Entropy-Rate Clustering: Cluster Analysis via Maximizing a Submodular Function subject to a Matroid Constraint

Ming-Yu Liu, *Member, IEEE,* Oncel Tuzel, *Member, IEEE,* Srikumar Ramalingam, *Member, IEEE,* and Rama Chellappa, *Fellow, IEEE*

Abstract—We propose a new objective function for clustering. This objective function consists of two components: the entropy rate of a random walk on a graph and a balancing term. The entropy rate favors formation of compact and homogeneous clusters, while the balancing function encourages clusters with similar sizes and penalizes larger clusters that aggressively group samples. We present a novel graph construction for the graph associated with the data and show that this construction induces a matroid— a combinatorial structure that generalizes the concept of linear independence in vector spaces. The clustering result is given by the graph topology that maximizes the objective function under the matroid constraint. By exploiting the submodular and monotonic properties of the objective function, we develop an efficient greedy algorithm. Furthermore, we prove an approximation bound of $\frac{1}{2}$ for the optimality of the greedy solution. We validate the proposed algorithm on various benchmarks and show its competitive performances with respect to popular clustering algorithms. We further apply it for the task of superpixel segmentation. Experiments on the Berkeley segmentation dataset reveal its superior performances over the state-of-the-art superpixel segmentation algorithms in all the standard evaluation metrics.

Index Terms—clustering, superpixel segmentation, graph theory, information theory, submodular function, discrete optimization

1 Introduction

Clustering is a fundamental task in many domains such as machine learning, computer vision, marketing, and biology. In almost every scientific field dealing with empirical data, researchers attempt to get a first impression on their data by identifying groups of similar characteristics. Several clustering methods have been proposed in different communities, and many of them have promising performances. However, they are usually based on different assumptions, and it is difficult to compare one criterion to another. Furthermore, most desirable criteria lead to NP-hard problems. Thus, further progress in clustering hinges on the careful design of new objective functions applicable to existing or newer problems with provable theoretical guarantees and promising performance on standard datasets. This is precisely the goal of this paper.

Among a wide variety of clustering algorithms, some compute clusters using a single objective function, some obtain clusters recursively using intermediate cost functions, and a few others identify clusters based on a particular projection (subspace, manifold) of data points. This work belongs to the first class. We formulate the clustering problem as a graph topology selection problem where data points and their pairwise relations are respectively mapped to the vertices and edges in a graph. Clustering is then solved via finding a graph topology maximizing the objective function.

Various objective functions have been proposed to measure the quality of a given cluster. However, the notion of a ’good’ cluster is problem dependent. Quite often it is possible to generate an example for which a given objective function fails. In this work, we are interested in obtaining compact, homogeneous, and balanced clusters. In a compact cluster, data points are close to each other. In a homogeneous cluster data points share similar inter-element properties. The notion of balanced clusters refers to avoiding large clusters that aggressively group samples. In order to obtain clusters with these qualities, we propose a novel objective function consisting of two components: 1.) the entropy rate of a random walk on a graph and 2.) a balancing term on the cluster distribution. The entropy rate favors compact and homogeneous clusters whereas the balancing term encourages clusters with similar sizes. They are motivated by the principle of maximum entropy [1]: we seek a graph topology such that the resulting random walk and cluster membership distribution yield a large uncertainty.

Our formulation leads to an algorithm with a provable bound on the optimality of the solution. We show that our objective function is a monotonically increasing submodular function. Submodularity appears in many real world applications such as facility location, circuit design, and set covering. It can be related to convexity through the Lovász extension while also sharing some similarities to concavity [2]. Knowing whether a function is submodular enables us to better understand the underlying optimization problem. In general, maximization of submodular functions leads to NP-hard problems, for which the global optimum solution is difficult to obtain. Nevertheless, by using a greedy algorithm and exploiting the matroid structure in our problem formulation, we obtain a bound of $\frac{1}{2}$ on the optimality of the solution. Recently,

- *MY. Liu, O. Tuzel, and S. Ramalingam are with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139*
E-mail: {mliu, oncel, ramalingam}@merl.com
- *R. Chellappa is with University of Maryland College Park, MD 20742*
E-mail: rama@umiacs.umd.edu

maximization of submodular functions with various constraints has been applied in several real word problem domains: sensor placement [3] (subject to a cardinality constraint), outbreak detection in networks [4] (subject to a modular cost constraint), and word alignment [5] (subject to a matroid constraint).

We evaluate the proposed algorithm using standard datasets in the UCI repository and compare it to the state-of-the-art clustering algorithms. In addition, we study a particular clustering problem in computer vision — the superpixel segmentation problem [6]. Superpixel segmentation is a processing which divides an image into disjoint and perceptually uniform regions, termed superpixels. A superpixel representation greatly reduces the number of primitives in an image and provides coherent spatial support for feature computations. It has become a common preprocessing step for many advanced vision algorithms [7][8][9][10]. The desired properties of a superpixel segmentation algorithm depend on the application of interest. We list some of the desired properties below:

- Every superpixel should overlap with only one object.
- The set of superpixel boundaries should be a superset of object boundaries.
- The mapping from pixels to superpixels should not reduce the achievable performance of the intended application.
- The above properties should be obtained with as few superpixels as possible.

We show that the proposed objective function possesses these required properties. Specifically, the entropy rate favors compact and homogeneous clusters—encouraging division of images on perceptual boundaries, whereas the balancing term encourages superpixels with similar sizes—avoiding large superpixels straddling multiple objects.

1.1 Related Work

There is a large body of work in clustering. Below we only review a few related works and refer the interested reader to survey papers such as [11], [12], [13], [14].

1.1.1 Graph-theoretic approaches

Graph-theoretic clustering methods are preferred when only the pairwise relations of data are available. Some representative works in this category include [15], [16], [17], [18]. In [15], clustering is achieved by partitioning a minimal spanning tree into disjoint sets. It first constructs a minimal spanning tree from data graph and then sequentially deletes edges whose similarity score are significantly smaller than the neighboring edges. The edge deletion process uses a single threshold and does not accommodate intra-cluster variation. The proposed algorithm also forms disjoint sets via spanning trees, but the formation is attained through maximizing a submodular function defined on the graph topology.

Wu and Leahy [16] propose using the min-cut algorithm to iteratively bisecting the graph. The min-cut cost can be solved optimally within each iteration. Nevertheless it prefers dividing a small set of isolated vertices and is vulnerable to outliers. This drawback is elegantly handled in a seminal paper on normalized cut (NCut) [17] using a normalization term favoring balanced clusters. NCut is related to spectral clustering[19]. While NCut is effective, computing an NCut solution requires eigen-decomposition, which is computationally intense for large-scale problems [20]. We formulate clustering as a graph topology optimization problem and propose an objective function favoring the formation of compact, homogeneous, and balanced clusters. The resulting algorithm is efficient and can be easily applied to large datasets.

Correlation clustering [18] seeks a clustering output that maximizes both the number of similar edges within clusters and the number of dissimilar edges between clusters. In some sense, the entropy rate function encourages a similar objective; however, the balancing function further promotes the formation of clusters having a similar size. Our problem formulation is related to the K-balanced partitioning problem [21] where a graph is partitioned in K connected components and the number of elements in each component is about the same. Our balancing function imposes a soft constraint for obtaining clusters of similar sizes.

1.1.2 Random walk modeling

Meilă and Shi [22] discuss the link between the NCut objective function and a random walk model. They show that solving the NCut partition is equivalent to finding the low conductivity set in a random walk. Harel and Koren [23] propose a separation operator based on the escape probability in a random walk, to sharpen the distinction between intra-cluster links and inter-cluster links. The operator is applied repeatedly until the graph is divided into several disconnected components.

Yen et al. [24] propose a similarity measure for clustering, which is based on the average passing time between two states in a random walk. Computing this metric requires solving the pseudo inverse of the graph Laplacian matrix. Leo [25] presents an interactive image segmentation algorithm based on random walk modeling. With user-specified labels on some pixels, it computes the probabilities that a random walk reaches these labeled pixels starting from an unlabeled pixel. The unlabeled pixel is then assigned the label with the largest probability.

1.1.3 Information-theoretic approaches

Banerjee et al. [26] propose a K-means like clustering algorithm based on mutual information. The length of minimal spanning trees is used as an estimator of mutual information in a clustering formation in [27]. Our clustering objective function is also derived using information theory where the entropy rate and entropy are used to measured the randomness in a random process and random variable respectively.

1.1.4 Submodular objective functions

Narasimhan et al. [28] present two submodular clustering objective functions. The first one is based on the minimal distance between the elements of different clusters; whereas the second is related to the description length of the clusters. Nagano et al. [29] use an objective function based on minimum average cost. Clustering with these objective functions leads to submodular function minimization problems and can be solved optimally in polynomial time. Our formulation leads to a constrained submodular maximization problem, which is more difficult. Recently, Jegelka and Bilmes [30] propose a submodular cost function for image segmentation called the cooperative graph cut. It gives bias to cutting edges exhibiting cooperative patterns. To solve the cooperative cut problem, they derive a bounding function and show that the st-cut algorithm [31] can be used to iteratively minimize the bounds to produce the desired graph partitioning.

1.1.5 Superpixel segmentation

Graph-based image segmentation work of Felzenszwalb and Huttenlocher (FH) [32], mean shift [33], and watershed [34] are three most popular superpixel segmentation algorithms. FH and watershed are fast; mean shift is robust to local variations. However, they tend to produce superpixels with irregular sizes and shapes, which sometimes straddle multiple objects as pointed out in [35], [36].

