{y}} = \min_{\mathbf{y}^T Q\mathbf{1}=0, S\subset V} \frac{(1+b)^2 C(S,\bar{S})}{q(S)+b^2 q(\bar{S})}. \tag{7}$$

We now substitute $\mathbf{y}^T Q\mathbf{1}=0$ by the equivalent expression $b=(q(S))/(q(\bar{S}))$:

$$\frac{(1+b)^2 C(S,\bar{S})}{q(S)+b^2 q(\bar{S})} = \frac{\left(1+\frac{q(S)}{q(\bar{S})}\right)^2 C(S,\bar{S})}{q(S)+\left(\frac{q(S)}{q(\bar{S})}\right)^2 q(\bar{S})} = \frac{\left(1+\frac{q(S)}{q(\bar{S})}\right)^2 C(S,\bar{S})}{q(S)\left(1+\frac{q(S)}{q(\bar{S})}\right)} = \frac{\left(1+\frac{q(S)}{q(\bar{S})}\right) C(S,\bar{S})}{q(S)}.$$

Hence,

$$\min_{\mathbf{y}^T Q\mathbf{1}=0, y_i\in\{-b,1\}} \frac{\mathbf{y}^T \mathscr{L}\mathbf{y}}{\mathbf{y}^T Q\mathbf{y}} = \min_{S\subset V} C(S,\bar{S})\left[\frac{1}{q(S)}+\frac{1}{q(\bar{S})}\right],$$

as claimed. □

3.1. The Continuous Relaxation and the Spectral Method

Because the quantity-normalized cut is NP-hard even for $q_i=1$, it follows that the Rayleigh problem is intractable even for $Q=I$. One possible heuristic approach for solving the problem is to consider the *continuous relaxation* attained by eliminating the requirements on the variables that $y_i\in\{-b,1\}$ and permitting each y_i to assume any real value. The *spectral relaxation* can be used to minimize this relaxation, referred to as the Rayleigh ratio $\mathscr{R}r(Q)$:

$$\mathscr{R}r(Q) = \min_{\mathbf{y}^T Q\mathbf{1}=0} \frac{\mathbf{y}^T \mathscr{L}\mathbf{y}}{\mathbf{y}^T Q\mathbf{y}}.$$

The discrete Rayleigh ratio problem b-DRR for a scalar $b\geq 0$ is the relaxation of the Rayleigh ratio in which the sum constraint is removed:

$$R(Q,b) = \min_{y_i\in\{-b,1\}} \frac{\mathbf{y}^T \mathscr{L}\mathbf{y}}{\mathbf{y}^T Q\mathbf{y}}.$$

For the Raleigh problem and the discrete Raleigh ratio, any solution $\mathbf{y}$ implies a partition $(S,\bar{S})$:

$$y_i = \begin{cases} 1 & \text{if } i\in S, \\ -b & \text{if } i\in\bar{S}. \end{cases}$$

The scalar b, which seems out of place, is in fact the "balance" between the size of S and $\bar{S}$ as implied by the *orthogonality constraint* $\mathbf{y}^T Q\mathbf{1}=0$. For $Q=I$, this constraint $|S|=b|\bar{S}|$, and hence for $b=1$, the partition is into two sets of equal size. For general diagonal matrix Q, the partition satisfies a "weighted balance" $\sum_{i\in S} q_i = b\sum_{j\in\bar{S}} q_j$.

For Q nonnegative, the optimal solution to the continuous relaxation $\mathscr{R}r(Q)$ is attained by setting the vector $\mathbf{y}$ equal to the second-smallest eigenvector solving $\mathscr{L}\mathbf{y}=\lambda Q\mathbf{y}$ for the smallest nonzero eigenvalue λ. To see that, note that the trivial minimum of $\mathscr{R}r(Q)$ is 0, which corresponds to the smallest eigenvalue 0 and eigenvector $\mathbf{y}=\mathbf{1}$. To rule out this trivial solution that does not correspond to a partition, one solves $Q^{-\frac{1}{2}}\mathscr{L}Q^{-\frac{1}{2}}\mathbf{z}=\lambda\mathbf{z}$ for the eigenvector $\mathbf{z}$ that

corresponds to the second-smallest eigenvalue, setting $\mathbf{y} = Q^{-\frac{1}{2}}\mathbf{z}$. These second-smallest eigenvector and eigenvalue are called the Fiedler eigenvector and Fiedler eigenvalue, respectively. The orthogonality constraint rules out the trivial solution by requiring that the solution vector $\mathbf{y}$ is orthogonal to $\mathbf{1}$. To map the Fiedler eigenvector solution to a partition, one sets all the positive entries of y_i to 1, and hence i is in S, and all the remaining ones are assigned to $\bar{S}$. Alternatively, another threshold value is chosen, and all values of y_i that exceed the threshold value are set to S and the others to $\bar{S}$. In an implementation known as the *spectral sweep method*, one guesses all possible $n-1$ thresholds and selects the one that has the smallest value of the normalized cut objective. The spectral method was proposed as an effective technique for image segmentation, which is to separate a salient feature from the background. In Section 5, we report on experimentation comparing the HNC algorithm, which solves the discrete Raleigh ratio problem, with the spectral method of Shi and Malik [28], for which an implementation is available in Couret al. [9]. We also report, in Hochbaum et al. [21], on a set of experiments in which HNC is compared with the spectral sweep method.

3.2. The Discrete Relaxation and HNC

The discrete Rayleigh ratio problem for a scalar $b \geq 0$ is attained by relaxing the orthogonality sum constraint:

$$R(Q,b) = \min_{y_i \in \{-b,1\}} \frac{\mathbf{y}^T \mathscr{L}\mathbf{y}}{\mathbf{y}^T Q\mathbf{y}}.$$

From the proof of Lemma 2 and Equation (7), $R(Q,b)$ is equal to minimizing a certain combinatorial expression:

$$\min_{y_i \in \{-b,1\}} \frac{\mathbf{y}^T \mathscr{L}\mathbf{y}}{\mathbf{y}^T Q\mathbf{y}} = \min_{S \subset V} \frac{(1+b)^2 C(S,\bar{S})}{q(S) + b^2 q(\bar{S})}.$$

When $q_i = d_i$ for all nodes (and $Q = D$) and $b = 0$, the combinatorial expression is $(C(S,\bar{S}))/(d(S))$, which was proved in Lemma 1 to be equivalent to HNC. Although HNC appears to be a special case, we showed in Hochbaum [18] that this discrete relaxation is solved for any matrix Q, and for all values of b simultaneously, in the complexity of a single minimum cut on the graph. We showed in Hochbaum [18] that the solution is represented as a collection of no more than n breakpoints, which are the same for any value of b. Note that the solution for the optimal ratio differs for different values of b, but it is always one of the breakpoint solutions. The description of the algorithm is provided next. Note that for $b = 0$ we get the standard HNC ratio problem.

