Learning CRFs Using Graph Cuts

Martin Szummer[1], Pushmeet Kohli[1], and Derek Hoiem[2]

[1] Microsoft Research, Cambridge CB3 0FB, United Kingdom
[2] Beckman Institute, University of Illinois at Urbana-Champaign, USA

Abstract. Many computer vision problems are naturally formulated as random fields, specifically MRFs or CRFs. The introduction of graph cuts has enabled efficient and optimal inference in associative random fields, greatly advancing applications such as segmentation, stereo reconstruction and many others. However, while fast inference is now widespread, parameter learning in random fields has remained an intractable problem. This paper shows how to apply fast inference algorithms, in particular graph cuts, to learn parameters of random fields with similar efficiency. We find optimal parameter values under standard regularized objective functions that ensure good generalization. Our algorithm enables learning of many parameters in reasonable time, and we explore further speedup techniques. We also discuss extensions to non-associative and multi-class problems. We evaluate the method on image segmentation and geometry recognition.

1 Introduction

The availability of efficient and provably optimal inference algorithms, such as graph cuts [1] and its approximate extensions [2], has inspired progress in many areas of computer vision. For example, the efficiency of graph cut based algorithms enables interactive image [3,4] and real-time video segmentation tasks. The optimality guarantees of these algorithms allow computing the maximum a posteriori (MAP) solution of the model distribution.

The ability to compute the minimum energy solution has revealed that simple energy models (e.g., with one unary term and an isotropic smoothing penalty in grid labeling problems) are insufficient to model the complex structures inherent in computer vision problems [5,6]. Despite this knowledge, overly simplistic hand-tuned random field models continue to be common practice. We believe that the continued use of such impoverished models reflects, not a belief in the supremacy of such representations, but the absence of tractable machine learning techniques for large MRF and CRF problems. Currently, the most widely used learning algorithms include cross-validation and simple partition function approximations [7]. However, many works do not perform learning at all and rely on hand-tuned parameters.

In this paper, we describe an efficient and easy-to-implement technique that is capable of learning dozens of parameters from millions of pixels in minutes. Our algorithm is based on the structured support vector machine (SVMstruct)

© Springer-Verlag Berlin Heidelberg 2008

framework of Tsochantaridis *et al.* [8] and the maximum-margin network learning of Taskar *et al.* [9,10]. These works were not focused on computer vision problems. They did not explore graph cuts for learning, and instead chose approximate inference algorithms (such as sum-product loopy BP) which do not perform well on grid-graph problems encountered while dealing with images [6].

For a given parameterized energy function, our goal is to learn the parameters so that the ground truth has the lowest energy by the largest possible margin, or, if that is not possible, that the energy of the ground truth is as close as possible to that of the minimum energy solution. We begin with arbitrary initial parameters and an empty "contrastive" solution set. Then, iteratively, we find the minimum energy solution for the current parameters, add it to our alternative set, and find the parameters that maximize our objective function, considering only the ground truth and the current alternative set of solutions.

When the energy is a linear or quadratic function of the parameters (as is typically the case) and can be optimally minimized, we can estimate the parameters accurately in a highly efficient manner, typically requiring a few dozen iterations for problems involving dozens of unary and pairwise random-field parameters. We also show that, when our objective function is maximized, the sufficient statistics (those statistics necessary to compute the energy) of our solutions will closely match the sufficient statistics of the ground truth. Additionally, we describe several extensions of our technique, evaluating the suitability of alternative objective functions, and dealing with cases for which optimal energy minimization is not possible. We validate our algorithm and its extensions on the problems of image segmentation and multi-class geometry labeling.

Contributions. We present an efficient and practical algorithm to train random field models for images. The contributions of our paper include:

1. Use of graph-cuts to *efficiently* do maximum margin learning of parameters *exactly* for submodular MRFs and CRFs. Generalization to new images is ensured via a large margin regularizer. Approximations such as pseudolikelihood are not required.
2. Learning parameters of non-submodular problems using alpha-expansions.
3. Investigation of loss functions for segmentation and geometry labeling.
4. Formulation of the polygon-bounded image segmentation task, for the purpose of obtaining refined segmentations from rough user interactions.

