Image Segmentation with Context

Anders P Eriksson, Carl Olsson, Fredrik Kahl
anderspe, calle, fredrik@maths.lth.se

Centre for Mathematical Sciences
Lund University, Lund, Sweden

Abstract. We present a technique for simultaneous segmentation and classification of image partitions using combinatorial optimization techniques. By combining existing image segmentation approaches with simple learning techniques we show how prior knowledge can be incorporated into the visual grouping process through the formulation of a quadratic binary optimization problem. We further show how such to efficiently solve such problems through relaxation techniques and trust region methods. This has resulted in an method that partitions images into a number of disjoint regions based on previously learned example segmentations. Preliminary experimental results are also presented in support of our suggested approach.

1 Introduction

Image segmentation is normally defined as the task of distinguishing objects from background in unseen images. This visual grouping process is typically based on low-level cues such as intensity, homogeneity or image contours. Popular approaches include thresholding techniques, edge based methods and region-based methods. Regardless of the method, the difficulty lies in formulating and describing the perception of what constitutes foreground and background in an arbitrary image. Furthermore, such a grouping is also highly contextually driven, certain image regions may be labeled differently depending on the task at hand - are we looking for people, buildings or trees? If one also allows for more labels than only foreground and background, the problem becomes increasingly harder and requires a much higher level of scene understanding. And even then, what constitutes visually relevant regions is not always obvious.

In this paper we make an attempt at addressing the problem of contextually based multiclass image segmentation. By combining image segmentation approaches with standard learning techniques we seek to include prior knowledge into the visual grouping process. Our wish is to segment an image into any number of parts based on previously seen and manually annotated examples.

The approach taken here is based on graph cut techniques from combinatorial optimization. This choice was motivated by the proven success of these methods and that it allowed for a straightforward incorporation of prior knowledge into its formulation. We show how this problem can be stated as a large-scale quadratic

0-1 optimization program with linear constraints. Typically, such proven NP-complete problems are solved by relaxing or simply dropping some constraints and rewriting the problem as one that can be solved efficiently, see for example [1] for semidefinite relaxation techniques. Another popular approach is the energy minimization method of [2] for optimizing objective functions that are submodular; such discrete problems can be solved exactly. Unfortunately, multiclass labeling does not belong to this set of objective functions. Nevertheless, in [3] the authors suggest that these types of problems can be approximated by solving a number of subproblems exactly. We argue that this level of accuracy is not required for image segmentation problems, since the problem itself is vaguely formulated striving for such exactness may be wasteful.

Instead we propose the use of efficient relaxation techniques capable of handling large problem instances. We have examined two different such methods. Spectral relaxation is a standard relaxation technique, based on eigenvalue computations it is well suited for large-scale problems. In [4], a relaxation technique for solving non-submodular large-scale quadratic combinatorial optimization problems is applied to image restoration, binary partitioning and registration. In addition to the theoretical developments in this paper, results on applying these two methods to the problem of multiclass segmentation with prior information will be given.

2 Combinatorial Optimization and Image Segmentation

A graph cut is the process of partitioning a directed or undirected graph into disjoint sets. The concept of optimality of such cuts is usually introduced by associating an energy to each cut. Problems of this kind have been well studied within the field of graph theory but can for graphs with more than only a few nodes be notoriously difficult (that is, NP-hard). Nevertheless, ever since it became apparent that many low-level vision problems can be posed as finding cuts in graphs, these techniques have received a lot of attention in the computer vision community. Graph cut methods have been successfully applied to stereo, image restoration, texture synthesis and image segmentation, for example [5–7]. Below we give a brief overview of graph cuts for image segmentation as well as an introduction to some basic definitions.