Ren and Malik [6] propose using NCut for superpixel segmentation. NCut has the nice property of producing superpixels with similar sizes and compact shapes which are preferred for some vision algorithms [6], [7]. One drawback of NCut is its computational requirement—it takes several minutes for segmenting an image of moderate (481x321) size. Levinshtein et al. [35] propose the TurboPixel algorithm as an efficient alternative. TurboPixel is based on evolving boundary curves from seeds uniformly placed in the image. Recently Veksler et al. [36] pose the superpixel segmentation problem as a GraphCut [37] problem. The regularity is enforced through a dense patch assignment technique for allowable pixel labels.

These methods produce nice image tessellations. Nevertheless, they tend to sacrifice finer image details owing to their preference for smooth boundaries. This is reflected in the low boundary recall reported in [35], [36]. In contrast, our balancing objective, which regularizes the cluster sizes, avoids the over-smoothing problem and hence better preserves object boundaries.

Moore et al. [38], [39] propose an alternative approach for obtaining superpixels aligned with a grid. In [38], a greedy algorithm is used to sequentially cut images along some vertical and horizontal strips; whereas in [39], the problem is solved using a GraphCut algorithm [37].

Superpixel segmentation can also be jointly solved with stereo matching. Taguchi et al. [40] propose an EM-like iterative procedure to jointly estimate scene depth and segmentation using various cues. Bleyer et al. [41] pose the joint estimation problem in an energy minimization framework.

1.2 Contributions

The main contributions of this paper are listed below:

- We pose the clustering problem as a maximization problem on a graph and present a novel objective function on the graph topology. This function consists of an entropy rate term and a balancing term for obtaining clusters with desired properties.
- We prove that the entropy rate and the balancing function are monotonically increasing and submodular.
- By embedding our problem in a matroid structure and using the properties of the objective function, we present an efficient greedy algorithm with an approximation bound of $\frac{1}{2}$.
- We evaluate the proposed algorithm for clustering using standard datasets and show that it renders improved performances in various clustering performance metrics.
- We show that the proposed algorithm significantly outperform many state-of-the-art superpixel segmentation algorithms in the standard performance metrics on the Berkeley segmentation benchmark— a reduced undersegmentation error up to 50%, a reduced boundary miss rate up to 40%, and a tighter bound on achievable segmentation accuracy. In addition, the proposed algorithm is efficient— takes only about 2.5 seconds to segment an image of size 481x321.

The paper is organized as follows. The notations and background discussions are given in Section 2. We present the objective function in Section 3 and discuss its optimization in Section 4. Extensive experimental validations are provided in Section 5. We conclude and discuss some promising future research directions in Section 6. A preliminary version of this work appeared as a superpixel segmentation study in [42]. In this paper, we extend it for the general clustering problem and provide additional experimental validation.

2 Preliminaries

In this section, we introduce the mathematical preliminaries, including graph notations, random walk models, and information-theoretic metrics. They are used for formulating the clustering objective function. We also discuss submodularity, monotonicity, and matroid concepts that are used for analyzing the properties of the objective function and the optimality of the optimization procedure.

Graph representation: We use $G = (V, E)$ to denote an undirected graph where V is the vertex set and E is the edge set. The vertices and edges are denoted by v_i and $e_{i,j}$ respectively. The similarity between vertices is given by the nonnegative weight function $w : E \rightarrow \mathbb{R}^+ \cup \{0\}$. In an undirected graph, the edge weights are symmetric, i.e. $w_{i,j} = w_{j,i}$.

Graph partition: A graph partition S refers to a division of the vertex set V into disjoint subsets $\mathcal{S} = \{S_1, S_2, ..., S_K\}$ such that $S_i \cap S_j = \emptyset$ for $i \neq j$ and $\bigcup_i S_i = V$. We pose the graph partition problem as a subset selection problem. Our goal is to select a subset of edges $A \in E$ such that the resulting graph (V, A) consists of K connected components/subgraphs.

Entropy: The uncertainty of a random variable is measured by entropy H. The entropy of a discrete random variable, X, with a probability mass function, p_X, is defined by

$$H(X) = -\sum_{x \in \mathcal{X}} p_X(x) \log p_X(x) \tag{1}$$

where $\mathcal{X}$ is the support of X. The conditional entropy, $H(X|Y)$, quantifies the remaining uncertainty in X given that the value of a dependent random variable, Y, is known. It is defined as

$$\begin{aligned} H(X|Y) &= \sum_{y \in \mathcal{Y}} p_Y(y) H(X|Y=y) \\ &= -\sum_{y \in \mathcal{Y}} p_Y(y) \sum_{x \in \mathcal{X}} p_{X|Y}(x|y) \log p_{X|Y}(x|y) \end{aligned} \tag{2}$$

where $\mathcal{Y}$ is the support of Y and $p_{X|Y}$ is the conditional probability mass function.

Entropy rate: The entropy rate quantifies the uncertainty of a stochastic process $\boldsymbol{X} = \{X_t | t \in T\}$ where T is some index set. For a discrete random process, the entropy rate is defined as an asymptotic measure

$$\mathcal{H}(\boldsymbol{X}) = \lim_{t \to \infty} H(X_t | X_{t-1}, X_{t-2}, ..., X_1), \tag{3}$$

which measures the remaining uncertainty of the random process after observing the past trajectory. For a stationary stochastic process, the limit in (3) always exists. In the case of a stationary $1st$-order Markov process, the entropy rate has a simple form

$$\begin{aligned} \mathcal{H}(\boldsymbol{X}) &= \lim_{t \to \infty} H(X_t | X_{t-1}) \\ &= \lim_{t \to \infty} H(X_2 | X_1) = H(X_2 | X_1). \end{aligned} \tag{4}$$

The first equality is due to the $1st$-order Markov property whereas the second equality is a consequence of stationarity. For more details, one can refer to [43, pp.77].

Random walks on graphs: Let $\boldsymbol{X} = \{X_t | t \in T, X_t \in V\}$ be a random walk on the graph $G = (V, E)$ with a nonnegative similarity measure w. We use a random walk model described in [43, pp.78]— the transition probability is defined as

$$p_{i,j} = Pr(X_{t+1} = v_j | X_t = v_i) = \frac{w_{i,j}}{w_i} \tag{5}$$

where $w_i = \sum_{k: e_{i,k} \in E} w_{i,k}$ is the sum of incident weights of the vertex v_i, and the stationary distribution is given by

$$\boldsymbol{\mu} = (\mu_1, \mu_2, ..., \mu_{|V|})^T = (\frac{w_1}{w_T}, \frac{w_2}{w_T}, ..., \frac{w_{|V|}}{w_T})^T \tag{6}$$

where $w_T = \sum_{i=1}^{|V|} w_i$ is the normalization constant. For a disconnected graph, the stationary distribution is not unique. However, $\boldsymbol{\mu}$ in (6) is always a stationary distribution. It can be easily verified through $\boldsymbol{\mu} = P^T \boldsymbol{\mu}$ where $P = [p]_{i,j}$ is the transition matrix. The entropy rate of the random walk can be computed by applying (2)

$$\begin{aligned} \mathcal{H}(\boldsymbol{X}) &= H(X_2 | X_1) = \sum_i \mu_i H(X_2 | X_1 = v_i) \\ &= -\sum_i \sum_j \frac{w_{i,j}}{w_T} \log \frac{w_{i,j}}{w_T} + \sum_i \frac{w_i}{w_T} \log \frac{w_i}{w_T} \end{aligned} \tag{7}$$

Submodularity: Let E be a finite set. A set function, $F: 2^E \to \mathbb{R}$, is submodular if

$$F(A \cup \{a_1\}) - F(A) \geq F(A \cup \{a_1, a_2\}) - F(A \cup \{a_2\}) \tag{8}$$

or, equivalently,

$$\delta F_{a_1}(A) \geq \delta F_{a_1}(A \cup \{a_2\} \tag{9}$$

for all $A \subseteq E$ and $a_1, a_2 \in E \setminus A$ where

$$\delta F_{a_1}(A) \equiv F(A \cup \{a_1\}) - F(A) \tag{10}$$

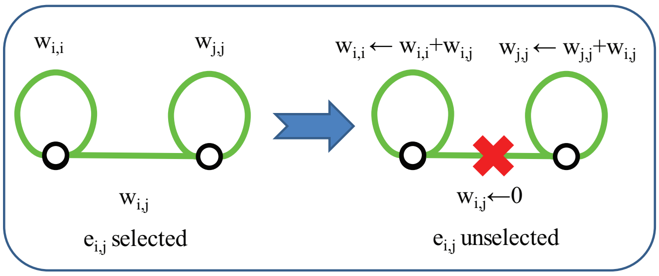


Fig. 1. Illustration of the graph construction. If an edge $e_{i,j}$ is unselected in cluster formation, its weight is redistributed to the loops of the two vertices.

is the marginal gain obtained by adding the element a_1 to the set A. This property is also referred as the diminishing return property, which says that the gain of a module is less if it is included in a later stage [44].

Monotonically increasing set function: A set function F is monotonically increasing if $F(A) \leq F(A \cup \{a_1\})$ for all $A \subseteq E$. We sometimes refer this property as monotonicity in the paper.

Matroid: A matroid is an ordered pair $M = (E, \mathcal{I})$ consisting of a finite set E and a collection $\mathcal{I}$ of subsets of E satisfying the following three conditions:

1) $\emptyset \in \mathcal{I}$.
2) If $I \in \mathcal{I}$ and $I' \subseteq I$, then $I' \in \mathcal{I}$.
3) If I_1 and I_2 are in $\mathcal{I}$ and $|I_1| < |I_2|$, then there is an element e of $I_2 - I_1$ such that $I_1 \cup e \in \mathcal{I}$.