4. The Algorithm Solving the Discrete Rayleigh Ratio Problem

The DRR algorithm solves a relaxation of the Rayleigh problem resulting from omitting constraint $\mathbf{y}^T D\mathbf{1} = 0$ and specifying the value of b, $R(Q,b)$. This problem is the *discrete Rayleigh relaxation*, or b-DRR:

$$(b\text{-DRR}) \quad R(Q,b) = \min_{y_i \in \{-b,1\}} \frac{\mathbf{y}^T \mathscr{L}\mathbf{y}}{\mathbf{y}^T Q\mathbf{y}} = \min_{\emptyset \subset S \subset V} \frac{(1+b)^2 C(S,\bar{S})}{q(S) + b^2 q(\bar{S})}. \tag{8}$$

To solve (8) we first "linearize" the ratio function: a general approach for minimizing a fractional (or as it is sometimes called, geometric) objective function over a feasible region $\mathscr{F}$, $\min_{x \in \mathscr{F}} [(f(x))/(q(x))]$, is to reduce it to a sequence of calls to an oracle that provides the yes/no answer to the λ- *question*:

Is there a feasible subset $x \in \mathscr{F}$ such that $(f(x) - \lambda g(x) < 0)$?

If the answer to the λ-question is *yes*, then the optimal solution has a value smaller than λ. Otherwise, the optimal value is greater than or equal to λ. A standard approach is then to utilize a binary search procedure that calls for the λ-question $O(\log(UF))$ times in order to solve the problem, where U is an upper bound on the value of the numerator and F an upper bound on the value of the denominator.

Therefore, if the linearized version of the problem—that is, the λ-question—is solved in polynomial time, then so is the ratio problem. Note that the number of calls to the linear optimization is not strongly polynomial but rather, if binary search is employed, depends on the logarithm of the magnitude of the numbers in the input. In our case, however, there is a more efficient procedure.

The λ-question of whether the value of b-DRR is less than λ is equivalent to determining whether

$$\min_{y_i \in \{-b,1\}} \mathbf{y}^T \mathcal{L}\mathbf{y} - \lambda \mathbf{y}^T D\mathbf{y} = \min_{S \subset V} (1+b)^2 C\big(S, \bar{S}\big) - \lambda\big[q(S) + b^2 q\big(\bar{S}\big)\big] \tag{9}$$

is negative. We next construct an s, t graph, G^b_{st}, which corresponds to $G = (V, E)$ for a fixed value of b. This construction is a special case of solving monotone integer programming proved in Hochbaum [14]. In Theorem 1 we prove that the source set S of the minimum s, t-cut in G^b_{st} is an optimal solution to (9). Furthermore, the solution, for all values of λ and thus for the optimal ratio, is generated in the complexity of a single minimum s, t-cut. The construction of the graph is illustrated when G is a grid graph where each interior node is adjacent to four neighbors.

The graph G^b_{st} is constructed as follows: We add a source node s and a sink node t. For each edge $[i, j] \in E$, there is a pair of arcs, $(i, j), (j, i) \in A_{st}$, both with capacity $(1+b)^2 w_{ij}$. For each node i, there is an arc (s, i) of capacity λq_i and an arc (i, t) of capacity $\lambda b^2 q_i$. Two nodes, $s', t' \in V$, are designated as the seed source and seed sink, respectively. This is done by assigning infinite capacity to (s, s') and to (t', t). This assignment guarantees that in any feasible solution, s' will be part of the source set S and t' will be part of the sink set $\bar{S}$. This will rule out the trivial solutions of $S = \emptyset$ or $S = V$.

Theorem 1. *The source set of a minimum cut in the graph G^b_{st} is an optimal solution to the linearized (b-DRR) (9).*

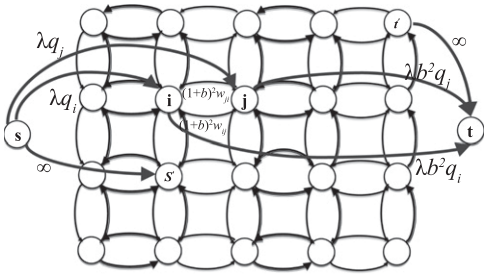
Proof. Let $(S \cup \{s\}, T \cup \{t\})$ be a partition of $V \cup \{s, t\}$ corresponding to a finite capacity s, t-cut in G^b_{st}. We compute this cut's capacity:

$$\begin{aligned} C(S \cup \{s\}, T \cup \{t\}) &= \lambda q(T) + \lambda b^2 q(S) + C(S, T) \\ &= \lambda\big(1 + b^2\big) q(V) - \lambda q(S) - \lambda b^2 q(T) + C(S, T). \end{aligned}$$

Now the first term, $\lambda(1+b^2)q(V)$, is a constant. Thus minimizing $C(S \cup \{s\}, T \cup \{t\})$ is equivalent to minimizing $(1+b)^2 C(S, \bar{S}) - \lambda[q(S) + b^2 q(\bar{S})]$, which is the objective of (9). □

Next we show that graph G^b_{st} can be simplified in the sense that it is equivalent to another graph where only the source-adjacent arcs' capacities and the sink-adjacent arcs' capacities depend on the parameters λ and b. In simplifying the graph, we distinguish between the cases where $b > 1$ or $b < 1$ or $b = 1$. When $b > 1$, all source-adjacent finite capacity arcs are saturated, as the flow of λq_i saturates the arc (s, i) and is below the capacity of (i, t), $\lambda b^2 q_i$. We can thus subtract the lower capacity from both, resulting in an equivalent graph with sink-adjacent capacities equal to $\lambda(b^2 - 1) q_i$ and source-adjacent capacities equal to zero (except for the seed s'). Similarly, for $b < 1$, we subtract $\lambda b^2 q_i$ from the capacity of (s, i) and (i, t). This results in no node adjacent to the sink. We therefore choose a sink "seed" node u, and connect it to t with an infinite capacity arc. The capacity of the arcs from s to each node i is then $\lambda(1 - b^2) q_i$,

Figure 1. The graph G^b_{st} for G a grid graph generated from an image segmentation problem with 4-neighborhood setup and node weights q_i.



as illustrated in Figure 2. In the case where $b = 1$, the analogous update results in all source-adjacent and sink-adjacent arcs having zero capacity.

We next *scale* the graph arc weights by multiplying all arc capacities by a nonnegative scalar, α. For $\alpha > 0$, a minimum cut of capacity $\mathcal{C}$ in a graph G^b_{st} is also a minimum cut of capacity $\alpha\mathcal{C}$ in the scaled graph $\alpha \cdot G^b_{st}$. Here, we choose $\alpha = 1/(1+b)^2$, as illustrated in Figure 3.