2.1 Graph Cuts

Given a graph $G = \{V, E, W\}$, where V denotes its nodes, E its edges and W the affinity matrix, which associates a weight to each edge in E. A cut on a graph is a partition of V into k subsets $A_1, ...A_k$ such that $\bigcup A_i = V, \quad A_i \bigcap A_j = \emptyset, \ i \neq j$. Perhaps the simplest and best known graph cut method is the minimal cut formulation. The min-cut of a graph is the cut that partitions G into disjoint segments such that the sum of the weights associated with edges between the

different segments are minimized. That is, the partition that minimizes

$$C_{min}(\{A_i\}_{i=1}^{k}) = \sum_{i=1}^{k} \sum_{u \in A_i, v \notin A_i} w_{uv}. \quad (1)$$

However, as this is in most cases an NP-hard combinatorial optimization problem, the task of finding the solution can be a formidable one. Instead we attempt to find such cuts by rewriting the original formulation and by relaxing some of the constraints and thus arriving at a problem that can be efficiently solved. First we will describe how our initial problem of segmenting images is connected to graphs cuts.

2.2 Graph Representations of Images

The general approach of constructing an undirected graph from an image is shown in fig. 2.2. Basically each pixel in the image is viewed as a node in a graph.

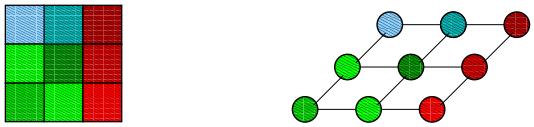


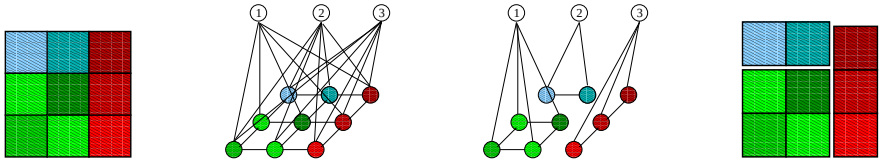
Fig. 1: Graph representation of a 3×3 image.

Edges are formed between nodes with weights corresponding to how alike two pixels are, given some measure of similarity, as well as the distance between them. In an attempt to reduce the number of edges in the graph, only pixels within a small, predetermined neighborhood $\mathcal{N}$ of each other are considered. Cuts made in such a graph will then correspond to a segmentation of the underlying image. Owing to the definition of image-pixel resemblance this segmentation should then be a partition such that pixels close to each other with a high degree of intensity similarity will end up in the same partition. Any spatial structure in the image will hopefully be preserved.

2.3 Including Prior Information

In order to be able to include prior information into the visual grouping process we modify the construction of the graphs in the following way. To the graph G we add k artificial nodes. These nodes do not correspond to any pixels in the image, instead they are meant to represent the k different classes the image is to be partitioned into. The contextual information that we wish to incorporate is modeled by a simple statistical model. Edges between the class nodes and the images nodes are added, with weights proportional to how likely a particular

pixel is to a certain class. With the labeling of the k class nodes fixed, a minimal cut on such a graph should group together pixels according to their class likelihood and still preserving the spatial structure, see fig. 2.



(a) Original image (b) Corresponding graph (c) Multiclass min-cut (d) Resulting segmentation

Fig. 2: A graph representation of an image and an example three-class segmentation. Unnumbered nodes corresponds to pixels and numbered ones to the artificial class nodes.

2.4 Combinatorial Optimization

In this section we derive the optimization problem related to (1). Let $Z = [z_1, \ldots, z_k] \in \{-1, 1\}^{n \times k}$ denote the $n \times k$ assignment matrix for all the n nodes. A 1 in row i of column j signifies that pixel i of the image belongs to class j, and of course -1 in the same position signifies the opposite. If we let W contain the inter-pixel affinities, the min-cut (without pixel class probabilities) can then be written

$$C_{min} = \inf_{Z} \sum_{i=1}^{k} \sum_{\substack{u \in A_i \\ v \notin A_i}} w_{uv} = \inf_{Z} \sum_{i,j,l} w_{jl}(z_{ij} - z_{il})^2 = \inf_{Z} \sum_{i=1}^{k} z_i^T (D - W) z_i. \quad (2)$$

