# Normalized Tree Partitioning for Image Segmentation

Jingdong Wang[1] Yangqing Jia[2] Xian-Sheng Hua[1] Changshui Zhang[2] Long Quan [3]

[1]Microsoft Research Asia Beijing, P. R. China
[2] Tsinghua University Beijing, P. R. China
[3] Hong Kong University of Science and Technology Hong Kong, P. R. China

## Abstract

*In this paper, we propose a novel graph based clustering approach with satisfactory clustering performance and low computational cost. It consists of two main steps: tree fitting and partitioning. We first introduce a probabilistic method to fit a tree to a data graph under the sense of minimum entropy. Then, we propose a novel tree partitioning method under a normalized cut criterion, called Normalized Tree Partitioning (NTP), in which a fast combinatorial algorithm is designed for exact bipartitioning. Moreover, we extend it to $k$-way tree partitioning by proposing an efficient best-first recursive bipartitioning scheme. Compared with spectral clustering, NTP produces the exact global optimal bipartition, introduces fewer approximations for $k$-way partitioning and can intrinsically produce superior performance. Compared with bottom-up aggregation methods, NTP adopts a global criterion and hence performs better. Last, experimental results on image segmentation demonstrate that our approach is more powerful compared with existing graph-based approaches.*

## 1. Introduction

Perceptual grouping is always a fundamental but challenging problem in pattern recognition and computer vision. Many real-world applications can benefit from satisfactory grouping. For example, in pattern recognition and multimedia analysis, image categorization and annotation usually adopt image segmentation results to extract features for matching. In managing/summarizing image search results from image search engines, grouping visually similar images has been known as an effective way. Computational efficiency and result reliability are two important factors for evaluating clustering methods. In this paper, we propose a novel clustering method, called Normalized Tree Partitioning (NTP), to improve clustering efficiency and reliability, and demonstrate its powerfulness on image segmentation.

There is a large literature on clustering, dating back over 30 years [8]. Generally, clustering methods can be divided into three categories: (1) clustering based on probability density, such as K-means, Gaussian mixture models and mean shift [4]; (2) clustering based on pairwise graph models, including spectral clustering [11], affinity propagation [7], incremental aggregation [5] and so on; (3) clustering using multiple-wise relationship, such as [1, 17]. The first category views each data point generated from one latent probability distribution and clusters data points to a local mode or component, but takes no consideration of spatial or local consistency and suffers from heavy computation load when dealing with large scale data. The second category usually constructs a weighted graph with the nodes representing the data points, the edges connecting pairs of points and the weights on the edges reflecting the similarities between the corresponding points, and then partitions the graph to finally cluster the data points. The last one models multiple-wise relations among data points, but brings expensive computational cost, making it impractical. Here we briefly review the second category that is very relevant to the proposed approach.

There are basically two types of pairwise graph based methods. The first aims to define a global criterion and design a top-down optimization algorithm, expecting to obtain a reliable clustering result. The normalized cuts (Ncuts) criterion, first proposed in [11], is thought as an ideal one. It aims to minimize the summation of the weights of the edges that are being split and meanwhile maximize the summation of the weights on each cluster. This criterion results in an NP hard problem if there exist cycles in the graph, and relaxation techniques are introduced to approximately solve it. One popular method, based on the spectral relaxation technique, is spectral clustering proposed in [11, 14]. Spectral clustering suffers from heavy computational load, and the results may be unsatisfactory due to unpredictable approximation though it adopts a good criterion.

The other type adopts a simple local criterion and performs a bottom-up strategy to heuristically aggregate the data points into more and more compact clusters. One of the representative methods is presented in [5]. This method incrementally unions two small clusters (initially a cluster

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

only consists of a single data point) into one cluster, based on the weights of the edges connecting the two clusters. This method is computationally efficient, but may not get satisfactory segmentation results due to the simple union criterion, which only considers the weights of the edges connecting two clusters. Graph theoretic methods proposed in [13, 15] find a minimum spanning tree from the data graph and cut the edges corresponding to minimum weights. It also suffers from the local criterion and results in unsatisfactory performance.

### 1.1. Our approach

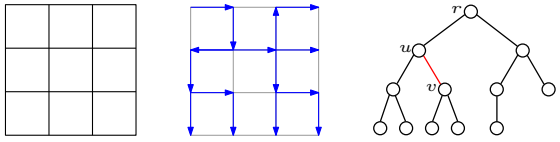
An ideal clustering method usually has two characteristics: satisfactory clustering performance and computational efficiency, which require a good criterion to measure the clustering result and an effective and efficient solution to this criterion. Few existing approaches meet such two requirements. Bottom-up aggregation methods, such as [5], is computationally efficient while its criterion only considers local similarities, i.e., minimizing inter-similarities between clusters, hence has no ability to evaluate the self-similarity in a single cluster. Ncuts is a criterion to measure both inter-similarities and self-similarities, but existing solutions approximately optimize it using numerical algorithms such as spectral clustering [11], which brings unpredictable approximations and leads to unsatisfactory results.

In this paper, we propose a novel clustering approach to optimize the Ncuts criterion, to reduce computational cost and boost clustering satisfaction. To solve the difficulty in optimizing the Ncuts criterion on a cyclic graph, we perform a tree fitting step by breaking the cycles, and then partition this tree under the Ncuts criterion. The two steps are specified as follows:

1. Construct a tree to probabilistically approximate a data graph under the minimum entropy criterion, presented in Sec. 2.
2. Perform the proposed normalized tree partitioning method, which provides an effective solution to Ncuts on a tree to obtain the clustering, detailed in Sec. 3.