In the updated graph G^b_{st}, either all source-adjacent capacities are proportional to λ or all sink-adjacent capacities are proportional to λ. This is therefore an instance of a parametric minimum cut where all the arcs adjacent to the source are monotone nondecreasing with the parameter λ and all arcs adjacent to the sink are monotone nonincreasing with the parameter λ. A parametric minimum cut procedure can then be applied to find the largest value of λ so that the λ question is answered in the affirmative. The running time of the procedure is the running time of solving for a single minimum cut, using the push-relabel algorithm as in Fishbain et al. [13], or using the HPF algorithm, as in Hochbaum [15]. Within this run time, all breakpoints of the parameter λ are generated, where the cut is changing by at least a single node. It is well known that there are no more than n breakpoints (Hochbaum [15]).

Because HNC is a special case with $Q = D$ and $b = 0$, this parametric cut procedure implies a polynomial time algorithm for the problem. The running time of the procedure is $T(n, m)$ for a graph or digraph on n nodes and m edges or arcs. This run time improves on the running time of the polynomial time algorithm in Hochbaum [16], which is based on a non-Rayleigh formulation and has a running time of $T(m, m)$. For $Q = D$ and $b = 1$, the problem is the simple minimum cut problem (we leave this to the reader to verify).

4.1. Solving for All Values of b Efficiently

For each value of b, the optimal solution to the ratio problem may be different. Yet any optimal solution for b corresponds to a bipartition from the same universal set of "breakpoint" solutions common to all values of b.

Figure 2. (Color online) Updated source and sink adjacent arcs' capacities.

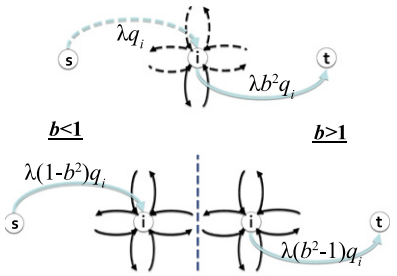
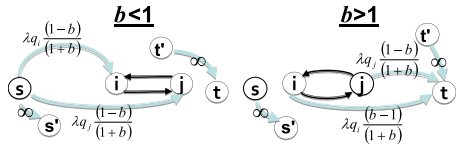


Figure 3. (Color online) Scaling arc weights in G^b_{st}.



To implement the parametric procedure efficiently, we choose a parameter $\beta = \lambda \cdot [(1-b)/(1+b)]$ for $b<1$ and a parameter $\beta = \lambda \cdot [(b-1)/(1+b)]$ for $b>1$. There are no more than n breakpoints for β. There are $\ell \leq n$ *nested* source sets of minimum cuts, each corresponding to one of the ℓ breakpoints. Given the values of β at the breakpoints, $\{\beta_1, \ldots, \beta_\ell\}$, we can generate, *for each value of* b, all the breakpoints of λ^b. For $b<1$, $\lambda^b_i = \beta_i \cdot [(1+b)/(1-b)]$. Consequently, by solving once the parametric problem for the parameter β, we obtain simultaneously all the breakpoint solutions for *every value of* b in the complexity of a single minimum cut, $T(n, m)$.

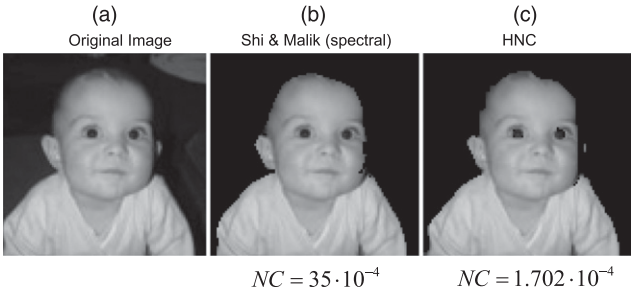
The one-breakpoint solution that minimizes the ratio for a given value of b depends on the value of b. But any scaling of the numerator or denominator of the ratio may give a different optimal solution, which coincides with a different breakpoint. We contend that a ratio is a rather arbitrary form of weighting the two different objectives in the numerator and denominator, and the scalar multiplication of the numerator can change that relative weighting. We use a "good" selection of a breakpoint in the experimental study reported in Hochbaum [18], which is the one that gives the best value of the respective objective function (normalized cut, or q-normalized cut).

5. Experimental Results for Image Segmentation

We tested the quality of the segmentation provided by HNC compared with the results delivered by the spectral method using Shi's implementation [9]. Because the spectral method is justified by its approximating of the optimal solution to the NC problem, we compare the solutions by their NC objective value. The similarity weights are generated, as is the standard in image segmentation, with Gaussian exponential weights. The following two experiments, presented in Figure 4 and Figure 5, use an image provided by Shi and an image provided by Fishbain for this purpose, respectively. The segmentation achieved using the spectral method and HNC are presented in the images by having the foreground pixels (the source set) as in the original image and coloring black all the background pixels (the sink set). The combinatorial algorithm for HNC is implemented using the HPF algorithm for max-flow min-cut (Hochbaum [17]). The results in Figures 4 and 5 were first presented in Hochbaum [16]. Each of the segmentations is a bipartition that is associated with a respective NC objective value. For the image in Figure 4, the discrete approach improves on the spectral approach by a factor of about 20, and for the image in Figure 5, the improvement is by a factor of about 80. In terms of the subjective visual quality of the segmentations, the quality of both the spectral segmentation and the HNC segmentation appear similar for the image in Figure 4. Yet, for the image in Figure 5, the background identified by the spectral method is obviously "wrong": it contains only half of the sky pixels.

In another, more extensive set of experiments, Hochbaum et al. [21] used 20 images from a benchmark of images to test the performance of HNC versus that of the spectral method. Here, there were two sets of experiments: In one set, the similarity weights were the Gaussian exponential weights, (1), and the node weights were d_i, the weighted degree of node i. In a second set, the node weights q_i were "entropy" weights determined by the neighborhood of each node (pixel). We then applied the spectral method to this q-normalized cut problem for the matrix Q still using Shi's code. The results are summarized in Figure 6, where for each one of the 20 images there are two bars, one indicating the ratio of the NC objective value for the

Figure 4. Normalized cut segmentation: (a) the input image (*Source*: Jianbo Shi), (b) the spectral method output segmentation using Shi's implementation, and (c) the output segmentation using the HNC algorithm.



spectral solution to the HNC solution and the other indicating the ratio of the respective value of the q-normalized cut objective. As can be seen, the HNC method performed significantly better for all instances. Note that the ratio bars are given on a logarithmic scale.