The members of $\mathcal{I}$ are the independent sets of M. Note that there exist several other definitions for matroids which are equivalent. For more details, one can refer to [45, pp.7∼15].

Later in the paper we prove that our objective function is monotonically increasing and submodular.

3 Problem Formulation

We pose clustering as a graph partitioning problem. To partition the graph into K clusters, we search for a graph topology that has K connected subgraphs and maximizes the proposed objective function.

3.1 Graph Construction

We map a dataset to a graph $G = (V, E)$ with vertices denoting the data points and the edge weights denoting the pairwise similarities given in the form of a similarity matrix. There are many ways for generating such a mapping. Some examples include the fully-connected graph, a local fixed-grid graph, or a nearest-neighbor graph. The proper choice of the graph structure is itself an important problem in clustering [46]; however, it is not the focus of the paper. We simply map a dataset into a k-nearest neighbor graph for clustering. For superpixel segmentation, we exploit the image grid structure and use a 8-connected graph.

Our goal is to partition the graph into disjoint components. It is achieved by selecting a subset of edges $A \subseteq E$ such that the resulting graph, $G = (V, A)$, contains exactly K connected subgraphs. In addition, we also assume that every vertex of

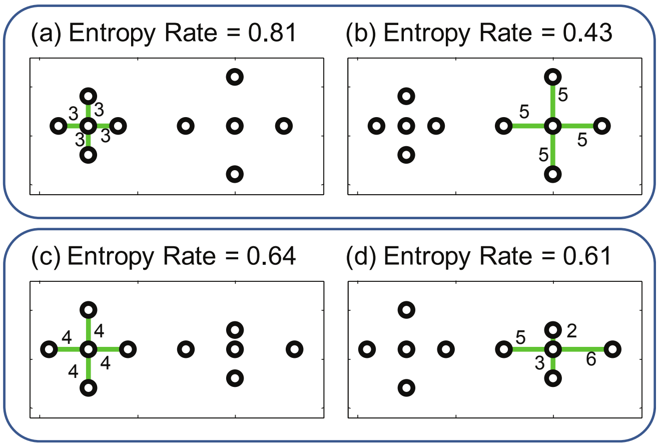


Fig. 2. We show the role of entropy rate in obtaining compact and homogeneous clusters. We use a Gaussian kernel to convert the distances, the numbers next to the edges, to similarities. Each of these clustering outputs contains six different clusters shown as connected components. As described in Section 3, every vertex has a loop which is not shown. The entropy rate of the compact cluster in (a) has a higher objective value than that of the less compact one in (b). The entropy rate of the homogeneous cluster in (c) has a higher objective value than that of the less homogeneous one in (d).

the graph has a self-loop. Although the self-loops do not affect graph partitioning, they are necessary for the proposed random walk model. When an edge is not included in A, we increase the edge weight of the self-loop of the associated vertices in such a way that the total incident weight for each vertex remains constant (See Figure 1).

3.2 Entropy Rate

We use the entropy rate of the random walk on the constructed graph as a criterion to obtain compact and homogeneous clusters. The proposed construction leaves the stationary distribution of the random walk (6) unchanged[1] where the set functions for the transition probabilities $p_{i,j} : 2^E \rightarrow \mathbb{R}$ are given below:

$$p_{i,j}(A) = \begin{cases} \frac{w_{i,j}}{w_i} & \text{if } i \neq j \text{ and } e_{i,j} \in A, \\ 0 & \text{if } i \neq j \text{ and } e_{i,j} \notin A, \\ 1 - \frac{\sum_{j: e_{i,j} \in A} w_{i,j}}{w_i} & \text{if } i = j. \end{cases} \tag{11}$$

Consequently, the entropy rate of the random walk on $G = (V, A)$ can be written as a set function:

$$\mathcal{H}(A) = -\sum_i \mu_i \sum_j p_{i,j}(A) \log(p_{i,j}(A)) \tag{12}$$

Although inclusion of any edge in set A increases the entropy rate, this increase is larger when selecting edges that form compact and homogeneous clusters, as shown in Figure 2.

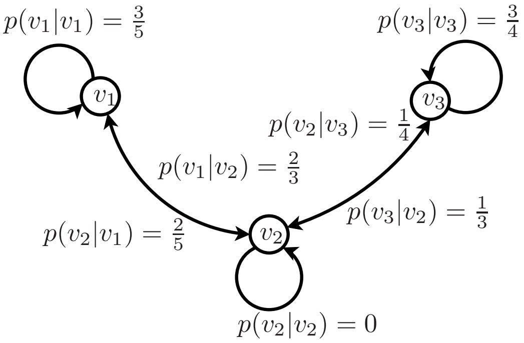

$p(v_1|v_1) = \frac{3}{5}$
$p(v_3|v_3) = \frac{3}{4}$
v_1
v_3
$p(v_2|v_3) = \frac{1}{4}$
$p(v_1|v_2) = \frac{2}{3}$
$p(v_3|v_2) = \frac{1}{3}$
$p(v_2|v_1) = \frac{2}{5}$
v_2
$p(v_2|v_2) = 0$

Fig. 3. Illustration of the transition probabilities given selecting edges $e_{1,2}$ and $e_{2,3}$.

We illustrate the computation of the entropy rate under the proposed graph construction using the following example which is also shown in Figure 3. Given a graph with three vertices $\{v_1, v_2, v_3\}$ and the input similarity matrix

$$W = \begin{pmatrix} - & 2.0 & 3.0 \\ 2.0 & - & 1.0 \\ 3.0 & 1.0 & - \end{pmatrix} \tag{13}$$

the task is to compute the entropy rate, $\mathcal{H}(\{e_{1,2} \cup e_{2,3}\})$; i.e. the entropy rate of the random walk when selecting the edges $e_{1,2}$ and $e_{2,3}$ as shown in Figure 3. From (6) the stationary distribution of the random walk is equal to

$$\boldsymbol{\mu} = (\frac{5}{12}, \frac{3}{12}, \frac{4}{12})^T, \tag{14}$$

and the transition matrix takes the following form

$$P = \begin{pmatrix} \frac{3}{5} & \frac{2}{5} & 0 \\ \frac{2}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{4} & \frac{3}{4} \end{pmatrix}. \tag{15}$$

The entropy rate is then equal to $\mathcal{H}(\{e_{1,2} \cup e_{2,3}\}) = 0.905$.

We establish the following result on the entropy rate of the random walk model.

Lemma 1: The entropy rate of the random walk on the graph $\mathcal{H} : 2^E \rightarrow \mathbb{R}$ is a monotonically increasing submodular function under the proposed graph construction.

It is easy to see that the entropy rate is monotonically increasing, since the inclusion of any edge increases the uncertainty of a jump of the random walk. The diminishing return property comes from the fact that the increase in uncertainty from selecting an edge is less in a later stage because it is shared with more edges. The proof is given in the supplementary material.

3.3 Balancing Function

We utilize a balancing function, which encourages grouping of data points into clusters that have similar sizes. Let A be the selected edge set, N_A be the number of connected components in the graph, and Z_A be the distribution of the cluster membership. For instance, let the graph partitioning for the edge set A be $\mathcal{S}_A = \{S_1, S_2, ..., S_{N_A}\}$. Then the distribution of Z_A is equal to

$$p_{Z_A}(i) = \frac{|S_i|}{|V|}, \quad i = \{1, ..., N_A\}, \tag{16}$$

[1] The total incident weight to a vertex remains unchanged since an edge weight contributes to the total incident weight to a vertex either via a non-loop edge or a self-loop.

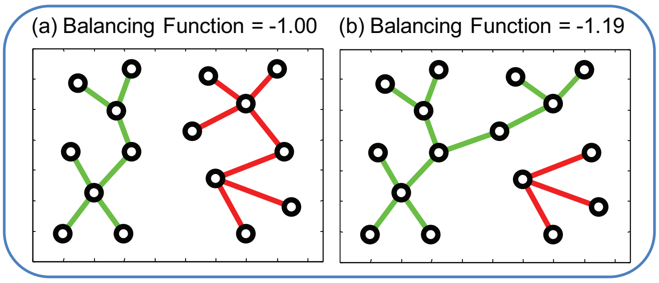

(a) Balancing Function = -1.00
(b) Balancing Function = -1.19

Fig. 4. We show the role of the balancing function in obtaining clusters of similar sizes. The connected components show the different clusters. The balancing function has a higher objective value for the balanced clustering in (a) compared to the less balanced one in (b).

and the balancing term is given by

$$\begin{aligned}\mathcal{B}(A) &\equiv H(Z_A) - N_A \\ &= -\sum_i p_{Z_A}(i)\log(p_{Z_A}(i)) - N_A. \end{aligned} \tag{17}$$

The entropy, $H(Z_A)$, favors clusters with similar sizes, while the cluster number term, N_A, is minimized by grouping data points. In Figure 4, we show an example of the preference where a more balanced partitioning is preferred for a fixed number of clusters.

Similar to the entropy rate, the balancing function is also a monotonically increasing and submodular function as stated in the following lemma:

Lemma 2: The balancing function $\mathcal{B} : 2^E \to \mathbb{R}$ is a monotonically increasing submodular function under the proposed graph construction.

The proof is given in the supplementary material.