Here D denotes $\mathrm{diag}(W\mathbf{1})$. The assignment matrix Z must satisfy $Z\mathbf{1} = (2-k)\mathbf{1}$. In addition, if the pixel/class-node affinities $P = [p_1, \ldots, p_k]$ (that is, the probabilities of a single pixel belonging to a certain class) are included and also the labels of the class-nodes are fixated, we get

$$C_{min} = \inf_{\substack{Z \in \{-1,1\}^{n \times k} \\ Z\mathbf{1} = (2-k)\mathbf{1}}} \sum_{i=1}^{k} z_i^T \underbrace{(D - W)}_{L} z_i - 2p_i^T z_i = \inf_{\substack{Z \in \{-1,1\}^{n \times k} \\ Z\mathbf{1} = (2-k)\mathbf{1}}} tr\left(Z^T L Z\right) +$$

$$+2 \underbrace{[-p_1^T, \ldots, -p_k^T]}_{b^T} \underbrace{\begin{bmatrix} z_1 \\ \vdots \\ z_k \end{bmatrix}}_{z} = \inf_{\substack{z \in \{-1,1\}^{nk} \\ Z\mathbf{1} = (2-k)\mathbf{1}}} z^T \underbrace{\begin{bmatrix} L & 0 & \ldots & 0 \\ 0 & L & \ldots & 0 \\ 0 & \vdots & \ddots & 0 \\ 0 & \vdots & 0 & L \end{bmatrix}}_{A} z + 2b^T z. \quad (3)$$

As $z \in \{-1,1\}^{nk} \Leftrightarrow z_i^2 = 1$, we arrive at the quadratically constrained quadratic program

$$\mu = \inf_z z^T A z + 2b^T z \tag{4}$$
$$\text{s.t.} \qquad z_i^2 = 1 \tag{5}$$
$$Z\mathbf{1} = (2-k)\mathbf{1}. \tag{6}$$

Since the constraint (5) implies $z^t z = nk$ this redundant equality can be added to (4) without changing the problem. The above problem is still equivalent to the original min-cut formulation (1). For efficiently solving this problem we here turn our attention to two relaxations that are tractable from a computational perspective. When dropping the $z_i^2 = 1$ and $Z\mathbf{1} = (2-k)\mathbf{1}$ constraints we obtain the problem

$$\mu_{tr} = \inf_{||z||^2 = nk} z^T A z + 2b^T z. \tag{7}$$

A common approach for solving this problem is to homogenize (7). That is, we add an extra variable z_{nk+1} and solve

$$\mu_{spec} = \inf_{||z||^2 + z_{nk+1}^2 = nk+1} \begin{pmatrix} z \\ z_{nk+1} \end{pmatrix}^T \begin{pmatrix} A & b \\ b^T & 0 \end{pmatrix} \begin{pmatrix} z \\ z_{nk+1} \end{pmatrix}. \tag{8}$$

Note that if we add the constraint $z_{nk+1} = 1$ to (8) we obtain (7). It is therefore clear that we always have $\mu_{spec} \leq \mu_{tr}$. Equality will only occur if z_{nk+1} happens to be ± 1. The reason for solving (8) instead of (7) is that (8) is easily solved by computing the eigenvector corresponding to the smallest eigenvalue of the matrix

$$H = \begin{pmatrix} A & b \\ b^T & 0 \end{pmatrix}. \tag{9}$$

The downside is of course that, in practice z_{nk+1} is often quite far away ± 1, resulting in poor relaxations. In our case it is easy to see that an incorrect value of z_{nk+1} changes the balance between the effects of the quadratic smoothing therm and the linear term containing the prior information. To remedy this problem we propose to solve (7), using a method borrowed from the trust region problem.

2.5 The Trust Region Subproblem

In this section we review how to solve (7) (see also [4], [8] and [9]). A problem closely related to (7) is

$$\inf_{||z||^2 \leq nk} z^T A z + 2b^T z. \tag{10}$$

This problem is usually referred to as the trust region subproblem. Solving the problem is one step in a general optimization scheme for descent minimization and it is known as the trust region method [10]. Instead of minimizing a general function, one approximates it with a second order polynomial $z^T A z + 2b^T z + c$. A

constraint of the type $||z||^2 \leq m$ then specifies the set in which the approximation is believed to be good (the trust region).