6. Combinatorial Algorithms for Data Mining

6.1. Supervised Normalized Cut

Here, we are referring to the linearized version of HNC as HNC. The HNC model can be used as either an *unsupervised* or *supervised* technique. To guarantee that the solution is nonempty and strictly contained in V, one assigns, in advance, at least one node to be included in the source set S and at least one node to be included in the sink set $\bar{S}$ (see Hochbaum [16] for details). These nodes are referred to as *seed nodes*. We exploit the seed node mechanism to use HNC as a *supervised* technique, forcing a priori the training data to be in either the source S or the sink $\bar{S}$, based on the training data labels. Specifically, for the *supervised HNC* method, which we call SNC, the input consists of three sets: two sets of nodes, A and B, which are associated with feature vectors acquired from two different labels' classes in the training set, M^1 and M^2, and a third set, I, corresponding to data points with unknown labels. The goal of the binary classification problem is to associate each data point in I with either M^1 or M^2.

The input to the classification problem is the graph, $G = (V, E)$, defined on the set of objects $V = A \cup B \cup I$ and the similarity weights associated with each pair of nodes (edge) $[i, j] \in E$. Two nodes s and t are added to the graph with an arc of infinite weight from s to each node in A and from each node in B to t. On this graph we seek a partition that minimizes the HNC criterion so that $s \in S$ and $t \in \bar{S}$. The nodes in I that end up in S are classified as A; that is,

Figure 5. Normalized cut segmentation: (a) the input image (*Source*: Barak Fishbain), (b) the spectral method output segmentation using Shi's implementation, and (c) the output segmentation using the HNC algorithm.

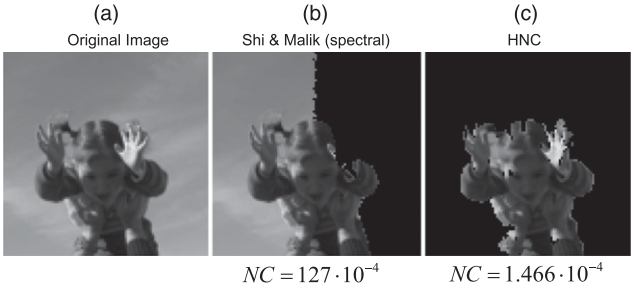
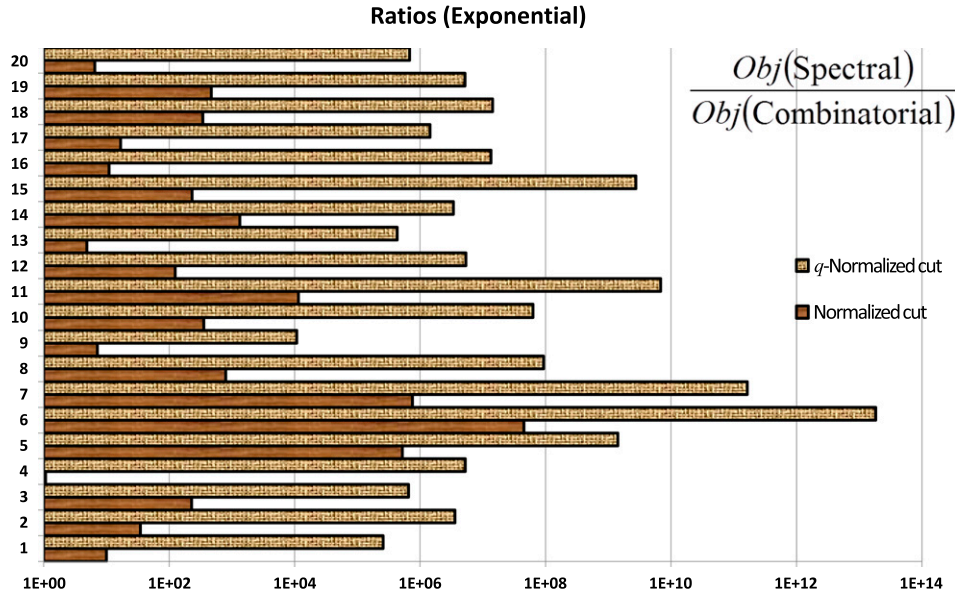


Figure 6. (Color online) The relative performance of the spectral method compared with the combinatorial method for the minimization problems of the q-normalized cut and normalized cut; the larger the ratio, the larger the gap and the better the performance of the combinatorial algorithm.

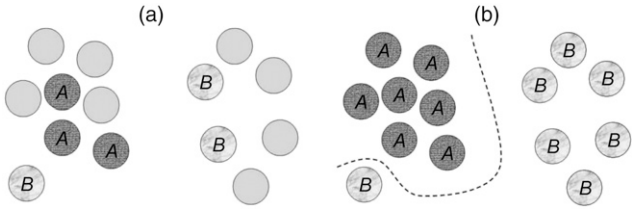


training data with associated class M^1 and nodes in I that end up in $\bar{S}$ are classified as B, thus acquired from M^2. This is illustrated in Figure 7 (adapted from Yang et al. [33]).

The adjustment of HNC to a supervised context, as described, is the *supervised* classification methodology. Because the adjustment of setting the seeds does not affect the running time, the computational complexity of SNC is the same as that of HNC.

The setup of HNC does not require any labeled nodes except one node s that belongs to the cluster S and one node t that belongs to the complement. The selection of s and t is to guarantee that the solution is nonempty and strictly contained in V. When only two nodes are specified as the *source* node and *sink* node, we refer to this variant of HNC as the *unsupervised* variant of HNC. However, HNC can also be implemented as a *supervised* classification method. In the supervised case, the input graph contains labeled nodes (training data) that refer to objects for which the class label (either positive or negative) is known and unlabeled nodes that refer to objects for which the class label is unknown. By assigning all labeled nodes with a positive label to set S as source nodes merged with s and all labeled nodes with a negative label to set $\bar{S}$ as sink nodes merged with t, the HNC model can be used in a supervised manner (see Figure 7). The goal is then to assign all unlabeled nodes either to set S or $\bar{S}$. Because of the preassignment of labeled nodes, the graph's size is reduced because all labeled

Figure 7. (a) Training sets A (dark grey) and B (light grey) and unclassified nodes C. (b) The solution consisting of two sets separated by a cut, with the set on the left containing nodes classified as A nodes and the set on the right containing nodes classified as B nodes.



nodes are merged with s or t, so the "supervised" graph contains only unlabeled nodes. This size reduction implies a corresponding reduction in the running time of the algorithm. We refer to the use of HNC in a supervised manner as a *supervised normalized cut* (SNC).

In the comparative study of Baumann et al. [4], we chose Gaussian similarity weights, which are monotone functions of the Euclidean distances between the feature vectors $v^{(i)}$ and $v^{(j)}$ associated with objects i and j as in Equation (1). The parameter ϵ representing the scaling factor is one of the two tuning parameters of SNC. The second one is the relative weighting parameter of the two objectives, λ, in the linearized HNC. The minimum cut problems were solved with the MATLAB implementation of the HPF pseudoflow algorithm, version 3.23, of Chandran and Hochbaum [6] that was presented in Hochbaum [17].