The main contributions are summarized in the following several perspectives:

1. An effective way is introduced to probabilistically approximate the data graph using a tree under the minimum entropy criterion.
2. An efficient algorithm is proposed to optimize Ncuts on a tree, called Normalized Tree Partitioning. Moreover, theoretic analysis is given to prove that the proposed algorithm is guaranteed to get an exact global optimum for normalized tree bipartitioning.




(a) Data graph (b) Candidate tree (c) Tree structure

Figure 1. (b) shows a candidate tree made up of directed edges to approximate the graph in (a). In (c), red edge $(u, v)$ is a candidate edge to be split.

3. A best-first recursive biparitioning scheme is proposed for $k$-way partitioning. The whole clustering algorithm is computationally efficient.

The remainder is organized as follows. Sec. 2 gives a probabilistic method to fit a tree to a data graph under the sense of minimum entropy. Then, Sec. 3 introduces the normalized tree partitioning method. Next, Sec. 4 presents comparison analysis with related methods. Experimental results in Sec. 5 demonstrate the effectiveness and efficiency of our approach. Finally, Sec. 6 concludes this paper.

## 2. Probabilistic tree fitting

Given a set of $n$ data points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$, we represent the data points in the form of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of $n$ nodes $\{i\}_{i=1}^n$ with node $i$ corresponding to $\mathbf{x}_i$, and $\mathcal{E}$ is a set of edges connecting the data points. Labels $L = \{l_i\}_{i=1}^n$ represent the cluster assignment of the data points. Suppose a distribution $P(L)$ corresponds to a graph $\mathcal{G}$, we aim to find a tree distribution $T(L)$ to fit $P(L)$. Here, $T(L) = \prod_{v\in\mathcal{V}} T_{v|pa(v)}(l_v|l_{pa(v)})$ corresponds to a tree structure $\mathcal{T} = \{(v, pa(v))\}_v$, where $pa(v)$ is the parent node of $v$, and $T_{v|pa(v)}(l_v|l_{pa(v)})$ is a conditional probability. For example, we want to find a tree $\mathcal{T}$ in Fig. 1(b) to approximate the graph in Fig. 1(a).

### 2.1. Background

We follow the idea in [3] to find a tree $\mathcal{T}$ in the sense of maximum likelihood, which is equivalent to minimizing the Kullback-Leibler divergence between $P$ and $T$.

The tree fitting process consists of searching both the best tree structure $\mathcal{T}$ and parameters $\{T_{v|pa(v)}\}_v$. With the deduction in [3], the tree structure $\mathcal{T}$ can be found by maximizing the summation of the mutual information:

$$J(\mathcal{T}) = \sum\nolimits_{(u,v)\in\mathcal{T}} I_{uv}, \tag{1}$$

where the mutual information $I_{uv}$ is defined as

$$I_{uv} = \int P_{uv}(l_u, l_v) \log \frac{P_{uv}(l_u, l_v)}{P_u(l_u)P_v(l_v)} \,\mathrm{d}l_u \,\mathrm{d}l_v.$$

The best tree parameters $T_{v|pa(v)}$ are obtained by copying the conditional distributions $P_{v|pa(v)}$ from $P(L)$, i.e., $T_{v|pa(v)}(l_v|l_{pa(v)}) = P_{v|pa(v)}(l_v|l_{pa(v)}), \forall v \in \mathcal{V}$.

In [3], the probability $P(l)$ is given by a set of discrete-valued examples, hence mutual information $I_{uv}$ can be approximated in a nonparametric way. However, in general cases, such as $P(L)$ given in a parametric function, $I_{uv}$ is not easily computed. In the following subsection, we introduce an approximation method to compute the mutual information for the graph constructed on the data points $\mathcal{X}$.

### 2.2. Tree fitting for a data graph

We use a first-order Markov random field to describe the data graph: $P(L) = \frac{1}{Z}\prod_{(u,v)\in\mathcal{E}} P(l_u, l_v)$, where $Z$ is the normalization constant, $l_u$ is the label of $\mathbf{x}_u$. $P(l_u, l_v) \propto P(\delta_{[l_u=l_v]})$ is used to measure the probability that $\mathbf{x}_u$ and $\mathbf{x}_v$ belong to the same cluster, and it is defined as a Potts model:

$$P(\delta_{[l_u=l_v]}) = \frac{1}{Z}\begin{cases} 1 & \delta_{[l_u=l_v]} = 1 \\ 1-\exp(-\lambda g(\mathbf{x}_u, \mathbf{x}_v)) & \delta_{[l_u=l_v]} = 0, \end{cases}$$

where $g(\mathbf{x}_u, \mathbf{x}_v)$ is a distance measure between data points $\mathbf{x}_u$ and $\mathbf{x}_v$, $Z$ is the normalization parameter, $\lambda$ is set to 1 by default, and $\delta_{[l_u=l_v]}$ is an indicator function.