2 Basic Learning Algorithm

Many computer vision tasks such as segmentation and stereo estimation can be modeled with random fields, which describe a joint probability distribution of all random variables, either jointly or conditionally on the input pixels. The probability is expressed as an exponentiated energy. The energy typically consists of a sum of terms, including node energies for each random variable and interaction energies that model dependencies between adjacent variables. Usually, the energies are parameterized as a linear function of these unary and pairwise features.

Applying this model requires 1) learning the parameters of the model from training data, and 2) inferring the most likely labels for the test data, using the learned parameters. Inference in this model amounts to an energy minimization problem, which can frequently be solved via graph cuts.

Parameter learning aims to find parameters that fit the training data and that also generalize to unseen test data. The learning is formulated as minimizing a loss function on the training set. A common loss function is the negative likelihood of the training labels, but for random fields this leads to an intractable optimization problem. Instead, we employ a tractable approach that uses other loss functions. In this section, we describe a simple algorithm that maximizes the difference between the energy of ground truth labeling and other labelings. In section 3, we refine the loss function to incorporate degrees of labeling error. This makes the method robust to outliers and ensures good generalization to test data. We begin by detailing the model and motivating the loss function.

2.1 The Random Field Model

Let $X = \{\boldsymbol{x}^{(n)}\}_n$ be the input collection of instances $\boldsymbol{x}^{(n)}$, with corresponding labelings $Y = \{\boldsymbol{y}^{(n)}\}_n$, where n indexes the instances. For example, an instance may be an image with $\boldsymbol{x}$ denoting the pixel values and $\boldsymbol{y}$ their segmentation labels (foreground or background). Our model is a conditional random field of the form

$$P(Y \mid X, \boldsymbol{w}) = \frac{1}{Z} \mathrm{e}^{-\sum_n E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w})}, \tag{1}$$

where $\boldsymbol{w}$ are parameters (weights), and Z is a normalizer (the partition function). A typical energy decomposes over nodes $\mathcal{V}$ (e.g. individual pixels) and edges $\mathcal{E}$ (e.g. pairs of adjacent pixels). We consider energies E that are linear in the parameters $\boldsymbol{w}$ and the set of node and edge features $\boldsymbol{\phi}$ such as:

$$E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w}) = \sum_{i \in \mathcal{V}} \boldsymbol{w}_1 \boldsymbol{\phi}_i^{(1)}(y_i, \boldsymbol{x}) + \sum_{(i,j) \in \mathcal{E}} \boldsymbol{w}_2 \boldsymbol{\phi}_{ij}^{(2)}(y_i, y_j, \boldsymbol{x}), \tag{2}$$

which we can abbreviate as $E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})$. Thus, we have a log-linear model in the exponential family.

To infer labels of a test input $\boldsymbol{x}$, given parameters $\boldsymbol{w}$, we will find the maximum a posteriori (MAP) labelings $\boldsymbol{y}^* = \mathrm{argmax}_{\boldsymbol{y}} P(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$.

During training we look for weights $\boldsymbol{w}$ that assign the training labels $\boldsymbol{y}^{(n)}$ a greater than or equal probability of any other labeling $\boldsymbol{y}$ of instance n, i.e.,

$$P(\boldsymbol{y}^{(n)} \mid \boldsymbol{x}^{(n)}, \boldsymbol{w}) \geq P(\boldsymbol{y} \mid \boldsymbol{x}^{(n)}, \boldsymbol{w}) \;\; \forall \boldsymbol{y} \neq \boldsymbol{y}^{(n)} \; \forall n. \tag{3}$$

If this were successful, the inferred MAP labelings for the training data would equal the training labels (except for possible ties). We can cancel the normalizer Z from both sides of the constraints (Eq. 3) and express the constraints in terms of energies

$$E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \leq E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \;\; \forall \boldsymbol{y} \neq \boldsymbol{y}^{(n)} \; \forall n. \tag{4}$$