The objective function combines the entropy rate and the balancing function and, therefore, favors compact, homogeneous, and balanced clusters. Clustering is achieved via optimizing the objective function with respect to the edge set:

$$\begin{aligned} \max_{A\subseteq E} \quad & \mathcal{F}(A) \\ \text{subject to} \quad & N_A \geq K, \end{aligned} \tag{18}$$

where $\mathcal{F}(A) = \mathcal{H}(A) + \lambda\mathcal{B}(A)$ is the objective function. The parameter, $\lambda \geq 0$, is the weight of the balancing term. Linear combination with nonnegative coefficients preserves submodularity and monotonicity [44], therefore the objective function is also submodular and monotonically increasing. The additional constraint on the number of connected subgraphs enforces exactly K clusters since the objective function is monotonically increasing.

The proposed formulation is closely related to the principle of maximum entropy, which says that the probability distribution that best represents our knowledge of the underlying problem is the one with the largest entropy. This distribution makes the minimal assumption of the problem and is the least biased one [1]. Our objective function encourages a graph partition such that the random walk in the graph has a large entropy rate and the cluster membership distribution has a large entropy. It largely captures the maximum entropy principle.

4 Optimization and Implementation

We present a greedy heuristic for optimizing the proposed objective function. Its optimality, efficient implementation, and complexity are discussed. We also introduce a method to automatically determine the balancing weight parameter λ.

4.1 Greedy Heuristic

One standard approach for maximizing a submodular function is through a greedy algorithm [44]. The algorithm starts with an empty set (a fully disconnected graph, $A = \emptyset$) and sequentially adds edges to the set. At each iteration, it adds the edge that yields the largest gain. The iterations are terminated when the number of connected subgraphs reaches a preset number, $N_A = K$.

In order to achieve an additional speedup, we put a constraint on the edge set, A, which forces that A cannot include any cycle. This constraint immediately ignores the edges that are within a connected subgraph and greatly reduces the number of evaluations required at each iteration of the greedy search. Notice that the ignored edges do not change the partitioning of the graph. Although the constraint leads to a smaller solution space (only tree-structured subgraphs are allowed), the clustering results are similar in practice.

The cycle-free constraint in conjunction with the cluster number constraint, $N_A \geq K$, leads to an independent set definition, which induces a matroid $M = (E, \mathcal{I})$. We establish this in the following lemma:

Lemma 3: Let E be the edge set, and let $\mathcal{I}$ be the set of subsets, $A \subseteq E$, which satisfies: 1.) A is cycle-free and 2.) A constitutes a graph partition with more than or equal to K connected components. Then the pair $M = (E, \mathcal{I})$ is a matroid.

The proof is given in the supplementary material.

With the cycle-free constraint, the graph partition problem becomes a problem of maximizing a submodular function subject to a matroid constraint, given by

$$\begin{aligned} \max_{A\subseteq E} \quad & \mathcal{F}(A) \\ \text{subject to} \quad & A \in \mathcal{I}. \end{aligned} \tag{19}$$

The associated greedy algorithm for solving (19) is similar to the standard one except that it only considers the edges upon whose addition to the current solution set will satisfy the independent set constraint. A pseudocode of the algorithm is given in Algorithm 1.

Maximization of a submodular function subject to a matroid constraint is an active subject in combinatorial optimization research. It is shown in Fisher et al. [47] that the greedy algorithm gives a $\frac{1}{2}$ approximation bound for maximizing a monotonically increasing submodular function. Following the same argument, we achieve the same ($\frac{1}{2}$ approximation) guarantee, which is stated in the following theorem:

Algorithm 1: Pseudocode of the greedy algorithm. The objective function is defined as $\mathcal{F} \equiv \mathcal{H} + \lambda\mathcal{B}$.

```
Data: $G = (V, E)$, $w : E \to \mathbb{R}^+$, $K$, and $\lambda$
Result: $A$
$A \leftarrow \emptyset$, $U \leftarrow E$
repeat
    $\hat{a} \leftarrow \underset{a \in U}{arg \max} \quad \mathcal{F}(A \cup \{a\}) - \mathcal{F}(A)$
    if $A \cup \{\hat{a}\} \in \mathcal{I}$ then
        $A \leftarrow A \cup \{\hat{a}\}$
    $U \leftarrow U - \{\hat{a}\}$
until $U = \emptyset$
```

Theorem 1: Let A_{opt} be an optimal solution for Problem (19), and let A_{greedy} be a solution obtained by applying Algorithm 1. Then the inequality,

$$\frac{\mathcal{F}(A_{greedy}) - \mathcal{F}(\emptyset)}{\mathcal{F}(A_{opt}) - \mathcal{F}(\emptyset)} \geq \frac{1}{2}, \tag{20}$$

holds true
The proof follows immediately by applying Theorem 2.1 in [47].

Theorem 1 shows that the difference between the objective value of the greedy solution and that of the empty set is within $\frac{1}{2}$ of the difference between the optimal solution and the empty set.

4.2 Efficient Implementation

In each iteration, the greedy algorithm selects the edge that yields the largest gain in the objective function subject to the matroid constraint. A naive implementation of the algorithm loops $O(|E|)$ times to add a new edge into A. At each loop, it scans through the edge list to locate the edge with the largest gain; therefore the complexity of the algorithm is $O(|E|^2)$.[2] Since each vertex in our graph is connected to a constant number of few neighbors, the complexity of the algorithm is $O(|V|^2)$. In the following, we show that by exploiting the submodularity of the objective function we can achieve a more efficient implementation, which is called lazy greedy [4].

Initially, we compute the gain of adding each edge to A and construct a max heap structure. At each iteration, the edge with the maximum gain is popped from the heap and included to A. The inclusion of this edge affects the gains of some of the remaining edges in the heap; therefore, the heap needs to be updated. However, the submodular property allows an efficient update of the heap structure. The key observation is that, throughout the algorithm, the gain for each edge can never increase due to the diminishing return property. Therefore, it is sufficient to keep a heap structure where only the gain of the top element is updated but not necessarily the others. Since the top element of the heap is updated and the values for the other elements can only decrease, the top element always has the maximum value.

Although the worst case complexity of the lazy greedy algorithm is $O(|V|^2 \log |V|)$, in practice the algorithm runs much faster than the naive implementation. On average, very few updates are performed on the heap at each iteration, and hence the complexity of the algorithm approximates $O(|V| \log |V|)$. In our superpixel segmentation experiments, it provides a speedup by a factor of 200–300 for image size 481x321 and on average requires 2.5 seconds.

We present a method to automatically adjust the balancing weight, λ. Given an initial user-specified value, λ', the final balancing parameter, λ, is adjusted based on: 1.) the number of clusters, K, and 2.) the data dependent dynamic parameter, β. The cluster number, K, emphasizes balancing term more when a large number of clusters is required. The data dependent term is computed from the input data. It is given by the ratio of the maximal entropy rate increase and the maximal balancing term increase upon including a single edge into the graph $\beta = \frac{\max_{e_{i,j}} \mathcal{H}(e_{i,j}) - \mathcal{H}(\emptyset)}{\max_{e_{i,j}} \mathcal{B}(e_{i,j}) - \mathcal{B}(\emptyset)}$. This choice has the effect of compensating the magnitude difference between the two terms in the objective function. The final balancing parameter is given by $\lambda = \beta K \lambda'$.

5 Experiments

We conducted extensive experiments on clustering and superpixel segmentation to evaluate the proposed algorithm. Throughout the experiments we used $\lambda' = 0.5$ to determine the balancing weight.

5.1 Clustering

We conducted clustering experiments using both standard datasets and challenging vision datasets. They include the ionosphere, letters, satellite, digits, breast cancers, iris, wine, glass, and movement libras datasets from the UCI repository. In the preprocessing step, the samples were normalized to have a zero mean and unit variance for each feature dimension. To measure the distance between the samples, we used the Euclidean distance. Two vision datasets were also used for performance evaluation: the natural scene recognition dataset [48] and MPEG-7 shape database (MPEG-7) [49]. The natural scene dataset contains images from eight different nature scenes ranging from coast, forest, highway, inside city, mountain, open country, street, to tall building. Some of the images are shown in Figure 5. This dataset is very challenging: images of the same scene are usually very different due to the various locations and seasons under which they were captured, while images of different scenes can be very similar due to the common spatial layout. In order to measure the pairwise similarity, we used GIST descriptors [48], the spatial envelope of the image. We used the Euclidean distance in the GIST descriptor space as the distance measure. The MPEG-7 datasets contains 1400 shapes evenly distributed among 70 object classes. Some of the shapes are shown in Figure 6. Samples in the dataset exhibit great intra-class variations including deformation and articulation. We applied the inner distance shape context (IDSC) algorithm [50] to compensate the intra-class variations.

[2] An edge gain can be computed in constant time.

The proposed algorithm requires pairwise similarity scores as inputs. We use a Gaussian kernel given by $w(v_i, v_j) = exp(-\frac{d(v_i,v_j)^2}{2\sigma^2})$ to convert the above distance measures to similarity scores where $d(v_i, v_j)$ is the pairwise distance between samples i and j and σ is the kernel bandwidth. We then construct a neighbor graph where each sample is connected to its 30 nearest neighbors prior to clustering.

In the experiments we set the number of clusters equal to the true number K for all the algorithms. For comparison, we use two standard clustering performance metrics: 1.) clustering accuracy and 2.) Rand index.