As mentioned before, the difficulty of tree fitting is computing mutual information $I_{uv}$ because it requires that the marginal and joint probabilities, $P_u$, $P_v$ and $P_{uv}$, are precomputed, while estimating those probabilities is NP-hard in a cyclic graph. Therefore, we approximate $P_{uv}$ using local prior $P(l_u, l_v)$. Without any bias considered, $P(l_u)$ can be simply approximated as a uniform distribution. Then the mutual information is approximated as

$$\begin{aligned} I_{uv} &\approx \sum\nolimits_{l_u,l_v} P(l_u,l_v)\log\frac{P(l_u,l_v)}{(1/M)(1/M)} \\ &= \sum_{l_u,l_v} P(l_u,l_v)\log P(l_u,l_v) + 2\log M \sum_{l_u,l_v} P(l_u,l_v) \\ &\approx \sum\nolimits_{\delta_{[l_u=l_v]}} P(\delta_{[l_u=l_v]})\log P(\delta_{[l_u=l_v]}) \\ &\quad +2\log M\sum\nolimits_{l_u,l_v} P(l_u,l_v) \\ &= -H_{uv} + 2\log M. \end{aligned} \tag{2}$$

Here $\log M$ is constant for any edge $(u, v)$ with $M$ the number of clusters, hence it can be omitted. We call the spanning tree as the *minimum entropy spanning tree* since the objective is equivalent to minimizing the summation of entropies $H_{uv}$.

The tree fitting procedure is summarized as follows:

**Algorithm 1** Minimum entropy tree spanning

1. Compute entropy $H_{uv}$ for all edges$(u, v)$.
2. Construct a minimum entropy spanning tree $\mathcal{T}$ such that $\sum_{(u,v)} H_{uv}$ is minimized by some maximum weight spanning tree algorithms, such as Prim's or Kruskal's algorithms.
3. Assign tree parameters $T_{v|pa(v)}$ as $P_{v|pa(v)}$.

## 3. Normalized tree partitioning

Given a tree $\mathcal{T} = (\mathcal{V}, \mathcal{E}')$, the bipartition is defined as separating the nodes $\mathcal{V}$ into two disjoint sets, $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A}\cup\mathcal{B} = \mathcal{V}$, $\mathcal{A}\cap\mathcal{B} = \emptyset$, by simply removing edges connecting the two sets. The proposed normalized tree partitioning aims to minimize the Ncuts criterion:

$$Ncuts = \frac{cut(\mathcal{A}, B)}{assoc(\mathcal{A}) + cut(\mathcal{A}, \mathcal{B})} + \frac{cut(\mathcal{B}, \mathcal{A})}{assoc(\mathcal{B}) + cut(\mathcal{B}, \mathcal{A})},$$

where $assoc(\mathcal{A}) = \sum_{u,v\in\mathcal{A}} a(u, v)$ is the association to measure the self-similarity of a cluster, $cut(\mathcal{A}, \mathcal{B}) = \sum_{u\in\mathcal{A},v\in\mathcal{B}} a(u, v)$ is the cut to measure the inter-similarity between clusters, and $a(u, v)$ is the similarity between data points $u$ and $v$.

### 3.1. Computing the optimal bipartition

We present the optimal solution to separate a tree into two partitions under the normalized cuts criterion. We first give a theorem to show that only one edge is required to be removed to obtain the global optimum for the normalized cut criterion, and then present the proposed algorithm.

**Theorem 1.** *On a tree, the global optimum for the normalized cut criterion must correspond to two subtrees: $\mathcal{A}$ and $\mathcal{B}$, which are connected respectively.*

*Proof.* We prove this theorem by contradiction. Suppose there exists an optimum where $m$ ($> 1$) edges, $\{(u_i, v_i)\}_{i=1}^m$, $u_i \in \mathcal{A}$, $v_i \in \mathcal{B}$, are removed, and this leads to $m + 1$ connected subtrees $\{\mathcal{V}_j\}_{j=0}^m$. Then the normalized cut can be written as

$$\begin{aligned} NC &= \frac{\sum_i a(u_i, v_i)}{a_\mathcal{A} + \sum_i a(u_i, v_i)} + \frac{\sum_i a(u_i, v_i)}{a_\mathcal{B} + \sum_i a(u_i, v_i)} \\ &= \frac{a_\mathcal{T}\sum_i a(u_i, v_i)}{(a_\mathcal{A} + \sum_i a(u_i, v_i))(a_\mathcal{B} + \sum_i a(u_i, v_i))}, \end{aligned} \tag{3}$$

where $a_\mathcal{T} \equiv a_\mathcal{A} + \sum_i a(u_i, v_i) + a_\mathcal{B} + \sum_i a(u_i, v_i) \equiv assoc(\mathcal{A}\cup\mathcal{B}, \mathcal{A}\cup\mathcal{B}) \equiv assoc(\mathcal{V}, \mathcal{V})$ by definition. Consider the $m$ possible partitions, $\{(\mathcal{A}_i, \mathcal{B}_i)\}_{i=1}^m$, corresponding to splitting one single edge from $\{(u_i, v_i)\}_{i=1}^m$. The normalized cut of $(\mathcal{A}_i, \mathcal{B}_i)$ is written as

$$\begin{aligned} NC_i &= \frac{a(u_i, v_i)}{a_{\mathcal{A}_i} + a(u_i, v_i)} + \frac{a(u_i, v_i)}{a_{\mathcal{B}_i} + a(u_i, v_i)} \\ &= \frac{a_\mathcal{T} a(u_i, v_i)}{(a_{\mathcal{A}_i} + a(u_i, v_i))(a_{\mathcal{B}_i} + a(u_i, v_i))}. \end{aligned} \tag{4}$$

Denote $\overline{NC}_i = \frac{NC_i}{a_\mathcal{T}}$. The following inequality holds

$$\min(\{\overline{NC}_i\}_{i=1}^m) \tag{5}$$

$$\leq \frac{\sum_i a(u_i, v_i)}{\sum_i (a_{\mathcal{A}_i} + a(u_i, v_i))(a_{\mathcal{B}_i} + a(u_i, v_i))} \tag{6}$$

$$< \frac{\sum_i a(u_i, v_i)}{(a_\mathcal{A} + \sum_i a(u_i, v_i))(a_\mathcal{B} + \sum_i a(u_i, v_i))}. \tag{7}$$