Thus, we desire weights that give the training labels at least as low an energy as any other label configurations on the training data. However, the inequalities in (4) may have multiple or no solutions. We resolve this by finding the parameters which satisfy the inequalities with the largest possible energy margin γ, so that the ground truth labeling has the lowest energy relative to other labelings. This large margin concept serves to regularize the problem and provide generalization to unseen test data. The margin may be negative if the original inequalities had no solutions. Thus we have

$$\max_{\boldsymbol{w}:\|\boldsymbol{w}\|=1} \gamma \quad \text{such that} \tag{5}$$
$$E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \geq \gamma \quad \forall \boldsymbol{y} \neq \boldsymbol{y}^{(n)} \ \forall n.$$

We have constrained the weight norm $\|\boldsymbol{w}\|$ to unity to prevent weights from growing without bounds.

2.2 An Efficient Learning Algorithm

Here we describe a simple but efficient learning algorithm based on the optimization (5). It is not feasible to solve that program, as it has an exponential number of constraints, one for each possible labeling $\boldsymbol{y}$ of each instance. Instead, we shall perform the optimization over a much smaller set of labelings $\{\boldsymbol{S}^{(n)}\}_n$. We explicitly enforce that the ground truth has lower energy than labelings in this set. We check whether this also holds true for the remaining labelings, by finding a labeling that has the lowest energy via graph cuts, or any other efficient method. If this labeling has lower energy than the ground truth, or doesn't achieve the margin, we add it to the constraint set. Finally, we update the parameters to satisfy the new constraint, and iterate. If the parameters do not change (for example, if we did not find any example better than the ground truth), then we stop. This is detailed in Figure 1. Each step of this algorithm is a standard task that can be solved efficiently. A key observation is that the MAP labeling in Step 1 can be translated to a graph cut problem for many vision tasks. Graph cut algorithms can quickly find global optima for a large class of so-called *submodular* energies [1,11].

Step 3 is written as a general optimization for clarity, but is in fact just a quadratic program (given in Section 3). Thus it is convex and is free from local minima. The overall procedure converges as there are only a finite number of labelings that can be added. Crucially, it converges in a low number of steps in practice, and even in the worst case, is proven to require only a polynomial number of steps if a global optimum can be found in step 1 (see [8]), which is true for the case of submodular energy functions. At convergence, all label configurations outside of $\boldsymbol{S}^{(n)}$ will be at least a margin distance away from the ground truth which is why it is important to study the choice of margin measure (loss function).

The constraint $\boldsymbol{w}_2 \geq 0$ in Eq. 6 is needed in the context of graph cuts to ensure that the problem remains submodular for energies $E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w})$ that would

```
BASIC ALGORITHM
Input:
  – input-labeling pairs $\{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}$ training set
  – empty set of competing low energy labelings: $\boldsymbol{S} = \emptyset$
  – initial parameters: $\boldsymbol{w} = \boldsymbol{w}_0$
Repeat until $\boldsymbol{w}$ is unchanged (within a tolerance)
   Loop over training instances $n$
  1. Find the MAP labeling of instance $n$, using e.g. graph cuts
     $\boldsymbol{y}^* \leftarrow \mathrm{argmin}_{\boldsymbol{y}} E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w})$
  2. Add $\boldsymbol{y}^*$ to the constraint set
     $\boldsymbol{S}^{(n)} \leftarrow \boldsymbol{S}^{(n)} \cup \{\boldsymbol{y}^*\}$
  3. Update the parameters $\boldsymbol{w}$ to ensure ground truth has the lowest energy
```

$$\max_{\boldsymbol{w}:\|\boldsymbol{w}\|=1} \quad \gamma \quad \text{such that} \tag{6}$$
$$E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \geq \gamma \quad \forall \boldsymbol{y} \in \boldsymbol{S}^{(n)} \ \forall n$$
$$\boldsymbol{w}_2 \geq 0$$

Fig. 1. Pseudocode for the basic random field learning algorithm

become non-submodular for $\boldsymbol{w}_2 < 0$ (we have assumed that all pairwise potentials $\boldsymbol{\phi}^{(2)}_{ij}(y_i, y_j, \boldsymbol{x})$ are non-negative and convex); for other exact optimization techniques (e.g. max-product belief propagation on junction trees, or branch and bound), this constraint can be lifted. Multiple parameter optima may exist, but will all result in the same energy for the training data.