- **Clustering accuracy (CA)** is a classification-accuracy like performance metric. Let $\mathcal{C} = \{C_1, C_2, ..., C_K\}$ be the ground truth distributions of clusters where C_i is the set of indices for samples in the ith cluster. Similarly let $\mathcal{S} = \{S_1, S_2, ..., S_K\}$ be the computed cluster distribution with S_i denoting the index set of samples assigned to the ith cluster. The clustering accuracy is given by

$$CA = \max_J \frac{1}{n} \sum_i |C_i \cap S_{J(i)}| \tag{21}$$

  where n is the total number of samples in the dataset and J represents any possible permutation of the sequence $\{1, 2, ..., K\}$. Equation (21) requires searching for the best permutation which is solved using the Hungarian algorithm.
- **Rand index (RI)** is a measure of similarity between two clusterings: the ground truth and estimated. Let TP be the number of sample pairs that are in the same cluster for both ground truth and estimated clusterings, TN be the number of sample pairs that are in different clusters for the ground truth and estimated clusterings, FP be the number of sample pairs that are in different clusters for the ground truth clustering but are in the same cluster for the estimated clustering, and FN be the number of sample pairs that are in the same cluster for the ground truth clustering but are in different clusters for the estimated clustering output. In other words, TP, TN, FP, and FN correspond to the counts of true positive, true negative, false positive, and false negative sample pairs respectively. The Rand index is given by percentage of agreed cluster assignment

$$RI = \frac{TP + TN}{TP + TN + FP + FN}. \tag{22}$$

We compare our results with state-of-the-art clustering algorithms including AP [51], K-means, NCut [17], and the cutting plane maximum margin clustering algorithm (CPMMC) [52]. They represent a variety of clustering methods from example-based, centroid-based, graph-theoretic, to maximum margin-based methods. For AP the number of clusters is implicitly controlled by the preference parameter; a binary search on the parameter value is performed for obtaining the output with the desired number of clusters. We used the implementation available from the author's website. The K-means algorithm is sensitive to initialization. We initialized the K-means algorithm with 100 different configurations using the implementation available in MATLAB. Both the NCut algorithm and the proposed algorithm have a kernel bandwidth parameter. Following the setup in [52], we exhaustively searched a range of the parameter values and report the best performance obtained for each of the algorithm. Specifically, we computed the minimum distance and the maximum distance for all the sample pairs prior to clustering. The kernel bandwidth values were then varied from 20% of the minimum distance to the maximum distance linearly in 240 steps. We used the implementation of NCut available provided in [17]. The performance numbers of the CPMMC algorithm were duplicated from a recent paper [52]. The results for clustering accuracy and rand index are shown in Table 1 and Table 2 respectively.

TABLE 1
Clustering performance comparison: clustering accuracy.

Dataset	Proposed	NCut	AP	K-means	CPMMC
Ionosphere	**92.59**	83.19	70.94	70.00	75.48
Letters	94.45	94.28	91.83	93.38	**95.02**
Satellite	**99.51**	97.50	62.30	94.10	98.79
Digits 0689	**98.24**	91.83	90.31	78.46	96.74
Digits 1279	**95.97**	91.70	85.51	89.32	94.52
Breast Cancers	92.97	92.09	**93.32**	91.04	n/a
Iris	**94.00**	86.67	86.00	83.33	n/a
Wine	96.63	**98.31**	93.82	96.63	n/a
Glass	50.93	**55.14**	40.19	45.33	n/a
Movement Libras	**53.06**	50.83	46.94	44.44	n/a
Natural Scenes	47.36	**56.36**	43.64	47.70	n/a
MPEG-7 Shapes	**74.00**	71.64	69.14	n/a	n/a

TABLE 2
Clustering performance comparison: rand index.

Dataset	Proposed	NCut	AP	K-means	CPMMC
Ionosphere	**0.86**	0.72	0.59	0.58	0.65
Letters	0.90	0.89	0.85	0.88	**0.92**
Satellite	**0.99**	0.95	0.53	0.89	0.97
Digits 0689	**0.98**	0.93	0.92	0.87	0.97
Digits 1279	**0.96**	0.92	0.87	0.90	**0.96**
Breast Cancers	0.87	0.85	**0.88**	0.84	n/a
Iris	**0.93**	0.86	0.85	0.83	n/a
Wine	0.96	**0.98**	0.92	0.95	n/a
Glass	**0.73**	0.70	0.66	0.70	n/a
Movement Libras	**0.92**	**0.92**	0.91	0.91	n/a
Natural Scenes	0.82	**0.84**	0.81	0.83	n/a
MPEG-7 Shapes	**0.99**	**0.99**	**0.99**	n/a	n/a

TABLE 3
Clustering performance comparison: performance rank averages in clustering accuracy and rand index.

Algorithm	Proposed	NCut	AP	K-means	CPMMC
CA	**1.5**	2.2	3.8	3.7	2.0
RI	**1.4**	2.1	3.6	3.6	1.8

From Tables 1 and 2, we see that the proposed algorithm produces slightly better performances in clustering. It outperforms the competing algorithms in 7 out of the 12 datasets according to the clustering accuracy measure. We also achieve better performance according to Rand index: better in 8 out of the 12 datasets. For the two challenging vision datasets, all the algorithms did not perform well. This is mainly due to the insufficiency of the descriptors in modeling the intra-class and inter-class variations of the datasets. We

Fig. 5. Example images from the natural scene recognition dataset [48]. From left to right, the image classes are coast, forest, highway, inside city, mountain, open country, street, and tall building. The images of the same class exhibit great variation due to different imaging conditions such as locations and seasons.

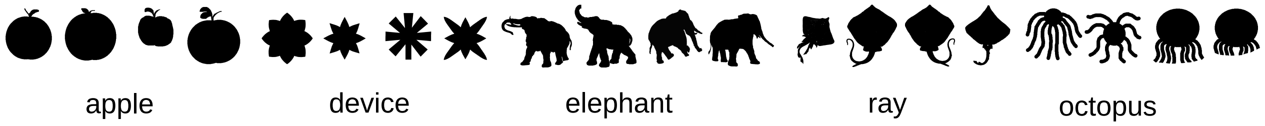
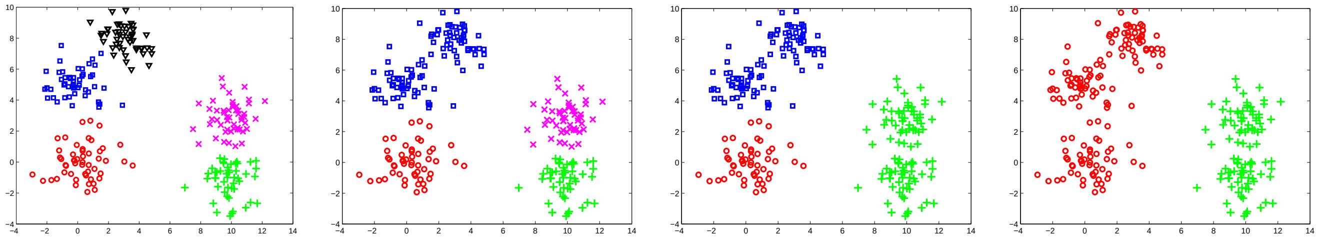


Fig. 6. Example silhouettes from the MPEG-7 shape dataset [49]. The dataset contains 70 different shape classes and each class have 20 instances in various deformation and articulation. We show four instances for each of the apple, device, elephant, ray, and octopus classes.



(a) (b) (c) (d)

Fig. 7. We show the intermediate results of dichotomizing a dataset consisting of 5 Gaussian clouds. After the first few iterations, we recover the 5 Gaussian clouds in different clusters in (a). The subsequent combinations results in 4, 3, and 2 clusters as shown in (b), (c), and (d) respectively.

obtained a better clustering accuracy for the MPEG-7 shape dataset while our results are inferior to NCut in the natural scene clustering task. We summarize the performances using their average performance ranks in Table 3. The proposed algorithm has an average performance rank of 1.5 and 1.4 for clustering accuracy and Rand index, which outperforms the other algorithms.

The proposed algorithm can be viewed as an agglomerative clustering algorithm, which iteratively groups samples to form a hierarchical structure. In the next experiment, we demonstrate this agglomerative behavior in dichotomizing a dataset consisting of 5 Gaussian clouds as shown Figure 7. One can see that it first discovers the 5 Gaussian clouds in Figure 7(a) and subsequently combines proximate ones until the number of remaining clusters equal to two as shown in Figure 7(b)(c)(d). The agglomerative property is useful for identifying the internal structure of dataset and scientific visualization.

TABLE 4
Comparison to agglomerative clustering algorithms.

Dataset	Proposed	Single	Complete	Average
Ionosphere	**92.59**	64.39	67.24	64.39
Letters	94.45	**94.47**	61.86	94.02
Satellite	**99.51**	68.60	92.62	94.45
Digits 0689	**98.24**	25.27	25.27	25.27
Digits 1279	**95.97**	25.49	25.44	25.44
Breast Cancers	**92.97**	63.09	63.09	63.27
Iris	**94.00**	66.00	78.67	68.67
Wine	**96.63**	37.64	83.71	38.76
Glass	**50.93**	36.45	40.65	37.85
Movement Libras	**53.06**	10.83	43.61	39.17

We further compare the proposed algorithm to other agglomerative clustering algorithms including single-linkage, complete-linkage, and average-linkage methods on the UCI datasets. Specifically, we first construct the hierarchical clustering tree using these methods with the pairwise dissimilarity given by the Euclidean distance. We then find the horizontal cut through the tree that gives a desired number of clusters for output. The performance comparison is shown in Table 4. One can see that the proposed algorithm consistently outperforms the other agglomerative clustering methods. This is because in addition to the compactness criterion common in agglomerative clustering the proposed algorithm also encourages the formation of homogeneous and balanced clusters.