The inequality from Eqn. (5) to Eqn. (6) can easily be validated. The inequality from Eqn. (6) to Eqn. (7) holds when $\sum_i (a_{\mathcal{A}_i} + a(u_i, v_i))(a_{\mathcal{B}_i} + a(u_i, v_i)) > (a_\mathcal{A} + \sum_i a(u_i, v_i))(a_\mathcal{B} + \sum_i a(u_i, v_i))$, which is in detail proved in Appendix. This means that at least one partition $(\mathcal{A}_i, \mathcal{B}_i)$ has smaller normalized cut value than $(\mathcal{A}, \mathcal{B})$, and this is in contradiction with the assumption. Consequently, the theorem holds. □

The above theorem reflects a straightforward fact that only one edge needs to be removed to optimize the normalized cut criterion on a tree. Considering the tree in Fig. 1(c), which is rooted from node $r$ and denoted as $\mathcal{T}_r$, we can just remove edge $(u, v)$, then the tree is partitioned into two parts: one, denoted as $\mathcal{T}_v$, is rooted from node $v$, and the other one, denoted as $\mathcal{T}_{r\setminus v}$ and called complementary subtree, is still rooted from the original root $r$ but excludes $\mathcal{T}_v$ and edge $(u, v)$. For convenience, the removed edge $(u, v)$ is used to represent such a bipartition.

The (connected) tree structure only consists of $n - 1$ edges, where $n$ is the number of the nodes. So it only takes $O(n)$ time to find the optimal edge to be split by just traversing all the edges, if all combinations of associations and cuts are pre-computed. Because there is only one edge linking two complementary subtrees, the cut value can be directly obtained. In Fig. 1(c), the cut value of the partition, $cut(\mathcal{T}_{r\setminus v}, \mathcal{T}_v)$, is the similarity $a(u, v)$ of nodes $u$ and $v$.

The difficulty lies in how to efficiently compute the association. A naive method is just exhaustively calculating all kinds of combinations. This will result in computational inefficiency and its time complexity is $O(n^2)$ because there are $O(n)$ possible bipartitions and it takes $O(n)$ time to calculate the association for each bipartition.

To speed up association calculation, we propose a recursive method by exhibiting the properties of association complementarity and overlapping subproblems in evaluating the association. More specifically, our approach is based on the following two properties. The first is association complementarity. From Fig. 1(c), it is obvious that $a_{r\setminus v} = a_r - a_v - 2a(u, v)$, where $a_{r\setminus v} = a_{\mathcal{T}_r\setminus v}$ and $a_r = a_{\mathcal{T}_r}$. Hence, it is sufficient to calculate association values for all subtrees $\mathcal{T}_v$. The second property is overlapping of association evaluation between a subtree and its child trees. According to the definition of association, it can be easily derived that the association of subtree $\mathcal{T}_v$ is equal to the summation of the associations of the subtrees, rooted from $v$'s child nodes, and double cut values between $v$ and $v$'s child nodes. Mathematically, this overlapping can be written as a recursive formulation:

$$a_v = \begin{cases} \sum_{w\in C_v}(a_w + 2a(v, w)) & v \text{ an internal node} \\ 0 & v \text{ a leaf node,} \end{cases} \quad (8)$$

where $C_v$ represents the set of $v$'s child nodes. By this recursion, the associations of all the subtrees can be evaluated in a bottom-up manner from the leaves to the root. In summary, the bisection of a tree consists of two steps:

**Algorithm 2** Normalized tree bipartitioning

1. Calculate recursively association $a_v$ for each ubtree $T_v$ according to Eqn. (8), and association $a_{r/v}$ of subtree $T_{r/v}$.
2. Traverse all the edges to find the optimal bipartition.

### 3.2. $K$-way partitioning

When the partition number $k$ is larger than 2 in $k$-way partitioning, it is difficult to optimize the Ncuts criterion on a tree. Although the generalization of Theorem. 1 for $k$-way partitioning still holds, its solution is not easy. A naive algorithm may check all kinds of combinations independently in a brute force manner, which will lead to $O(n^k)$ time complexity. To our knowledge, there does not exist an exact $k$-way partitioning algorithm similar to the above bipartitioning algorithm. Therefore, we propose a best-first recursive bipartitioning algorithm to approximately optimize $k$-way partitioning. The recursive procedure is as follows:

**Algorithm 3** $K$-way normalized tree partitioning

1. Bisect the input tree $\mathcal{T}$ into $\mathcal{A}_1$ and $\mathcal{A}_2$. Set the number of current partitions $p = 2$.
2. Try to bisect all the current subtrees $\{\mathcal{A}_i\}_{i=1}^{p}$.
3. Find the subtree $\mathcal{A}_t$ that produces the smallest normalized cut value, denote its bisected subtrees $\mathcal{B}_1$ and $\mathcal{B}_2$, and let $\mathcal{A}_t = \mathcal{B}_1$ and $\mathcal{A}_{p+1} = \mathcal{B}_2$, increase $p$ by 1.
4. Output the $k$-way partitions if $p = k$, otherwise go to step 2.

## 4. Analysis and comparison

### 4.1. Time complexity

The proposed clustering method consists of a tree fitting step and a tree partitioning step. In tree fitting, we first build a neighbor graph for all the data points, which can be implemented in $O(n \log n)$ time using approximate nearest neighbor algorithms such as [2]. The minimum entropy spanning tree can be obtained using Prim's or Kruskal's algorithms, which takes $O(n \log n)$ time in our sparse graph case. In normalized tree partitioning, we evaluate the association, using a recursive way costing $O(n)$ time in that each node is only involved once to calculate the association of its parent tree, and the optimal bipartition is found in $O(n)$ time. For $k$-way partitioning, the total time complexity is $O(kn)$. In summary, the time complexity of the clustering method is $O(n(k + \log n))$. The comparison of time complexity with representative existing graph based methods is presented in Tab. 1.