Non-submodular Energy Functions. Some problems in computer vision involve non-submodular energy functions that are NP-hard and cannot generally be exactly minimized in polynomial time. Examples include multi-label problems such as multi-class geometric labelling [12] and stereo [2]. In these cases, efficient approximate inference can be used in Step 3 of the cutting plane algorithm in Figure 1. The choice of the optimization technique affects the convergence and correctness guarantees of this method [13].

The approximate algorithms commonly used in computer vision include (1) message passing algorithms such as tree-reweighed message passing (TRW) and loopy belief propagation (LBP), and (2) move-making algorithms such as alpha-expansion [2] and Fast-PD[14]. Algorithms such as TRW and Fast-PD work by formulating the energy minimization problem as an integer programming problem and then solving its linear relaxation. These algorithms in addition to giving an approximate solution, also produce a per-instance lower-bound of the energy which can be used to check how close the approximate solution is to the global minima. We could use this information to isolate good solutions and add them to the solution set. If for a particular set of weights, the lower bound is far from the energy of the resulting solution, we know that this set of parameter is not appropriate as it results in a difficult inference problem and would not lead to good solutions.

For our experiments with the nonsubmodular geometric labeling problem (Section 4.2), we chose to use alpha-expansion. We were motivated by the result [14] showing that alpha-expansion yields close to optimal results for the nonsubmodular Potts model (although our model is more complex).

Relation to Previous Work. This work presents a method for max-margin learning in loopy graphs such as grids (images). In contrast, [15] uses Viterbi on a linear chain, an easier problem not applicable to images. Tsochantaridis et al. [8] and Taskar et al. [10] rely on MAP inference but do not use graphcut for learning. Anguelov et al. [9] use a generic CPLEX solver, which can only work on small vision problems (equivalent to 30000 pixels, unlike 10^7 pixels in our case).

The work of Kumar et al. [7] is most related to us, as it does use graphcut. However, it attempts to use a max likelihood framework, which is not compatible with MAP inference, requiring a drastic approximation of the partition function by a single MAP configuration. Our choice, the large margin framework, allows us to solve the problem exactly with high probability (see Theorem 18 in [8] and [13]).

2.3 Parameter Learning for Image Segmentation

In Figure 2 we illustrate our method with one such problem: the binary image segmentation into foreground (cow) and background based on color models estimated from user-given patches and smoothness constraints. In this example, the energy has a single node term (negative log likelihood of the color given the label) and two interaction terms: a constant penalty and a contrast-based penalty C for neighboring pixels having different labels. Our energy function can be written as

$$E(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}; \boldsymbol{w}) = w_1 \sum_{i \in V} -\log P(x_i^{(n)} \mid y_i^{(n)}) + \tag{7}$$

$$w_2 \sum_{(i,j)\in\mathcal{E}} 1_{y_i^{(n)} \neq y_j^{(n)}} + w_3 \sum_{(i,j)\in\mathcal{E}} 1_{y_i^{(n)} \neq y_j^{(n)}} C(x_i^{(n)}, x_j^{(n)}),$$

where 1 denotes an indicator function equaling one for differing labels. The indicator functions enforce a Potts model penalty on the labelling. The likelihood $P(x_i^{(n)} \mid y_i^{(n)})$ is computed from color models of the foreground and background. These models are built using user-specified cues or seeds. The working of the simple parameter learning algorithm for the image segmentation problem is illustrated in figure (2).

3 Objectives

In this section we consider alternative objectives that are more robust to noisy training data and have refined loss functions. Using the transformation $\|\boldsymbol{w}\| \leftarrow 1/\gamma$, we can write program (5) as a standard quadratic program (recall that $E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w})$ is linear in $\boldsymbol{w}$ (Eq. 2)).