The trust region subproblem have been studied extensively in the optimization literature ([8, 9, 11–13]). A remarkable fact is that it is a non convex problem with no duality gap (see [14]). This is always the case when we have quadratic objective function and only one quadratic constraint. The dual problem of (10) is

$$\sup_{\lambda \leq 0} \inf_{z} z^T A z + 2b^T z + \lambda(nk - z^T z). \tag{11}$$

In [12] is shown that z^* is the global optimum of (10) if and only if (z^*, λ^*) is feasible in (11) and fulfills the following system of equations:

$$(A - \lambda^* I)z^* = -b \tag{12}$$

$$\lambda^*(nk - z^T z) = 0 \tag{13}$$

$$A - \lambda^* I \succeq 0. \tag{14}$$

The first two equations are the KKT conditions for a local minimum, while the third determines the global minimum. From equation (14) it is easy to see that if A is not positive semidefinite, then λ^* will not be zero. Equation (13) then tells us that $||z||^2 = nk$. This shows that for an A that is not positive semidefinite problems (7) and (10) are equivalent. Note that we may always assume that A is not positive semidefinite in (7). This is because we may always subtract mI form A since we have the constant norm condition. Thus replacing A with $A - mI$ for sufficiently large m gives us an equivalent problem with A not positive definite.

A number of methods for solving this problem has been proposed. In [13] semidefinite programming is used to optimize the function $nk(\lambda_{\min}(H(t)) - t)$, where

$$H(t) = \begin{pmatrix} A & b \\ b^T & t \end{pmatrix}, \tag{15}$$

and $\lambda_{\min}$ is the algebraically smallest eigenvalue. In [15] the authors solve $\frac{1}{\psi(\lambda)} - \frac{1}{\sqrt{nk}} = 0$ where $\psi(\lambda) = ||(A - \lambda I)^{-1} b||$. This is a rational function with poles at the eigenvalues of A. To ensure that that $A - \lambda I$ is positive semidefinite a Cholesky factorization is computed. If one can afford this, Cholesky factorization is the preferred choice of method. However, the LSTRS-algorithm developed in [8] and [9] is more efficient for large scale problems. LSTRS works by solving a parameterized eigenvalue problem. It searches for a t such that the eigenvalue problem

$$\begin{pmatrix} A & b \\ b^T & t \end{pmatrix} \begin{pmatrix} y \\ 1 \end{pmatrix} = \lambda_{min} \begin{pmatrix} y \\ 1 \end{pmatrix} \tag{16}$$

or equivalently

$$\begin{aligned} (A - \lambda_{min} I)y &= -b \\ t - \lambda_{min} &= -b^T y \end{aligned} \tag{17}$$

has a solution. Finding this t is done by determining a λ such that $\phi'(\lambda) = nk$, where ϕ is defined by

$$\phi(\lambda) = b^T(A - \lambda I)^\dagger b = -b^T y. \tag{18}$$

It can be shown that λ gives a solution to (17). Since ϕ is a rational function with poles at the eigenvalues of A, it can therefore be expensive to compute. Instead rational interpolation is used to efficiently determine λ. For further details see [8] and [9].

Regardless of relaxation the solution will be a vector with continuous entries that will most likely not fulfil constraints (5) and (6). Obtaining a discrete solution that fulfills all the constraints of the original problem, from the relaxed optima - known as rounding - is hence necessary. From the optima of either relaxation, the vector z^*, a $n \times k$ matrix Z^* is formed and the discrete solution Z is found through non-maximum supression of the rows of Z^*. That is, the largest value in each row of Z^* is set to 1 and the others to -1 thus ensuring that both $Z \in \{-1, 1\}^{n \times k}$ and $Z\mathbf{1} = (2 - k)\mathbf{1}$ holds.