### 4.2. Comparison

The overall comparison with other clustering methods is summarized in Tab. 1. In the following, we give the detailed comparison with the most related two methods: spectral clustering and minimum tree partitioning.

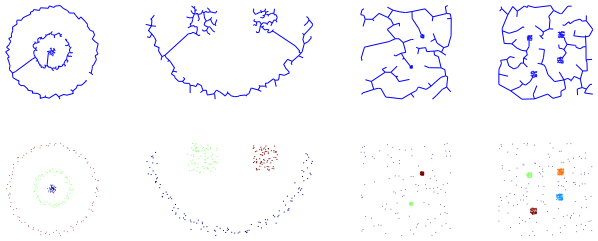
**Vs. spectral clustering** Spectral methods to optimize Ncuts usually consist of two steps: relaxation and discretization. It first relaxes the discrete optimization problem to a continuous problem solved as an eigen-decomposition problem,

| | NTP | SC | GBIS | MTP | AP |
|---|---|---|---|---|---|
| Criterion | Ncuts | Ncuts | Local | Mincuts | Exemplar |
| Complexity | $O(n(k+\log n))$ | $O(n^2)$ | $O(n\log n)$ | $O(n(k+\log n))$ | $O(Tn^2)$ |

Table 1. Comparison of the criterion and time complexity with other methods. The abbreviations have the following meanings: NTP = normalized tree partitioning, SC = spectral clustering, GBIS = the clustering method for graph based image segmentation in [5], MTP = minimum tree partitioning in [15], AP = affinity propagation in [7]. The criteria are of the following meanings: Ncuts = normalized cuts, Mincuts = minimum cuts, Local = local similarity, Exemplar = exemplar identification. Note: The time complexity of SC is for the basic implementation in the case of sparse graph.

(a) Original image (b) NTP (c) MTP

Figure 2. A comparison of segmentation results. (a) shows the original image, (b) shows the result of normalized tree partitioning, which is satisfactory, (c) shows the result of the tree based approach in [15]. Note: NTP = our approach, and MTP = tree approach in [15].



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

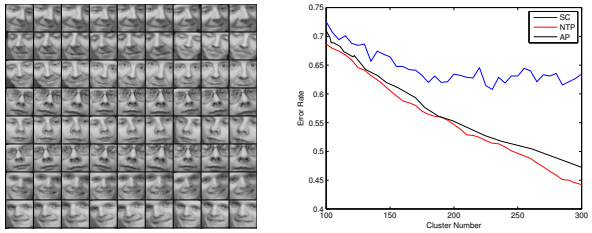
Figure 3. Illustration of normalized tree partitioning on 2D toy examples. The top row shows the fitted trees, and the bottom shows the corresponding clustering results with different colors representing different clusters.

and then discretizes the continuous solution into the discrete solution. The latent approximation in the two steps is not easily predicted. Our approach solves the optimization completely in the discrete space, and the approximation is mostly introduced in tree fitting. This difference implies that normalized tree partitioning has the potential to get superior performances over spectral clustering. Comparison results in Figs. 6 and 7 also demonstrate this superiority.

**Vs. minimum tree partitioning** Compared with the approach based on minimum cuts in [13, 15], we optimize the normalized cut criterion, which was shown to be better as it considers the self-similarity of a cluster. We implemented the algorithms in [13, 15], and the segmentation usually results in cutting small regions. In the example shown Fig. 2(c), the small nose is cut out with the approach in [15].

## 5. Experimental results

We demonstrate our method on toy examples and image segmentation on the Berkeley image dataset.




(a) (b)

Figure 4. (a) Sample face images. (b) Comparison with spectral clustering (SC) and affinity propagation (AP). It can be seen that the performance of NTP is better than SC and AP.

### 5.1. Toy example

First, we illustrate the NTP clustering performance on 2D toy examples, shown in Fig. 3. We show the results of two simple examples shown in (a) and (b) and two difficult examples shown in (c) and (d), which are used in [16] [1]. We construct a data graph by connecting 4 nearest neighbors. The top row in Fig. 3 shows the fitted tree, and the bottom row shows the clustering results of our approach. This example shows that our approach has the ability to cluster the complex data points as shown in Fig. 3 (c) and (d).

Next, we do a clustering experiment on another toy dataset[2]. The dataset contains 900 face images generated from the first 100 face images in the Olivetti database with simple editing. We build a sparse graph on the images by connecting each image and its 50 nearest neighbors. We vary the clustering number between 100 and 300, and compute the error rate against the ground truth (all the images, which are generated from the same original image, are considered to have the same label). The comparison results with spectral clustering and affinity propagation [7] are presented in Fig. 4. It can be seen that NTP performs better than other two methods. It is also worth pointing out that the error rate of NTP is monotonically decreasing, while that of spectral clustering may oscillate, due to the instability of its discretization.