The K-supervised normalized cut (KSNC) is a variant of SNC in which we use q-HNC, $\min_{\emptyset \subset S \subset V} [(C(S, \bar{S}))/(\sum_{i \in S} q_i)]$, with node weights q_i equal to the average class label of the K-nearest labeled objects to i. For example, if $K = 3$ and the three nearest objects to i, in terms of similarity, have labels 0, 1, and 1, then q_i is $2/3$. In contrast to the weights d_i, which capture the pairwise similarities between any pairs, whether or not the nodes are labeled, the weights q_i as defined above capture only the effect of the labeled nodes on the unlabeled nodes. KSNC is therefore a sort of hybrid between SNC and KNN. Here again, we consider the linearized version, which we refer to as KSNC:

$$\min_{\emptyset \subset S \subset V} C(S, \bar{S}) - \lambda \sum_{i \in S} q_i.$$

The tuning parameters for KSNC are the relative weighting parameter of the two objectives λ, the scaling factor of the exponential weights ϵ, and the integer parameter K.

6.2. Commonly Used Machine Learning Techniques

In this section, we provide a brief description of the established classification techniques tested in this study. The reader can find a detailed description of such techniques in Baumann et al. [4]. These techniques can be divided into four groups: decision tree-based techniques, regression-based techniques, similarity-based techniques, and other techniques. Our methods of SNC and KSNC belong to the similarity-based techniques category. A fourth group cannot be quite classified under the first three categories and is referred to as "other":

1. *Decision tree-based machine learning techniques*: In this study, we tested classification and regression trees (CART) and three ensemble methods that combine multiple classification trees into one machine learning technique: ensemble with adaptive boosting (EADA), ensemble with bagging (EBAG), and ensemble with gentle adaptive boosting (EGAB).
2. *Regression-based machine learning techniques*: These techniques include linear regression (LIN), logistic regression (LOG), and Lasso regression (LASSO).
3. *Similarity-based machine learning techniques*: These techniques include the K-nearest neighbors algorithm (KNN) and support vector machines (SVM and SVMR). In the K-nearest neighbors algorithm, the majority label among the K training objects most similar to an unlabeled object is assigned to the object. For support vector machines, we tested two different tuning settings. In the first setting, linear, polynomial, and radial basis function kernels are used for tuning. The algorithm that uses this setting is referred to as SVM. In the second setting, only radial basis function kernels are used for tuning. The algorithm that uses this setting is referred to as SVMR. We note that SVMR utilizes pairwise similarities.
4. *Other machine learning techniques*: Among the techniques in this category are the naïve Bayes classifier (CNB) and artificial neural networks (ANN).

A detailed analysis of the techniques and the source of their respective codes is provided in Baumann et al. [4].

6.3. Performance Measures

The study in Caruana and Niculescu-Mizil [5] demonstrated that different performance measures are highly correlated. We therefore focused on the four most widely used performance measures: F_1-score, precision, recall, and accuracy. Let TP, TN, FP, and FN denote the number of true positives, true negatives, false positives, and false negatives, respectively. Then F_1-score, precision, recall, and accuracy are defined as follows:

$$F_1\text{-score} = \frac{2\,TP}{2\,TP + FP + FN},$$

$$\text{Precision} = \frac{TP}{TP + FP},$$

$$\text{Recall} = \frac{TP}{TP + FN},$$

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}.$$

Note that the F_1-score is the harmonic mean of precision and recall.

6.4. Data Sets

The machine learning techniques were evaluated on 20 data sets that were downloaded from the UCI Machine Learning Repository (see Asuncion and Newman [1]) and the LIBSVM (Library for Support Vector Machines) website (see Chang and Lin [7]). The selected data sets represent a variety of fields including life sciences, physical sciences, engineering, social sciences, business, and others. The data sets differ in the number of objects, the number of features, and the distribution of class labels and therefore required some preprocessing. The collection comprises binary and multiclass classification problems. To avoid bias in the conversion of multiclass problems into binary classification problems, we applied the following selection rule: the multiclass labels (which are all numerical) were sorted in ascending order, and then the first (lowest) label was selected as the positive class and all other labels as the negative class. This ensures that all multiclass data sets are converted in the same way and there is no exploitation of any domain knowledge for the conversion into binary classification problems. Some data sets have missing values. In those sets, we removed the objects that contained missing values. Categorical feature values were replaced by a set of Boolean features (one Boolean feature per category). For a description of each data set and a summary of its characteristics, we refer the reader to Baumann et al. [4].

Extensive experimental results on accuracy, recall, F_1-scores, ranks, and testing times are reported in Baumann et al. [4]. Here, we only provide the results on the relative F_1-scores in Table 1.

6.5. Major Conclusions from the Study

The similarity-based algorithms SNC, SVMR, KSNC, and KNN are all among the top five techniques. The highest average rank as well as the highest average normalized F_1-score was achieved by SNC. With the exception of EBAG, the average performance of all nonsimilarity-based machine learning techniques is considerably lower than the performance of SNC. Perhaps the most surprising result is that SNC is also superior to all other machine learning techniques in terms of robustness. The SNC algorithm achieves not only the highest average normalized F_1-score but also the highest average relative F_1-score and the highest minimum relative F_1-score.