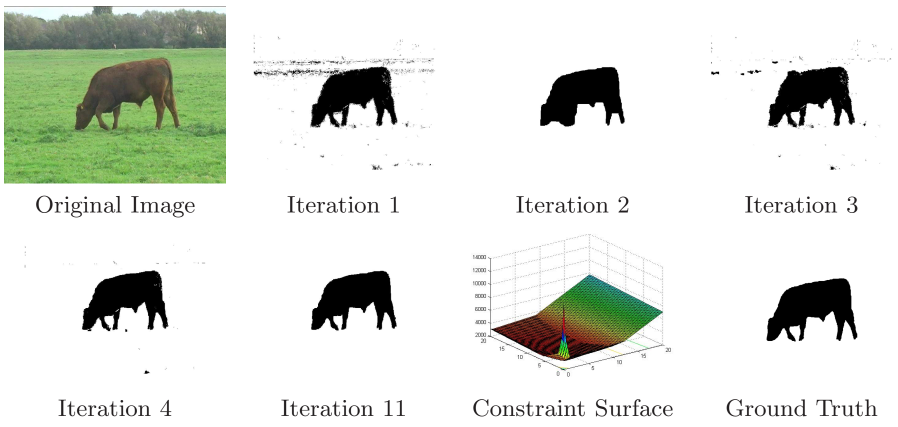

Fig. 2. Parameter Learning for Image Segmentation. The figure illustrates the working of the basic parameter learning algorithm on the image segmentation problem modelled with three parameters (7). The segmentations shown in the figure correspond to the results of using the parameters estimates in iterations 1, 2, 3, 4 and 11 (in order) of the learning algorithm. These solutions are added to the minimal solution set $\boldsymbol{S}^{(n)}$. The algorithm converges in 11 iterations. For the final iteration (11), we also show the plot of the 0-1 loss cost function surface on the smoothness w_2 and contrast w_3 parameters of the segmentation energy (7) keeping w_1 constant. The optimal values of the weights were $w_2 = 1.96$ and $w_3 = 6.94$.

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 \quad \text{s.t.} \tag{8}$$
$$E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \geq 1 \quad \forall \boldsymbol{y} \neq \boldsymbol{y}^{(n)} \ \forall n$$

To add robustness for noisy training data, we relax the margin constraints by adding slack variables ξ_n. Thus, the individual margins may be smaller, but that is discouraged through a slack penalty in the objective, regulated by the slack penalty parameter C.

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_n \xi_n \quad \text{s.t.} \tag{9}$$
$$\begin{aligned} E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) &\geq 1 - \xi_n \ \forall \boldsymbol{y} \neq \boldsymbol{y}^{(n)} \ \forall n \\ \xi_n &\geq 0 \qquad\qquad\qquad \forall n \end{aligned}$$

These are all standard large margin formulations. However, in the context of optimization over joint labelings $\boldsymbol{y}$, we now consider refinements to the standard margin.

3.1 Rescaled Margin

The energy constraints above enforce the same unit margin for all labelings $\boldsymbol{y}$ competing with the training data labeling $\boldsymbol{y}^{(n)}$, regardless of how similar $\boldsymbol{y}$ and

```
MARGIN RESCALED ALGORITHM
Input: as in Fig. 1
Repeat until $\boldsymbol{w}$ is unchanged (within a tolerance)
  Loop over training instances $n$
  1. Run graph cuts to find the MAP labeling of instance $n$
     $\boldsymbol{y}^* \leftarrow \mathrm{argmin}_{\boldsymbol{y}}\, E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - \Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y})$
  2. If $\boldsymbol{y}^* \neq \boldsymbol{y}^{(n)}$, add $\boldsymbol{y}^*$ to the constraint set
     $\boldsymbol{S}^{(n)} \leftarrow \boldsymbol{S}^{(n)} \cup \{\boldsymbol{y}^*\}$
  3. Update the parameters $\boldsymbol{w}$ to ensure ground truth has the lowest energy
```