### 5.2. Application to image segmentation

We demonstrate our approach on image segmentation. All the images are firstly smoothed to make our approaches

[1] http://www.vision.caltech.edu/lihi/Demos/SelfTuningClustering.html
[2] http://www.psi.toronto.edu/affinitypropagation/Faces.JPG

more robust to noises. The results based on tree partitioning are all obtained by segmenting the superpixels, which are generated by fragmenting the image. The graph is constructed by setting the superpixels as the nodes and connecting two superpixels iff they are spatial neighbors. The distance between neighboring superpixels is evaluated using a method similar to [5]. In our experiments, the ranges for RGB value in color images are all normalized to [0, 1]. All the running times are given on 1.9 GHz Pentium 4 desktop PC.

#### 5.2.1 Illustration of $k$-way segmentation

We present recursive segmentation results to illustrate $k$-way segmentation shown in Fig. 5. The superpixels are shown in (b), the graph on the superpixels is shown in (c). The minimum entropy spanning tree approximating this graph is shown in (d). Next, the optimum bisegmentation by our method is shown (e), Last, The 3-way and 4-way partitions using the recursive scheme are shown in (f) and (g).

#### 5.2.2 Segmentation on the Berkeley image dataset

We present visual and quantitative comparisons of image segmentation on the Berkeley image segmentation dataset [9]. The dataset contains 200 training images and 100 testing images with size of $480 \times 320$ or $320 \times 480$. Here we do unsupervised segmentation, and hence perform segmentation on both training and testing images.

By comparison, we present segmentation results of other two popular graph partitioning methods, multi-class spectral clustering (SC) based on the Ncuts criterion in [14] and heuristic graph based image segmentation (GBIS) [5]. We run the matlab implementation of spectral clustering [3], and c++ implementation of graph based image segmentation [4]. The segmentation numbers of NTP and spectral clustering are set to be the same. For GBIS, we tune the parameters so that it has similar segmentation numbers on average. Some visual comparisons of segmentation results are shown in Figs. 6 and 7. It can be seen that our results are more satisfactory.

We also present quantitative comparison, and four criteria against the human annotations are used for evaluation: probabilistic rand index (PRI) [12], variation of information (VoI) [10], global consistency error (GCE) [9], and boundary displacement error (BDE) [6]. PRI score counts the number of pairs of pixels whose labels are consistent between the segmentation and the ground truth. The score is averaged over multiple ground truth segmentations to take scale variation into consideration in human perception. VoI score defines the distance between two segmentations as the average conditional entropy of one segmentation given the other, and thus roughly measures the amount of randomness in one segmentation that cannot be explained by the other. GCE score measures the extent to which one segmentation can be viewed as a refinement of the other. Segmentations which are related in this manner are considered to be consistent, because they could represent the same natural image segmented at different scales. BDE score measures the average displacement error of boundary pixels between two segmented images. The segmentation is viewed better if PRI is larger or the other three are smaller.

[3] http://www.cis.upenn.edu/~jshi/software/

[4] http://people.cs.uchicago.edu/~pff/segment/

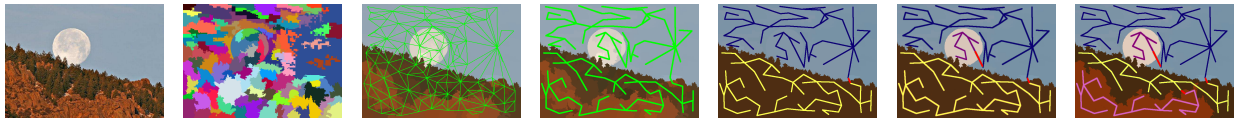
| Method | PRI | VoI | GCE | BDE | RT |
|---|---|---|---|---|---|
| NTP | **0.7521** | **2.4954** | 0.2373 | 16.30 | **0.326** |
| SC | 0.7357 | 2.6336 | 0.2469 | **15.40** | 4.25 |
| GBIS | 0.7139 | 3.3949 | **0.1746** | 16.67 | 0.578 |

Table 2. Quantitative comparison over the Berkeley dataset between normalized tree partitioning (NTP), spectral clustering (SC) and graph-based image segmentation (GBIS). The best score is emphasized in bold fonts.

We report the average scores over the images in Tab. 2. It can be seen that our method performs better than spectral clustering in terms of three criteria: PRI, VoI, and GCE, and better than GBIS in terms of three criteria: PRI, VoI, and BDE. The superiority of NTP over SC is because our method can obtain a better solution by introducing the tree structure, while spectral clustering suffers from the two approximation steps, as discussed in Subsec. 4.2. The superiority of NTP over GBIS comes from the Ncuts criterion considering both inter-similarities and self-similarities of clusters, while GBIS only utilizes the local similarity criterion and has no ability to measure the self-similarity of a cluster. In addition, we also present the running time of all the three methods in the last column of Tab. 2. We can see that the running time (RT) of our approach is much less than GBIS and SC. It should be noted that the running time for our approach is recorded for the whole segmentation process including both preprocessing, graph construction, tree fitting and normalized tree partitioning. Experience shows that PRI and VoI seem to be more correlated with human segmentation in term of visual perception. Therefore, in summary, our approach is more powerful than GBIS and SC in the two perspectives of both segmentation satisfaction and running time.

## 6. Conclusion

In this paper, we proposed a novel cluster method to improve clustering satisfaction and reduce computational cost. It consists of two steps: tree fitting and normalized tree partitioning. The main contributions include: 1) We construct a tree to approximate a graph under the sense of minimum entropy; 2) We propose an effective method to optimize the



(a) Image (b) Superpixels (c) Graph (d) Tree (e) Bipartition (f) 3-way partition (g) 4-way partition

Figure 5. Illustration of recursive bipartitioning.