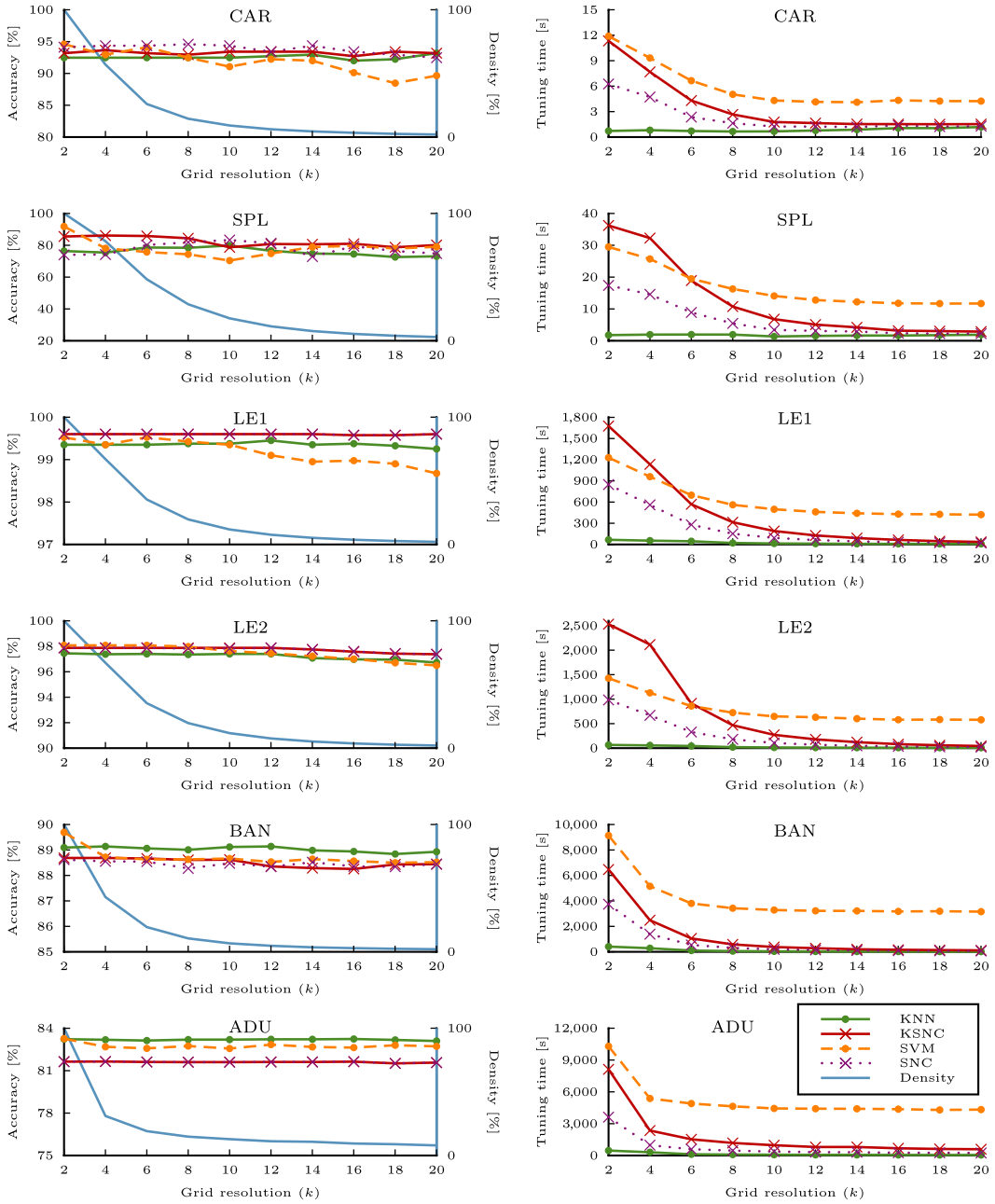
Table 1. Relative F_1-score: In interpreting the table, one can see that SNC and KSNC are the leading techniques in terms of average and minimum relative F_1-score across data sets.

	ANN	CART	CNB	EADA	EBAG	EGAB	LASSO	LIN	LOG	SVM	SVMR	KNN	KSNC	SNC	Average
IRS	**107.7**	**107.7**	**107.7**	0.0[a]	**107.7**	**107.7**	**107.7**	**107.7**	**107.7**	**107.7**	**107.7**	**107.7**	**107.7**	**107.7**	100.0
WIN	100.6	93.8	**102.4**	**102.4**	99.5	99.5	99.5	99.5	99.5	**102.4**	99.5	99.5	99.5	**102.4**	100.0
PAR	95.1	98.9	95.1	100.8	101.1	101.5	95.4	97.9	97.6	93.3	105.0	**105.9**	**105.2**	**107.3**	100.0
SON	99.6	92.2	95.6	**109.9**	**106.7**	**109.7**	97.9	93.2	95.8	100.4	98.4	103.3	100.2	97.0	100.0
GLA	95.3	86.3	85.2	94.3	**114.8**	95.8	92.6	99.2	97.9	**111.6**	105.7	105.5	106.7	109.0	100.0
HEA	100.8	**105.3**	100.7	96.8	104.1	93.5	**104.2**	102.9	102.5	74.2	103.6	**105.7**	102.7	102.9	100.0
HAB	90.1	92.7	107.2	75.5	97.9	78.1	42.6	**135.7**	**134.9**	110.7	87.3	104.8	120.6	121.8	100.0
VER	100.9	98.0	92.9	101.6	102.4	99.5	101.2	99.5	101.8	91.3	**102.9**	100.0	**104.9**	**103.2**	100.0
ION	97.8	98.4	99.3	101.1	102.6	100.9	96.1	96.3	98.5	**103.4**	**103.4**	98.9	100.3	**103.0**	100.0
DIA	104.8	98.2	108.3	94.6	97.2	99.1	94.1	**109.4**	**115.1**	72.7	96.2	93.7	106.5	**110.1**	100.0
BCW	102.0	103.9	103.9	102.7	**104.4**	102.1	**104.2**	103.5	**104.0**	55.8	**104.0**	**104.0**	102.2	103.1	100.0
AUS	100.2	101.9	**104.5**	103.8	103.4	99.1	**104.8**	**105.1**	103.6	59.8	103.8	102.1	103.4	104.4	100.0
BLD	87.4	114.2	**129.7**	108.6	82.0	83.0	52.8	**132.5**	**133.8**	63.3	97.4	98.2	105.3	112.0	100.0
FOU	**116.9**	115.4	86.1	105.7	116.5	106.1	76.7	79.1	78.8	51.2	**116.9**	**116.9**	**116.9**	**116.9**	100.0
TIC	100.2	99.0	86.3	100.2	101.3	101.4	100.5	100.5	100.5	**102.7**	**102.7**	**102.2**	101.0	101.5	100.0
GER	92.3	105.3	101.9	100.4	101.0	100.0	103.3	**115.6**	**112.4**	88.8	95.5	84.8	93.8	104.9	100.0
CAR	101.1	100.5	98.0	**101.9**	**102.2**	**102.0**	100.2	98.3	99.5	93.5	101.8	100.1	100.5	100.5	100.0
SPL	100.2	**106.0**	98.1	105.2	**109.0**	**105.4**	97.0	96.9	96.5	99.0	103.1	90.8	95.8	96.9	100.0
LE1	77.7	132.6	84.0	101.5	141.2	119.5	0.0	0.0	0.0	**149.9**	**151.2**	146.8	147.4	**148.3**	100.0
LE2	107.0	106.5	84.1	93.7	112.2	94.5	82.9	85.4	85.6	98.5	**113.0**	111.9	**112.2**	**112.6**	100.0
Average	98.9	102.8	98.6	100.0[a]	**105.4**	99.9	87.7	97.9	98.3	91.5	105.0	104.1	**106.6**	**108.3**	
Minimum	77.7	86.3	84.0	75.5[a]	82.0	78.1	0.0	0.0	0.0	51.2	87.3	84.8	**93.8**	**96.9**	

Note. The top three performers for each data set, as well as the top performers for the average and minimum, are indicated in bold.

[a]Failure of EADA for the first data set is excluded from the average and minimum.

Figure 8. (Color online) Impact of grid resolution on accuracy, density, and tuning times.