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_n \xi_n \quad \text{s.t} \quad \forall \boldsymbol{y} \in \boldsymbol{S}^{(n)}\ \forall n \tag{11}$$

$$E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \geq \Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y}) - \xi_n$$
$$\xi_n \geq 0$$
$$\boldsymbol{w}_2 \geq 0$$

Fig. 3. Pseudocode for the margin rescaled learning algorithm

$\boldsymbol{y}^{(n)}$ are (as long as they are not exactly identical). However, for vision problems it is sensible to adapt the margin according to how much competing labelings differ from the ground truth labeling. The idea is to enforce a relatively larger margin for labelings that are far from the ground truth. The desired difference between the ground truth $\boldsymbol{y}^{(n)}$ and a labeling $\boldsymbol{y}$ will be quantified via a loss function $\Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y})$. For example, an appropriate loss may be the Hamming loss $\Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y}) = \sum_i \delta(y_i^{(n)}, y_i)$. For a 0-1 loss function $\Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y}) = \delta(\boldsymbol{y}^{(n)}, \boldsymbol{y})$ we reduce to the previous formulation.

Taskar et al. [16] proposed to rescale the margin to enforce the constraint

$$E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \geq \Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y}) - \xi_n \tag{10}$$

for all $\boldsymbol{y} \neq \boldsymbol{y}^{(n)}$ and all n.

To construct a learning algorithm that takes loss into account, we again iteratively collect a set of competing labelings. Now however, we must find examples that violate these new constraints, which require us to look beyond minimum energy labelings with respect to E.

Importantly, the margin rescaled formulation still allows graph cuts to find the minimum energy solutions under Hamming loss, or any loss that decomposes the same way as the energy. The Hamming loss decomposes across nodes, which allows us to absorb the loss into the node energies, and define a refined energy function $E'(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w}) = E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w}) - \Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y})$. We then arrive at the algorithm in Figure 3. Note that the loss Δ is only incorporated during parameter learning. When inferring labels of test data, energy minima of E are found as before.

Generating multiple solutions using graph cuts. The objectives (programs (8)–(10)) require us to enforce constraints for all $\boldsymbol{y} \neq \boldsymbol{y}^{(n)}$. In step 1 (Figure 3) it is possible that graph cuts find a minimum energy labeling $\boldsymbol{y}^*$ equal to the ground truth $\boldsymbol{y}^{(n)}$. This would not provide any constraint on which to enforce the margin. If this occurred for all training instances, the parameters would not change and the algorithm would terminate without enforcing the margin. This problem can be overcome by adding multiple low-energy labelings at every iteration, instead of just one. The global minima of a submodular function can be found using graph cuts. Kohli and Torr [17] showed how exact min-marginal energies can be efficiently computed for submodular functions. We use these min-marginals to compute the N-best solutions for the energy minimization problem. The time taken for computing these multiple solutions is 3–4 times more than that taken for computing the MAP solution.

3.2 Rescaled Slack

Tsochantaridis *et al.* [8] proposed an alternative way to incorporate a loss into the margin constraints. They rescale the slack variables by the loss, requiring that $\forall \boldsymbol{y} \neq \boldsymbol{y}^{(n)}$:

$$E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}) \geq 1 - \frac{\xi_n}{\Delta(\boldsymbol{y}^{(n)}, \boldsymbol{y})} \tag{12}$$

This formulation has the advantage that the loss Δ and slack penalty C share the same scale. In particular, it enforces the same default margin of 1 for all examples. In contrast, the margin-rescaled formulation requires large margins of labelings differing significantly from the ground truth, which could cause the algorithm to focus on assigning high energies to poor labelings, rather than assigning low energies to labelings close to ground truth. Unfortunately, graph cut optimization cannot directly find labelings that violate the constraints (12). Instead, we use graph cut to find a minimum-energy solution and check whether it violates the constraints.

3.3 Minimum Energy Loss

While the rescaled margin and rescaled slack methods are a large improvement over the 0-1 loss, they may still minimize training error poorly because they focus on the constraints that require the largest slack, rather than the minimum-energy solutions for a particular set of parameters. For instance, in the case of Taskar *et al.*, even if the ground truth is the minimum energy solution, if any high loss solution does not have a sufficiently high margin, the algorithm will change the parameters. The modification of Tsochantaridis *et al.* is better, since the same margin is required of all examples, but the cost function still will often concern itself with solutions that are not minimum-energy solutions (and, thus, would not be returned during inference). We, therefore, provide a new cost function that minimizes the training error more directly.