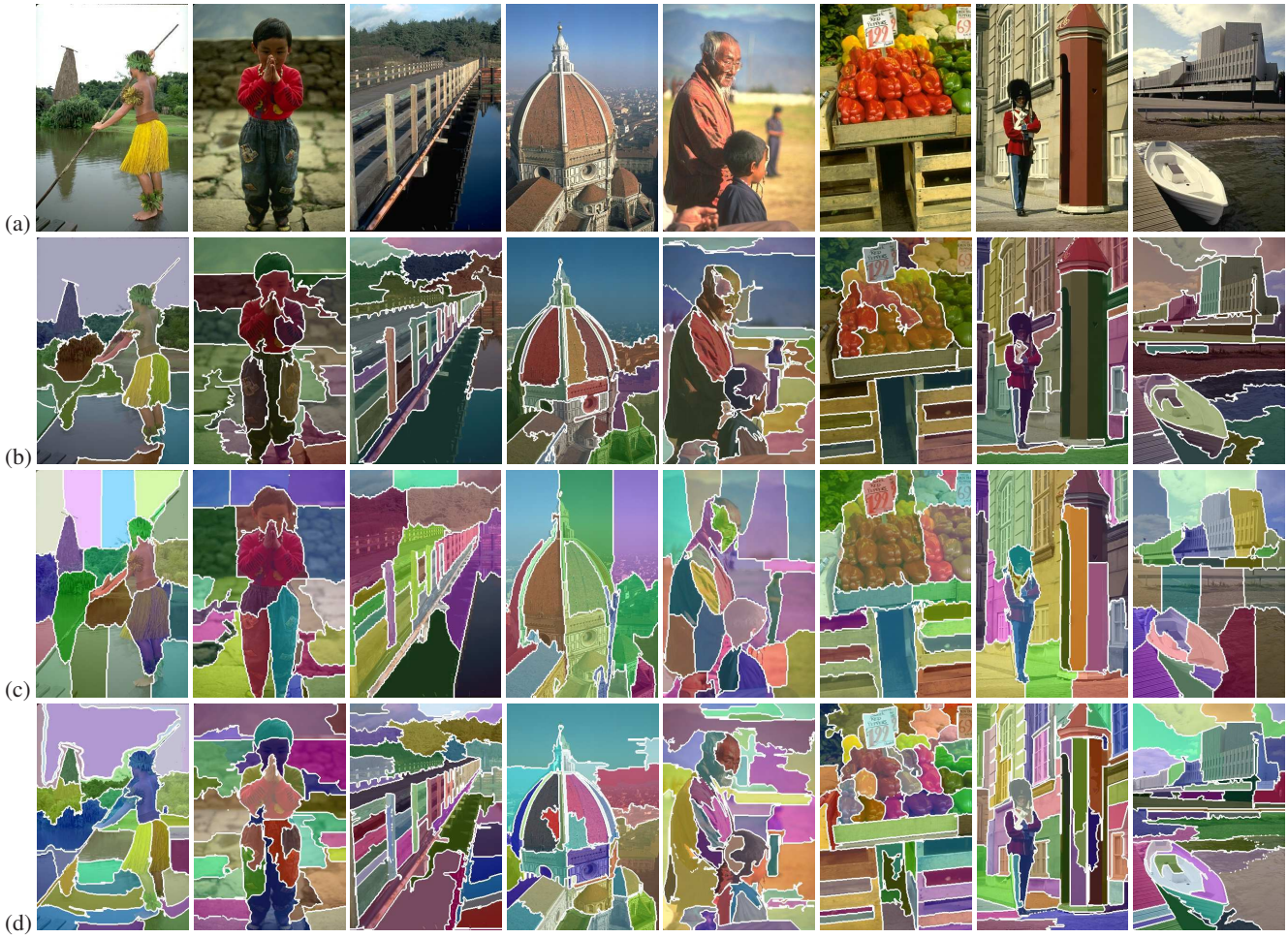


Figure 6. Visual comparisons of image segmentation on the Berkeley dataset. (a) is the original image, (b) is the segmentation of NTP, and (c) and (d) are the segmentation of spectral clustering and graph based image segmentation. It can be seen our results are visually better.

Ncuts criterion on a tree, in which it is theoretically guaranteed to find the exact global optimum solution for bipartitioning; 3) We extend it to $k$-way partitioning in a recursive bipartitioning scheme. The experiments on image segmentation demonstrate the effectiveness and efficiency of our approach.

## Acknowledgments

Yangqing Jia and Changshui Zhang are supported by NSFC (Grant No. 60675009). Long Quan is supported by Hong Kong RGC project 619006.

## References

[1] S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. J. Kriegman, and S. Belongie. Beyond Pairwise Clustering. In *CVPR (2)*, pages 838–845, 2005.

[2] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An Optimal Algorithm for Approximate Nearest Neighbor Searching Fixed Dimensions. *J. ACM*, 45(6):891–923, 1998.

[3] C. K. Chow and C. N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Trans. Information Theory*, 14(3):462–467, May 1968.

[4] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, 2002.

[5] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient Graph-Based Image Segmentation. *Int. J. Comput. Vision*, 59(2):167–181, 2004.

[6] J. Freixenet, X. Muñoz, D. Raba, J. Martí, and X. Cufí. Yet Another Survey on Image Segmentation: Region and Boundary Information Integration. In *ECCV (3)*, pages 408–422, 2002.

[7] B. Frey and D. Dueck. Clustering by Passing Messages Between Data Points. *Science*, 315(5814):972–976, 2007.

[8] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

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

Figure 7. More visual comparisons of image segmentation on the Berkeley dataset. (a) is the original image, (b) is the segmentation of NTP, and (c) and (d) are the segmentation of spectral clustering and graph based image segmentation. It can be seen our results are visually better.