5.2 Superpixel Segmentation

We conducted experiments for superpixel segmentation using the Berkeley segmentation benchmark [53]. The benchmark contains 300 grey images with human-labeled ground truths. In order to compute the pairwise similarity between neighboring pixels, we adopt the function

$$\exp(-\frac{(\|p-q\|_2|I(p)-I(q)|)^2}{2\sigma^2}) \tag{23}$$

where p and q are pixel coordinates, $\|p-q\|_2$ is their L_2 distance, and $|I(p)-I(q)|$ is the absolute value of their intensity difference. The kernel bandwidth is set to $\sigma = 5.0$ throughout the superpixel segmentation experiments.

Superpixel segmentation has a different goal than object segmentation, and therefore the performance metrics are also different. We computed three standard metrics which are commonly used for evaluating the quality of superpixels: undersegmentation error [35][36], boundary recall [6] and achievable segmentation accuracy [54]. For the sake of completeness we first describe these metrics. We use $\mathcal{G} = \{G_1, G_2, ..., G_{n_\mathcal{G}}\}$ to represent a ground truth segmentation with n_G segments and $|G_i|$ denotes the segment size.

- **Undersegmentation error (UE)** measures the fraction of pixel leak across ground truth boundaries. It evaluates the quality of segmentation based on the requirement that a superpixel should overlap with only one object. We utilize the undersegmentation error metric used in Veksler et al. [36],

$$UE_\mathcal{G}(\mathcal{S}) = \frac{\sum_i \sum_{k:S_k \cap G_i \neq \emptyset} |S_k - G_i|}{\sum_i |G_i|}. \tag{24}$$

  For each ground truth segment G_i we find the overlapping superpixels S_k's and compute the size of the pixel leaks $|S_k - G_i|$'s. We then sum the pixel leaks over all the segments and normalize it by the image size $\sum_i |G_i|$.

- **Boundary recall (BR)** measures the percentage of the natural boundaries recovered by the superpixel boundaries. We compute BR using

$$BR_\mathcal{G}(\mathcal{S}) = \frac{\sum_{p\in\delta\mathcal{G}} \mathbb{I}(\min_{q\in\delta\mathcal{S}}\|p-q\| < \epsilon)}{|\delta\mathcal{G}|}, \tag{25}$$

  which is the ratio of ground truth boundaries that have a nearest superpixel boundary within an ϵ-pixel distance. We use $\delta\mathcal{S}$ and $\delta\mathcal{G}$ to denote the union sets of superpixel boundaries and ground truth boundaries respectively. The indicator function $\mathbb{I}$ checks if the nearest pixel is within ϵ distance. In our experiments we set $\epsilon = 2$.

- **Achievable segmentation accuracy (ASA)** is a performance upperbound measure. It gives the highest accuracy achievable for object segmentation that utilizes superpixels as units. To compute ASA we label each superpixel with the label of the ground truth segment that has the largest overlap. The fraction of correctly labeled pixels is the achievable accuracy,

$$ASA_\mathcal{G}(\mathcal{S}) = \frac{\sum_k \max_i |S_k \cap G_i|}{\sum_i |G_i|}. \tag{26}$$

These performance metrics are plotted against the number of superpixels in an image. Algorithms producing better performances with a smaller number of superpixels are more preferable.

In the first experiment, we compared our results with FH [32], GraphCut superpixel [36], Turbopixels [35] and NCut superpixel [6] methods using the three evaluation metrics. The results were obtained by averaging over all the 300 gray images in the dataset.

Figure 8(a) shows the undersegmentation error curves. The curves for the other methods are duplicated from the original paper [36]. The proposed algorithm outperforms the state-of-the-art at all the superpixel counts where the error rate is reduced by more than 50%. It achieves an undersegmentation error of 0.13 with 350 superpixels while the same performance is achieved with 550 superpixels using GraphCut superpixel segmentation [36]. With 550 superpixels, our undersegmentation error is 0.06.

In Figure 8(b), we plot the boundary recall curves. Again, the curves for the other methods are duplicated from the original paper [36]. The proposed algorithm reduces the missed boundaries by more than 30% compared to the state-of-the-art at all the superpixel counts. The recall rates of the presented algorithm are 82% and 92% with 200 and 600 superpixels respectively. The recall rates with the same superpixel counts are 76 and 86 percents with FH.

In Figure 8(c), we plot the achievable segmentation accuracy curves. In this experiment we generated the curves for the other methods using the original implementations available online. The proposed algorithm yields a better achievable segmentation upper-bound at all the superpixel counts—particularly for smaller number of superpixels. The ASA is 95% with 100 superpixels where the same accuracy can only be achieved with 200 superpixels for the other algorithms.

In the second experiment, we evaluated the segmentation results visually. Several examples are shown in Figure 9 where the images are partitioned into 100 superpixels. For better visualization, the ground truth segments are color-coded and blended on the images, and the superpixel boundaries recovered by the algorithm are superimposed in white color. It is difficult to notice pixel leaks and the superpixels tend to divide an image into similar-sized regions which are important for region based feature descriptors.

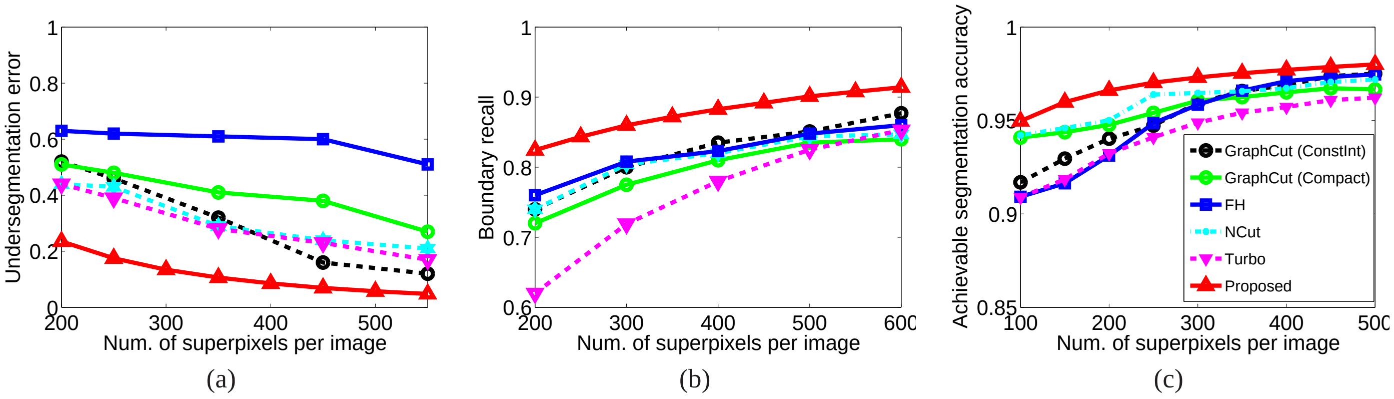


Fig. 8. Performance metrics: (a) undersegmentation error, (b) boundary recall, and (c) achievable segmentation accuracy curves. The proposed algorithm performs significantly better in all the metrics at all the superpixel counts.

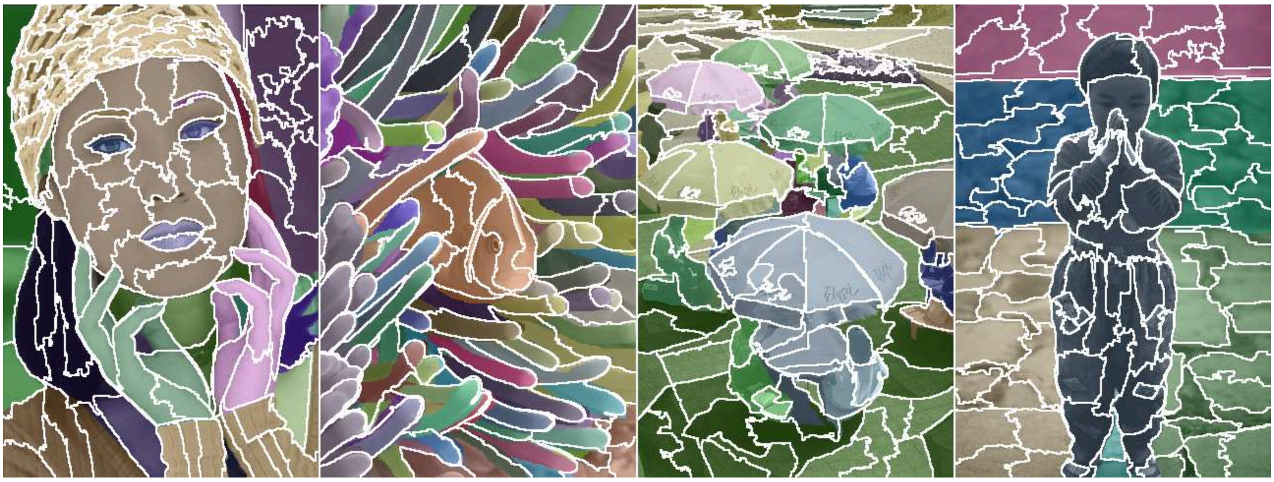


Fig. 9. Superpixel segmentation examples. The images contain 100 superpixels. The ground truth segments are color-coded and blended on the images. The superpixels (boundaries shown in white) respect object boundaries and tend to divide an image into similar-sized regions.