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_n \xi_n + \frac{C_2}{N}\sum_n \Delta(\boldsymbol{y}^{(n)}, \hat{\boldsymbol{y}}^{(n)}) \tag{13}$$

where $\hat{\boldsymbol{y}}^{(n)} = \operatorname{argmin}_{\boldsymbol{y} \in \boldsymbol{S}^{(n)} \cup \{\boldsymbol{y}^{(n)}\}} E(\boldsymbol{y}, \boldsymbol{x}^{(n)}; \boldsymbol{w})$.

Table 1. Number of training iterations and test accuracy for each cost function using our graph cuts learning method to learn parameters for segmentation. The training data consisted of rough 6 and 12-point polygon segmentations. The method is highly efficient under all cost functions (an iteration of processing 50 images of size 320×240 takes about 45 seconds). Our new loss-augmented functions provide the best test accuracy.

Polygon Type	12 Point Polygon	6 Point Polygon
Cost Function	(Iterations, Accuracy)	(Iterations, Accuracy)
Maximum likelihood, unary only	(N/A, 79.7%)	(N/A, 64.3%)
Pseudo-likelihood	(26, 81.9%)	(21, 74.0%)
0-1 Loss	(13, 80.7%)	(-,-)
Rescaled Margin	(43, 83.0%)	(11, 74.5%)
Rescaled Slack	(13, 82.3%)	(13, 70.7%)
Rescaled Margin + Loss	(45, 85.7%)	(184, 76.1%)
Rescaled Slack + Loss	(90, 84.8%)	(118, 76.2%)

To optimize this objective, we iteratively find the minimum-energy solution for the given parameters and find the parameters that minimize this cost. Loosely speaking, this is a greedy search for minimum training loss, locally approximating the loss gradient with the gradient of the slack. For large C_2, this is guaranteed to converge to a local minimum in training loss.

4 Experiments

The goal of these experiments is to demonstrate the efficacy of the max-margin learning method for computer vision problems, and to study the behavior of different loss functions in this framework. To do this, we perform experiments on two tasks: segmentation and geometry labeling. For these tasks, there are too many parameters to tune by hand. Our results demonstrate that our method is highly efficient and that training with our new loss-augmented cost functions improves test accuracy considerably.

4.1 Refining Segmentations

While completely automatic segmentation is extremely difficult, refining a coarse user-provided segmentation with a nearly pixel-perfect segmentation is achievable and quite useful. For instance, in a photo-editing program, a person could draw a polygon around the object, and the computer would attempt to produce a pixel-accurate segmentation. We evaluate our algorithm on this task, with run-time and accuracy comparisons to other methods. See Figure 4 for some qualitative examples.

The input to our algorithm is an image and a 4, 6 or 12-point polygon that approximates the foreground region. We train on fifty segments from fifty separate images in the Berkeley Segmentation Dataset and test on fifty segments from a different fifty images. We define three unary potentials: a foreground prior,