In terms of running time of the techniques, CNB, KNN, cCART, LIN, and LOG are clearly the fastest techniques. SNC has very low evaluation times for data sets that are small in terms of the number of objects. However, the evaluation times increase considerably when the number of objects increases. Also, the evaluation times of ANN, EADA, EBAG, EGAB, SVM, and SVMR go up sharply as the number of objects increases. For the similarity-based machine learning techniques (KNN, KSNC, SNC, and SVM), the evaluation time increase occurs

Table 2. Data sets (after modifications).

Abbreviation	Downloaded from	No. of objects	No. of attributes	No. of positives	No. of negatives	No. of positives/ no. of negatives
IRS	LIBSVM	150	4	50	100	0.50
WIN	LIBSVM	178	13	59	119	0.50
PAR	UCI	195	22	147	48	3.06
SON	UCI	208	60	111	97	1.14
GLA	LIBSVM	214	9	70	144	0.49
HEA	LIBSVM	270	13	120	150	0.80
HAB	UCI	306	3	81	225	0.36
VER	UCI	310	6	210	100	2.10
ION	UCI	351	34	225	126	1.79
DIA	UCI	392	8	130	262	0.50
BCW	UCI	683	10	239	444	0.54
AUS	LIBSVM	690	14	307	383	0.80
BLD	UCI	748	4	178	570	0.31
FOU	LIBSVM	862	2	307	555	0.55
TIC	UCI	958	27	626	332	1.89
GER	UCI	1,000	24	300	700	0.43
CAR	UCI	2,126	21	1,655	471	3.51
SPL	LIBSVM	3,175	60	1,648	1,527	1.08
LE1	UCI	20,000	16	753	19,247	0.04
LE2	UCI	20,000	16	9,940	10,060	0.99

because the number of pairwise similarities grows quadratically in the size of the data set. This hinders the applicability of similarity-based machine learning techniques for very large data sets. However, as discussed in Section 7 and in Baumann et al. [2, 3] and Hochbaum and Baumann [19], the method of sparse computation that generates only the relevant similarities results in sparse similarity matrices even for massively large data sets. With this technique, significant improvements in running time can be achieved with minimal loss in accuracy.

Overall, the study demonstrates that the combinatorial optimization algorithms have consistently best or close–to-best performance, and their performance is also the most robust. An important insight derived from this study is that similarity-based algorithms perform considerably better than nonsimilarity-based machine learning algorithms. This implies that further investigations of effective machine learning techniques should focus on similarity-based algorithms and on combinatorial optimization algorithms.

7. The Challenge of Similarities and the Quadratic Growth: Sparse Computation

In a general data set, there is no natural provided notion of adjacency. For this reason, one may have to construct the complete similarity matrix in which there is a similarity value for all pairs. This corresponds to evaluating a clique graph on the set of objects in the data set. Such an approach becomes prohibitive even for moderately sized data sets, as the number of similarities (or nonzero entries in the matrix) grows as $O(n^2)$ for a data set on n objects. Although there are various "sparsification" techniques known, these all require as input the complete similarity matrix and then remove some of the entries by setting them to zero according to various criteria. The point is that producing the complete similarity matrix in the first place is prohibitive. The technique of sparse computation introduced in Hochbaum and Baumann [19] is designed to determine, prior to computing the similarities, which similarities are "relevant" and compute only those.

The technique named *sparse computation* (Baumann et al. [2, 3], Hochbaum and Baumann [19]) recognizes the relevant pairwise similarities without first computing all pairwise

similarities. Sparse computation effectively removes similarities that it recognizes to be negligible without computing them first. For each pairwise similarity identified as negligible, there is no edge connecting the respective pair, rendering the graph sparse.

Sparse computation is achieved via a process of principal component analysis (PCA) and in fact uses "approximate" PCA on a sample of columns and rows. The PCA projects the feature vectors onto a lower p-dimensional space, most often to three-dimensional space; $p = 3$. Each dimension is then subdivided into k equal intervals, creating k^p blocks. The value of k is referred to as the *resolution*. The similarities are then only computed between blocks that are adjacent along one of the dimensions, or diagonally—that is, within an $L_{\max}$ distance of one block. Consequently, larger values of k correspond to finer resolutions and sparser graphs.

The sparse computation process has several advantages:

1. There are fewer similarities to compute, with substantial reduction in computational effort.
2. The graphs tend to be sparse. This is important as a mitigation of a phenomenon that happens with minimum cuts in very large (in terms of number of nodes) dense graphs. Even if the arc weights are small, an approximately balanced partition will have a large cut value because of the presence of a large number of arcs in the cut, roughly of quadratic size in the number of nodes. For this reason, minimum cuts in large, dense graphs tend to result in a very unbalanced partition, with almost all nodes on one side and the remaining few on the other. The approach of sparse computing then effectively sparsifies the graph and removes many edges (or arcs) but without computing their similarity values first.
3. In experimentation on very large benchmark data sets, the sparse computation approach appears to not have an adverse effect on accuracy, and occasionally, it even improves it compared with lower resolutions or complete matrices (if the latter are possible to compute). This is manifested for several data sets in Figure 8.

Sparse computation was shown to be effective at retaining accuracy while reducing considerably graph density with increasing resolutions Hochbaum and Baumann [19]. The effect of increasing the resolutions is illustrated in a computational study from Hochbaum and Baumann [19], with results depicted in Figure 8. In that figure, the experiments compare the accuracy and tuning time as a function of increased resolution (and reduced density), which is controlled by increasing the value of k to the accuracy achieved with a complete matrix. The complete matrix corresponds to the value of $k = 2$.

The data sets tested for the effectiveness of sparse computation in Hochbaum and Baumann [19] are listed in Table 2. The last column of the table lists the ratio of the number of objects in the positive class to the number of objects in the negative class. For Figure 8 we only tested data sets of sizes up to $50,000$, which was the limit of being able to store the complete matrix for the respective clique graph. For other data sets, the effect was tested as well but not compared with the full matrix. See Hochbaum and Baumann [19] for details.

The sparse computation method considers each pixel $i \in V$ as an object defined by its feature vector R_i. The method first projects these feature vectors onto a low-dimensional space of dimension p using a fast approximation of principal component analysis. The low-dimensional space is then subdivided into a grid with κ sections per dimension, resulting in a total of κ^p grid blocks. Pairs of pixels are considered relevant similarities if the pixels fall in the same or adjacent grid blocks in the low-dimensional space. These pairs of objects are selected for the edge set E. For these pairs, we will compute the pairwise similarities.

The use of sparse computation provides various advantages, such as improving the running time. For general machine learning problems, sparse computation significantly reduces the running time of similarity-based classifiers (Baumann et al. [3], Hochbaum and Baumann [19]), such as SNC, k-nearest neighbors, and support vector machines with radial basis function kernels. We have not tested the effect of sparse computation on the use of the spectral method

for data mining. The spectral method was compared with HNC for image segmentation data, where the graph is inherently sparse. But for data mining, it was not considered, likely because of the density of the Laplacian matrix. With sparse computation, it would be interesting to conduct a study to test the efficacy of the spectral method for general data mining.