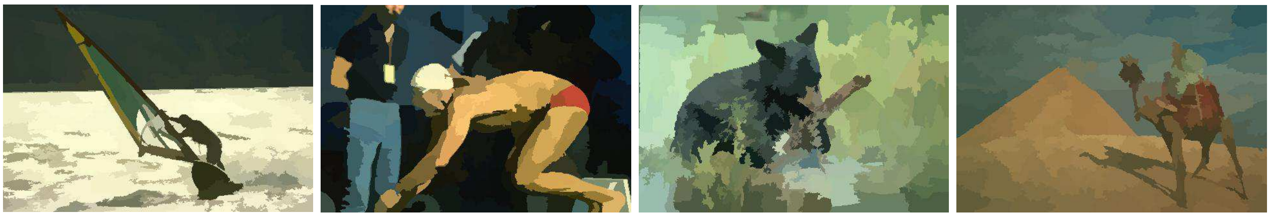


Fig. 10. Nonphotorealistic rendering using superpixels. The images are divided into 150 superpixels and each pixel is colored by the average color of the superpixel it belongs to. The balanced-size objective renders an artistic effect capturing the style of thick application of paintbrush common in post-impressionism.

In the third experiment, we evaluated the effectiveness of the proposed algorithm for nonphotorealistic rendering. Several examples are shown in Figure 10 which are computed by first dividing the images into 150 superpixels and coloring each pixel by the average color of the superpixel it belongs to. Though one might argue that similar effects can be achieved by other image smoothing techniques, the proposed algorithm renders similar-sized segments and the smoothing effect captures the style of thick application of paintbrush — a style popular in post-impressionism.

In Figure 11 we plot the distributions on superpixel size. We applied the proposed algorithm to segment the benchmark images with different numbers of superpixel counts, namely 200, 400, and 600. The superpixels computed using the same count are pooled to obtain the size distribution for the count. One can see that these distributions, though of different counts, all have a similar bell shape. Most of the superpixel sizes are close to the average size. The superpixels with very small spatial supports or very large spatial support are rare.

In the last experiment, we analyze the effects of the bal-

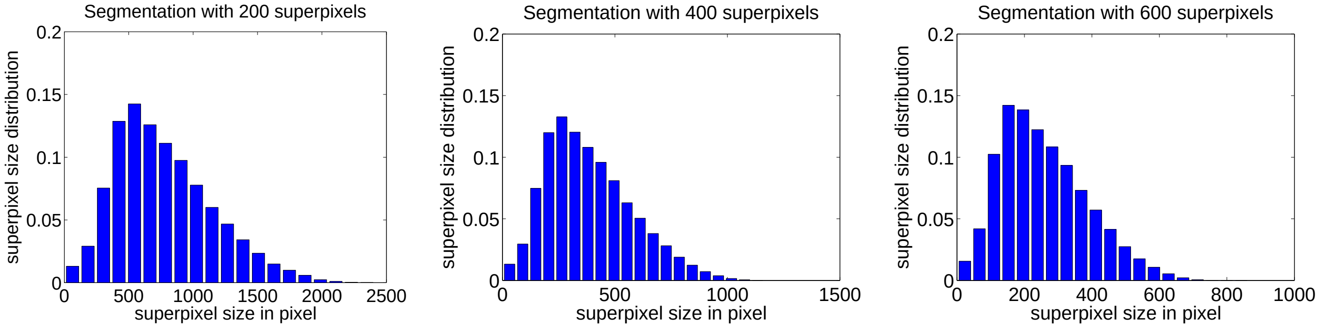


Fig. 11. Superpixel size distribution. We plot the distributions on superpixel sizes obtained by segmenting the image into (a) 200 superpixels (b) 400 superpixels and (c) 600 superpixels. Each of the distributions has a bell shape. The proposed algorithm divides the images into similar-sized regions and avoids producing superpixels with very small or large spatial support.

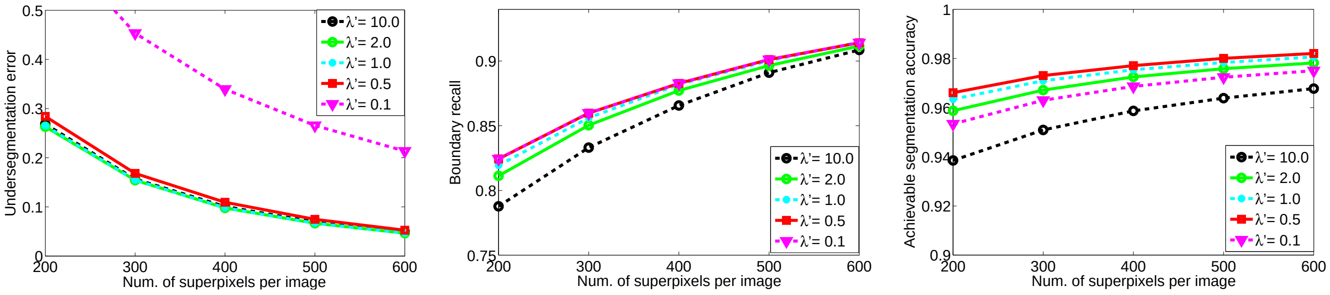


Fig. 12. Effect of the balancing preference on the performance metrics.

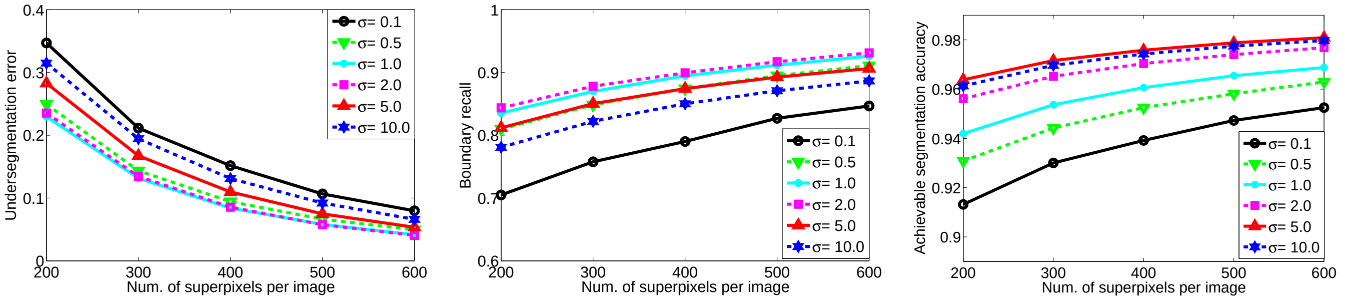


Fig. 13. Effect of the kernel bandwidth on the performance metrics.

ancing term, λ', and the kernel bandwidth, σ, parameters on the quality of segmentation. We observe that competitive segmentation results are achieved with a wide range of parameter selection.

In Figure 12, we plot the performance curves for a range of λ' values for a fixed $\sigma = 5.0$. We observed that smaller λ' results in better boundary recall rates especially for smaller superpixel counts, while the results are largely invariant to this parameter for larger superpixel counts. We further observed that better performances on undersegmentation error and achievable segmentation accuracy are achieved with a larger λ'. In general, there is a tradeoff among different metrics based on the λ' parameter, and empirically we found that $\lambda' = 0.5$ yields a good compromise among these metrics.

In Figure 13, we plot the performance curves for a range of σ values for a fixed $\lambda' = 0.5$. We observed that a large range of σ values results in comparable performances, namely from 0.5 to 5. The superpixels are largely insensitive to the selection of the σ parameter.

The proposed algorithm is among the fastest superpixel segmentation algorithms and takes an average of 2.5 seconds to segment an image on the Berkeley benchmark (481×321 pixels) on an Intel Core 2 Duo E8400 processor. Compared to the state-of-the-art methods, it is faster than the Graphcut superpixel [36] (6.4 seconds), turbopixel [35](15 seconds), and NCut (5 minutes), whereas it is slower than FH [32](0.5 seconds).

6 Summary

We presented a novel objective function for cluster analysis. It is a combination of the entropy rate of a random walk on the data graph and a balancing criterion. The property of this objective function and its optimization were analyzed. We showed that, by exploiting its submodularity and a matroid structure, a simple greedy algorithm can efficiently compute a performance-guaranteed solution. We have achieved significant performance improvements for superpixel segmentation tasks and competitive results compared to the state-of-the-art clustering algorithms on standard datasets.

We plan to explore user-specified constraints in the clustering problem. Another interesting research direction is to study hyper-graph clustering where edges are defined over a set of vertices.

Acknowledgement: We thank David W. Jacobs, Jay Thornton, Yuichi Taguchi, Yan Chen, and K.J. Ray Liu for their valuable discussions and suggestions. The research was conducted at MERL with support from MERL. Prof. Chellappa was supported by MURI from the Office of Naval Research under the Grant N00014-10-1-0934.

References

[1] E. T. Jaynes, "Information theory and statistical mechanics," *Physical Review*, vol. 106, no. 4, pp. 620–630, 1957.

[2] L. Lovász, "Submodular functions and convexity," *Mathematical Programming - State of the Art*, pp. 235–257, 1983.

[3] C. Guestrin, A. Krause, and A. P. Singh, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, pp. 235–284, 2008.

[4] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *The ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2007, pp. 420–429.

[5] H. Lin and J. Bilmes, "Word alignment via submodular maximization over matroids," in *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Short Papers*, 2011, pp. 170–175.

[6] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proceeding of IEEE International Conference on Computer Vision*, 2003.

[7] G. Mori, X. Ren, A. A. Efros, and J. Malik, "Recovering human body configurations: Combining segmentation and recognition," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[8] P. Kohli, L. Ladicky, and P. Torr, "Robust higher order potentials for enforcing label consistency," *International Journal on Computer Vision*, vol. 82, pp. 302–324, 2009.