[9] D. R. 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 *ICCV*, pages 416–425, 2001.

[10] M. Meila. Comparing Clusterings: an Axiomatic View. In *ICML*, pages 577–584, 2005.

[11] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

[12] R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward Objective Evaluation of Image Segmentation Algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):929–944, 2007.

[13] R. Urquhart. Graph Theoretical Clustering Based on Limited Neighborhood Sets. *Pattern Recognition*, 15(3):173–187, 1982.

[14] S. X. Yu and J. Shi. Multiclass Spectral Clustering. In *ICCV*, pages 313–319, 2003.

[15] C. Zahn. Graph Theoretic Methods for Detecting and Describing Gestalt Clusters. *IEEE Trans. Computers*, 20.

[16] L. Zelnik-Manor and P. Perona. Self-Tuning Spectral Clustering. In *NIPS*, 2004.

[17] D. Zhou, J. Huang, and B. Schölkopf. Learning with Hypergraphs: Clustering, Classification, and Embedding. In *NIPS*, pages 1601–1608, 2006.


# Appendix

**Lemma.**

$$\sum_{i=1}^{m}(a_{\mathcal{A}_i}+a(u_i,v_i))(a_{\mathcal{B}_i}+a(u_i,v_i)) > (a_{\mathcal{A}}+\sum_{i=1}^{m}a(u_i,v_i))(a_{\mathcal{B}}+\sum_{i=1}^{m}a(u_i,v_i)) \quad (9)$$

*Proof.* Interestingly, subtrees $\{\mathcal{V}_j\}_{j=0}^m$ and edges $\{(u_i,v_i)\}_{i=1}^m$ can be viewed as a super tree $\overline{\mathcal{T}}$ with subtrees as super nodes connected by edges $\{(u_i,v_i)\}_{i=1}^m$.

Then the following two statements holds: (1) $\mathcal{A}$ and $\mathcal{B}$ correspond to the subsets of super nodes with odd depths (blue nodes in Fig. 8) and even depths (red nodes in Fig. 8), respectively; (2) $(\mathcal{A}_i,\mathcal{B}_i)$ can be viewed as a partition of the super tree $\overline{\mathcal{T}}$ by removing edge $(u_i,v_i)$. With those two observations, the following inequality can be easily validated

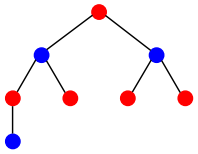


Figure 8. The super tree $(\{\mathcal{V}_j\}_{j=0}^m,\{(u_i,v_i)\}_{i=1}^m)$.

$$a_{\mathcal{A}}a_{\mathcal{B}}=(\sum_{\mathcal{V}_o\in\mathcal{A}}a_{\mathcal{V}_o})(\sum_{\mathcal{V}_e\in\mathcal{B}}a_{\mathcal{V}_e})<\sum_{i=1}^{m}(\sum_{\mathcal{V}_j\in\mathcal{A}_i}a_{\mathcal{V}_j}\sum_{\mathcal{V}_j\in\mathcal{B}_i}a_{\mathcal{V}_j}).$$

Transferring and expanding the left hand side of Eqn. (9), we can have:

$$\begin{aligned}
&\sum_{i=1}^{m}(a_{\mathcal{A}_i}+a(u_i,v_i))(a_{\mathcal{B}_i}+a(u_i,v_i))\\
=\;&\sum_{i=1}^{m}[\sum_{\mathcal{V}_j\in\mathcal{A}_i}a_{\mathcal{V}_j}+2\sum_{(u_j,v_j)\in\mathcal{A}_i}a(u_j,v_j)+a(u_i,v_i)]\\
&[\sum_{\mathcal{V}_j\in\mathcal{B}_i}a_{\mathcal{V}_j}+2\sum_{(u_j,v_j)\in\mathcal{B}_i}a(u_j,v_j)+a(u_i,v_i)]\\
>\;&\sum_{i=1}^{m}[\sum_{\mathcal{V}_j\in\mathcal{A}_i}a_{\mathcal{V}_j}\sum_{\mathcal{V}_j\in\mathcal{B}_i}a_{\mathcal{V}_j}\\
&+(\sum_{\mathcal{V}_j\in\mathcal{A}_i}a_{\mathcal{V}_j}+\sum_{\mathcal{V}_j\in\mathcal{B}_i}a_{\mathcal{V}_j})a(u_i,u_i)\\
&+(2\sum_{(u_j,v_j)\in\mathcal{A}_i}a(u_j,v_j)+a(u_i,v_i))\\
&\times(2\sum_{(u_j,v_j)\in\mathcal{B}_i}a(u_j,v_j)+a(u_i,v_i))]\\
>\;&a_{\mathcal{A}}a_{\mathcal{B}}+\sum_{j=0}^{m}a_{\mathcal{V}_j}\sum_{i=1}^{m}a(u_i,u_i)\\
&+\sum_{i=1}^{m}[a^2(u_i,v_i)+a(u_i,v_i)\sum_{j\neq i}a(u_j,v_j)]\\
=\;&a_{\mathcal{A}}a_{\mathcal{B}}+(a_{\mathcal{A}}+a_{\mathcal{B}})\sum_{i=1}^{m}a(u_i,v_i)+[\sum_{i=1}^{m}a(u_i,v_i)]^2\\
=\;&(a_{\mathcal{A}}+\sum_{i=1}^{m}a(u_i,v_i))(a_{\mathcal{B}}+\sum_{i=1}^{m}a(u_i,v_i)).
\end{aligned}$$

Therefore, the lemma holds. □