3 Experimental Results

As mentioned in the previous section prior knowledge is incorporated into the graph cut framework through the k artificial nodes. For this purpose we need a way to describe each pixel as well as model the probability of that pixel belonging to a certain class.

The image descriptor in the current implementation is based on color alone. Each pixel is simply represented by their three RGB color channels. The probability distribution for these descriptors are modeled using a Gaussian Mixture Model (GMM).

$$p(v|\Sigma, \mu) = \sum_{i=1}^{k} \frac{1}{\sqrt{2\pi|\Sigma_i|}} e^{(-\frac{1}{2}(v-\mu_i)^T \Sigma_i^{-1}(v-\mu_i))} \tag{19}$$

From a number of manually annotated training images the GMM parameters are then fitted through Expectation Maximization, [16]. This fitting is only carried out once and can be viewed as the learning phase of our proposed method.

The edge weight between pixel i and j and the weights between pixel i and the different class-nodes are given by

$$w_{ij} = e^{(-\frac{r(i,j)}{\sigma_R})} e^{(-\frac{||s(i)-s(j)||^2}{\sigma_W})} \tag{20}$$

$$p_{ki} = \alpha \frac{p(w(i)|i \in k)}{\sum_j p(w(i)|i \in j)}. \tag{21}$$

Here $||\cdot||$ denotes the euclidian norm, $r(i, j)$ the distance between pixel i and j and λ, σ_R and σ_W are tuning parameters weighing the importance of the different features. Hence, w_{ij} contains the inter-pixel similarity, that ensures that the segmentation more coherent. p_i describes how likely a pixel is to belong

to class k. α is a parameter weighting the importance of spatial structure vs. class probability.

Preliminary tests of the suggested approach were carried out on a limited number of images. We chose to segment the images into four simple classes, sky, grass, brick and background. Gaussian mixture model for each of these classes was firstly acquired from a handful of training images manually chosen as being representative of such image regions, see fig. 3. For an unseen image the pixel

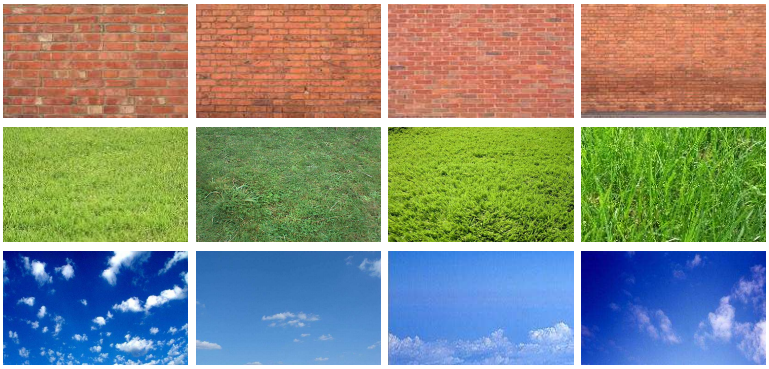


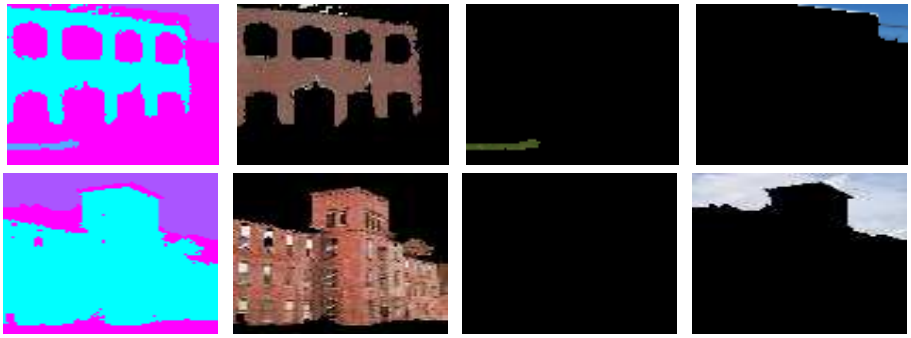
Fig. 3: Sample training images.

affinity matrix W and class probabilities were computed according to (20) and (21). The resulting optimization program was then solved using both the spectral relaxation and the trust region subproblem method. The outcome can be seen in fig. 4. Parameters used in these experiments were $\sigma_R = 1$, $\sigma_W = 1$, $\alpha = 10$ and $\mathcal{N}$ a 9×9 neighborhood structure.