8. Conclusions

We present here combinatorial methods based on flow procedures that are effective in image segmentation and as a general machine learning technique. The input to the methods includes pairwise comparisons that are shown to enhance the quality of clustering and classification while being implemented so as to enable scalability to very-large-scale data sets.

Acknowledgments

This research has been funded in part by the National Science Foundation [Award CMMI-1760102].

References

[1] A. Asuncion and D. Newman UCI machine learning repository. Accessed July 15, 2012, http://www.ics.uci.edu/mlearn/MLRepository.html, 2007.

[2] P. Baumann, D. S. Hochbaum and Q. Spaen. Sparse-reduced computation: Enabling mining of massively-large data sets. M. De Marsico, G. Sanniti di Baja, and A. Fred, eds. *Proc. 5th Internat. Conf. on Pattern Recognition Applications and Methods* (SCITEPRESS, Setúbal, Portugal), 224–231, 2016.

[3] P. Baumann, D. S. Hochbaum, and Q. Spaen. High-performance geometric algorithms for sparse computation in big data analytics. *Proc. IEEE Internat. Conf. on Big Data* (IEEE, Piscataway, NJ), 546–555, 2017.

[4] P. Baumann, D. S. Hochbaum and Y. Yang. A comparative study of the leading machine learning techniques and two new optimization algorithms. *European Journal of Operational Research*, Forthcoming, 2018.

[5] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. *Proc. 23rd Internat. Conf. on Machine Learning* (IEEE, Piscataway, NJ), 161–168, 2006.

[6] B. G. Chandran and D. S. Hochbaum. Hochbaum's PseudoFlow (HPF) parametric maximum flow solver. Accessed January 7, 2018, http://riot.ieor.berkeley.edu/Applications/Pseudoflow/parametric.html, 2012.

[7] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2(3):1–27, 2011.

[8] G. B. Coleman and H. C. Andrews. Image segmentation by clustering. *Proceedings of the IEEE* 67(5):773–785, 1979.

[9] T. Cour, S. Yu, and J. Shi. MATLAB normalized cut image segmentation code. Retrieved July 2011, http://www.cis.upenn.edu/∼jshi/software/, 2011.

[10] P. A. Dhawan. *Medical Imaging Analysis* (Wiley-Interscience, Hoboken, NJ), 2003.

[11] I. S. Dhillon, Y. Q. Guan, and B. Kulis. Kernel k-means: Spectral clustering and normalized cuts. *Proc. Internat. Conf. on Knowledge Discovery and Data Mining* (ACM, New York), 2004.

[12] I. S. Dhillon, Y. Q. Guan, and B. Kulis. Weighted graph cuts without eigenvectors: A multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(11):1944–1957, 2007.

[13] B. Fishbain, D. S. Hochbaum, and Y. Yang. A new approach for real-time target tracking in videos. *SPIE Newsroom* (January 29), https://doi.org/10.1117/2.1201301.004648, 2013.

[14] D. S. Hochbaum. Solving integer programs over monotone inequalities in three variables: A framework for half integrality and good approximations. *European Journal of Operational Research* 140(2):291–321, 2002.

[15] D. S. Hochbaum. The pseudoflow algorithm: A new algorithm for the maximum-flow problem. *Operations Research* 56(4):992–1009, 2008.

[16] D. S. Hochbaum. Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32(5):889–898, 2010.

[17] D. S. Hochbaum, HPF–Hochbaum's PseudoFlow. Accessed December 2, 2012, http://riot.ieor.berkeley.edu/Applications/Pseudoflow/maxflow.html, 2012.

[18] D. S. Hochbaum. A polynomial time algorithm for Rayleigh ratio on discrete variables: Replacing spectral techniques for expander ratio, normalized cut and Cheeger constant. *Operations Research* 61(1):184–198, 2013.

[19] D. S. Hochbaum and P. Baumann. Sparse computation for large-scale data mining. *IEEE Transactions on Big Data* 2(2):151–174, 2016.

[20] D. S. Hochbaum, C. N. Hsu, Y. T. Yang. Ranking of multidimensional drug profiling data by fractional adjusted bi-partitional scores. *Bioinformatics* 28(12):106–114, 2012.

[21] D.S. Hochbaum, C. Lu, and E. Bertelli. Evaluating performance of image segmentation criteria and techniques. *EURO Journal on Computational Optimization.* 1(1–2):155–180, 2013.

[22] M. S. Hosseini, B. N. Araabi, and H. Soltanian-Zadeh. Pigment melanin: Pattern for iris recognition. *IEEE Transactions on Instrumentation and Measurement* 59(4):792–804, 2010.

[23] T. N. Pappas. An adaptive clustering algorithm for image segmentation. *IEEE Transactions on Signal Processing* 40(4):901–914, 1992.

[24] D. L. Pham, C. Y. Xu, and J. L. Prince. Current methods in medical image segmentation. *Annual Review of Biomedical Engineering* 2:315–337, 2000.

[25] C. A. Roobottom, G. Mitchell, and G. Morgan-Hughes. Radiation-reduction strategies in cardiac computed tomographic angiography. *Clinical Radiology* 65(11):859–867, 2010.

[26] L. G. Shapiro and G. C. Stockman. *Computer Vision* (Prentice-Hall, Upper Saddle River, NJ), 2001.

[27] E. Sharon, M. Galun, D. Sharon, R. Basri, and A. Brandt. Hierarchy and adaptivity in segmenting visual scenes. *Nature* 442(7104):810–813, 2006.

[28] J. B. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8):888–905, 2000.

[29] Q. Spaen, D. S. Hochbaum, and R. Asín-Achá. HNCcorr: A novel combinatorial approach for cell identification in calcium-imaging movies. Working paper, University of California, Berkeley, Berkeley. https://arxiv.org/abs/1703.01999, 2017.

[30] D. A. Tolliver and G. L. Miller. Graph partitioning by spectral rounding: Applications in image segmentation and clustering. *IEEE Conference on Computer Vision and Pattern Recognition*, (IEEE, Piscataway, NJ), 1053–1060, 2006.

[31] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15(11):1101–1113, 1993.

[32] E. P. Xing and M. I. Jordan. On semidefinite relaxations for normalized k-cut and connections to spectral clustering. Technical Report UCB/CSD-3-1265, University of California, Berkeley, Berkeley, 2003.

[33] Y. T. Yang, B. Fishbain, D. S. Hochbaum, E. B. Norman and E. Swanberg. The supervised normalized cut method for detecting, classifying and identifying special nuclear materials. *INFORMS Journal on Computing*, 26(1):45–58, 2014.