[9] D. Hoiem, A. N. Stein, A. A. Efros, and M. Hebert, "Recovering occlusion boundaries from a single image," in *Proceeding of IEEE International Conference on Computer Vision*, 2007.

[10] S. Ramalingam, P. Kohli, K. Alahari, and P. H. S. Torr, "Exact inference in multi-label crfs with higher order cliques," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[11] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[12] P. Berkhin, "Survey of clustering data mining techniques," Tech. Rep., 2002.

[13] R. Xu and I. Wunsch, D., "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on*, vol. 16, no. 3, pp. 645 –678, May 2005.

[14] M. Filippone, F. Camastra, F. Masulli, and S. Rovetta, "A survey of kernel and spectral methods for clustering," *Pattern Recogn.*, vol. 41, pp. 176–190, January 2008.

[15] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters," *IEEE Transactions on Computers*, vol. 20, no. 1, pp. 68–86, 1971.

[16] 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*, vol. 15, no. 11, pp. 1101–1113, 1993.

[17] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[18] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Machine Learning*, vol. 56, no. 1-3, pp. 89–113, 2004.

[19] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *The Neural Information Processing Systems (NIPS) Foundation*. MIT Press, 2001, pp. 849–856.

[20] D. Yan, L. Huang, and M. I. Jordan, "Fast approximate spectral clustering," in *The ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2009, pp. 907–916.

[21] R. Krauthgamer, J. Naor, and R. Schwartz, "Partitioning graphs in to balanced components," in *In Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, 2009.

[22] M. Meilă and J. Shi, "A random walks view of spectral segmentation," in *IEEE Interntaional Conference on Artificial Intelligence and Statistics*, 2001.

[23] D. Harel and Y. Koren, "On clustering using random walks," in *Foundations of Software Technology and Theoretical Computer Science*, vol. 2245. Springer-Verlag, 2001, pp. 18–41.

[24] L. Yen, D. Vanvyve, F. Wouters, F. Fouss, M. Verleysen, and M. Saerens, "Clustering using a random-walk based distance measure," in *In Proceedings European Symposium on Artificial Neural Networks*, 2005.

[25] L. Grady, "Random walks for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1768–1783, 2006.

[26] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with bregman divergences," *Journal of Machine Learning Research*, vol. 6, pp. 1705–1749, 2005.

[27] A. C. Müller, S. Nowozin, and C. H. Lampert, "Information theoretic clustering using minimum spanning trees," in *DAGM/OAGM Symposium*, 2012, pp. 205–215.

[28] M. Narasimhan, N. Jojic, and J. Bilmes, "Q-clustering," in *The Neural Information Processing Systems (NIPS) Foundation*. MIT Press, 2006, pp. 979–986.

[29] K. Nagano, Y. Kawahara, and S. Iwata, "Minimum average cost clustering," in *The Neural Information Processing Systems (NIPS) Foundation*, 2010.

[30] S. Jegelka and J. Bilmes, "Submodularity beyond submodular energies: Coupling edges in graph cuts," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[31] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.

[32] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal on Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[33] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[34] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 583 – 598, 1991.

[35] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, and S. J. Dickinson, "Fast superpixels using geometric flows," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2290–2297, 2009.

[36] O. Veksler and Y. Boykov, "Superpixels and supervoxels in an energy optimization framework," in *Proceeding of European Conference on Computer Vision*, 2010.

[37] Y. Boykov, O. Veksler, and R. Zabih, "Efficient approximate energy minimization via graph cut," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1222–1239, 2001.

[38] A. P. Moore, S. J. D. Prince, J. Warrell, U. Mohammed, and G. Jones, "Superpixel lattices," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[39] A. P. Moore, S. J. D. Prince, and J. Warrell, ""lattice cut" – constructing superpixels using layer constraints," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[40] Y. Taguchi, B. Wilburn, and C. L. Zitnick, "Stereo reconstruction with mixed pixels using adaptive over-segmentation," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[41] M. Bleyer, C. Rother, P. Kohli, D. Scharstein, and S. Sinha, "Object stereo — joint stereo matching and object segmentation," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[42] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[43] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. John Wiley, 1991.

[44] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of the approximations for maximizing submodular set functions," *Mathematical Programming*, pp. 265–294, 1978.

[45] J. Oxley, *Matroid Theory*. Oxford University Press, 1992.

[46] M. Á. Carreira-Perpiñán and R. S. Zemel, "Proximity graphs for clustering and manifold learning," in *The Neural Information Processing Systems (NIPS) Foundation*. MIT Press, 2004, pp. 225–232.

[47] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of the approximations for maximizing submodular set functions - ii," *Mathematical Programming*, pp. 73–87, 1978.

[48] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal on Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[49] S. Jeannin and M. Bober, "Description of core experiments for mpeg-7 motion/shape," *Technical Report ISO/IEC JTC 1/SC29/WG 11 MPEG99/N2690, MPEG-7*, 1999.

[50] H. Ling and D. W. Jacobs, "Shape classification using the inner-distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 286–299, 2007.

[51] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[52] F. Wang, B. Zhao, and C. Zhang, "Linear time maximum margin clustering," *IEEE Transactions on Neural Networks*, vol. 21, no. 2, pp. 319–332, 2010.

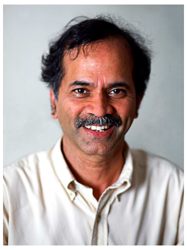
[53] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceeding of IEEE International Conference on Computer Vision*, 2001.

[54] S. Nowozin, P. Gehler, and C. Lampert, "On parameter learning in crf-based approaches to object class image segmentation," in *Proceeding of European Conference on Computer Vision*, 2010.

Ming-Yu Liu Ming-Yu Liu received the B.E. degree from the National Chiao Tung University, Taiwan, in 2003 and the M.S. and Ph.D. degrees in Electrical Engineering from the University of Maryland College Park, Maryland, USA, in 2010 and 2012 respectively. He is a research staff at Mitsubishi Electric Research Labs (MERL), Cambridge, Massachusetts, USA. His research interests are in computer vision, machine learning, and robotics. He is a member of the IEEE.

Oncel Tuzel Oncel Tuzel received the BS and the MS degrees in computer engineering from the Middle East Technical University, Ankara, Turkey, in 1999 and 2002, and PhD degree in computer science from the Rutgers University, Piscataway, New Jersey, in 2008. He is a principal member of research staff at Mitsubishi Electric Research Labs (MERL), Cambridge, Massachusetts. His research interests are in computer vision, machine learning, and robotics. He coauthored more than 30 technical publications, and has applied for more than 20 patents. He was on the program committee of several international conferences such as CVPR, ICCV, and ECCV. He received the best paper runner-up award at the IEEE Computer Vision and Pattern Recognition Conference in 2007. He is a member of the IEEE.

Srikumar Ramalingam Srikumar Ramalingam is a Principal Research Scientist at Mitsubishi Electric Research Lab (MERL). He received his B.E from Anna University in India and his M.S from University of California (Santa Cruz) in USA. He received a Marie Curie Fellowship from European Union to pursue his studies at INRIA Rhone Alpes (France) and he obtained his PhD in 2007. His thesis on generic imaging models received INPG best thesis prize and AFRIF thesis prize (honorable mention) from the French Association for Pattern Recognition. He has published more than 30 papers in flagship conferences such as CVPR, ICCV and ECCV. He has coauthored books, given tutorials and organized workshops on topics such as multi-view geometry and discrete optimization. His research interests are in computer vision, machine learning and robotics problems.



Rama Chellappa Prof. Rama Chellappa received the B.E. (Hons.) degree from the University of Madras, India and the M.E. degree with Distinction from the Indian Institute of Science, Bangalore in 1975 and 1977 respectively. He received the M.S.E.E. and Ph.D. Degrees in Electrical Engineering from Purdue University, West Lafayette, IN in 1978 and 1981 respectively. During 1981-1991, he was a faculty member in the department of EE-Systems at University of Southern California (USC). Since 1991, he has been a Professor of Electrical and Computer Engineering (ECE) and an affiliate Professor of Computer Science at University of Maryland (UMD), College Park. He is also affiliated with the Center for Automation Research and the Institute for Advanced Computer Studies (Permanent Member) and is serving as the Chair of the ECE department. In 2005, he was named a Minta Martin Professor of Engineering. His current research interests span many areas in image processing, computer vision and pattern recognition. Prof. Chellappa has received several awards including an NSF Presidential Young Investigator Award and four IBM Faculty Development Awards. He received the K.S. Fu Prize and two paper awards from the International Association of Pattern Recognition (IAPR). He is a recipient of the Society, Technical Achievement and Meritorious Service Awards from the IEEE Signal Processing Society. He also received the Technical Achievement and Meritorious Service Awards from the IEEE Computer Society. He is a recipient of the Excellence in teaching award from the School of Engineering at USC. At UMD, he received college and university level recognitions for research, teaching, innovation and mentoring undergraduate students. In 2010, he was recognized as an Outstanding ECE by Purdue University. Prof. Chellappa served as the Editor-in-Chief of IEEE Transactions on Pattern Analysis and Machine Intelligence and as the General and Technical Program Chair/Co-Chair for several IEEE international and national conferences and workshops. He is a Golden Core Member of the IEEE Computer Society, served as a Distinguished Lecturer of the IEEE Signal Processing Society and as the President of IEEE Biometrics Council. He is a Fellow of IEEE, IAPR, OSA and AAAS and holds four patents