Both relaxations produce visually relevant segmentations, based on very limited training data our proposed approach does appear to use the prior information in a meaningful way. Taking a closer look at the solutions supplied by the trust region method and the spectral relaxation for these two examples does however reveal one substantial difference. The spectral relaxation (8) was reached by ignoring the constraint on the homogenized coordinate $z_{nk+1} = 1$. The solutions to the examples in fig. 4 produces an homogeneous coordinate value of $z_{nk+1} \approx 120$, in both cases. As the class probabilities of the pixels are represented by the linear part of eq. 4, the spectral relaxation, in these two cases, thus yields an image partition that that weights prior information much higher than spatial coherence. Any spatial structure of an image will thus not be preserved, the spectral relaxation is basically just a maximum-likelihood classification of each pixel individually.

We conclude that the trust region formulation seems provide the degree of accuracy required for these types of problems and that spectral relaxation does not.

Original images.



(TSP) Resulting class labelling.

(SR) Resulting class labelling.

Fig. 4: Example segmentation/classification of an image using both Trust Region Subproblem (TSP) formulation and Spectral Relaxation (SR).

4 Summary and Conclusions

In this paper we have proposed a method for multiclass image segmentation with context. We describes how prior information can be brought into a graph cut framework through the use of terminal node weights and learning techniques. In particular, an efficient implementation that brings forward the trust region subproblem formulation as an alternative to existing approaches for finding these image partitions is presented. We also give some promising results on a number of color images.

References

1. Wolkowicz, H., Saigal, R., Vandenberghe, L., eds.: Handbook of Semidefinite Programming. Kluwer Academic Publishers (2000)
2. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE Trans. Pattern Analysis and Machine Intelligence **26** (2004) 147–159
3. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Analysis and Machine Intelligence **23** (2001) 1222–1239
4. Authors: Solving large scale binary quadratic problems: Spectral methods vs. semidefinite programming. submitted to CVPR (2007)
5. Kolmogorov, V., Zabih, R.: Multi-camera scene reconstruction via graph cuts. In: European Conf. Computer Vision. Volume III., Copenhagen, Denmark (2002) 82–96
6. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence **22** (2000) 888–905
7. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": interactive foreground extraction using iterated graph cuts. In: ACM Transactions on Graphics. (2004) 309–314
8. Rojas, M., Santos, S., Sorensen, D.: A new matrix-free algorithm for the large-scale trust-region subproblem. SIAM Journal on optimization **11** (2000) 611–646
9. Rojas, M., Santos, S., Sorensen, D.: Lstrs: Matlab software for large-scale trust-region subproblems and regularization. Technical Report 2003-4, Department of Mathematics, Wake Forest University (2003)
10. Fletcher, R.: Practical Methods of Optimization. John WIley & Sons (1987)
11. Sorensen, D.: Minimization of a large-scale quadratic fuction subject to a spherical constraint. SIAM J. Optim. **7** (1997) 141–161
12. Sorensen, D.: Newton's method with a model trust region modification. SIAM Journal on Nomerical Analysis **19** (1982) 409–426
13. Rendl, F., Wolkowicz, H.: A semidefinite framework for trust region subproblems with applications to large scale minimization. Math. Prog. **77** (1997) 273–299
14. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)
15. Moré, J., Sorensen, D.: Computing a trust region step. SIAM J. Sci. Stat. Comput. **4** (1983) 553–572
16. A. Dempster, M.L., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. J. R. Stat. Soc. (1977)