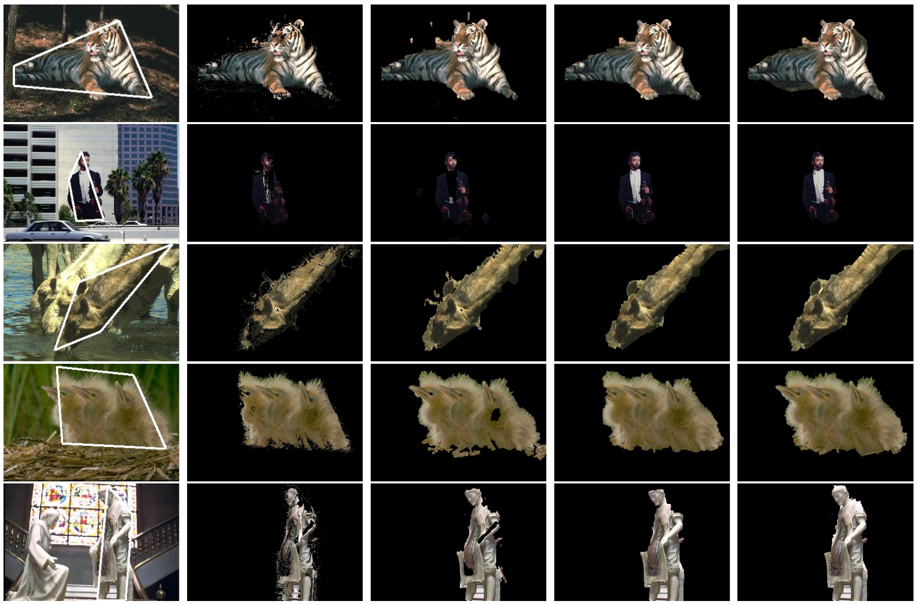

Fig. 4. Parameter learning for image segmentation. Given the input image and a 4-corner polygon denoting the approximate foreground region (column 1), the goal is to produce a refined pixel-perfect segmentation. Shown are the segmentation results from exact maximum likelihood learning using unary features (column 2), and after learning CRF parameters with our algorithm using rescaled margin (column 3), rescaled margin with augmented loss (column 4), and rescaled slack with augmented loss (column 5).

signed distance from the nearest polygon boundary, and a color histogram term. On an eight-connected graph, we define three associative pairwise terms that penalize connected pixels having different labels: constant, intensity contrast-based, and color contrast-based (Euclidean distance of A and B channels in Lab space). We learn separate parameters for horizontal, vertical, and diagonal edges to give a total of three unary and nine pairwise parameters. To evaluate, we define segmentation accuracy as the intersection to union ratio of the ground truth and resulting foreground regions, and report the average test accuracy.

Comparison of Cost Functions. In Table 1, we report accuracy when training based on the cost functions described in Section 3, using 12-point input polygons. The baseline case is to return the input polygon as the final foreground region (80.9% accuracy; note that this is based on a rough human segmentation and does not correspond to a learned model). As expected, the cost function based on 0-1 loss performs poorly (80.7%), while the margin-rescaling and slack-rescaling method offer some improvement (83.0% and 82.3%, respectively).

During training, we define loss as pixel error for the margin-rescaling method to ensure that we can reach the global minimum cost with our graph cut method. We cannot make the same guarantee when using the slack-rescaling method, regardless of the label loss function used, so we define loss as the intersection-union ratio, as is used to report test error. Note that more general (but much slower) minimization methods may be able to apply the slack-rescaling method more effectively. Training with our new loss-augmented cost functions provides significant further improvement to test accuracies of 85.7% and 84.8%, when using margin-rescaling and slack-rescaling, respectively. For comparison, we also trained a model using maximum likelihood including the three unary potentials only; this gave 79.7% accuracy (exact ML training including the pairwise terms is intractable). Note that these improvements of 4-5% over the baseline do not fully convey the large qualitative improvement that can be seen in Figure 4.

We also trained on coarser 6-point polygon inputs. This task is more challenging and accuracy decreases, as seen in table 1. For figure 4, even coarser 4-point polygons were used. We also performed experiments with the parameters learned by pseudolikelihood maximization. We found the training times to be similar to our method, but the resulting accuracies (74.0% for 6-point polygons, 81.9% for 12-point polygons) were below those achieved by our method with our proposed minimum energy loss functions.

4.2 Geometry Estimation

With this experiment, we seek to answer whether our algorithm remains tractable on a large-scale vision problem with several dozen parameters. Our task is to label each pixel in the image into “ground”, “vertical”, or “sky”, using the geometric context dataset provided by Hoiem *et al.* [12].

To do this, we define 8 unary parameters to model the priors, general appearance, and image-specific appearance. The appearance terms are modeled as the logistic outputs of boosted decision tree classifiers trained separately on four sets of features described by Hoiem *et al.*: color, texture, position and shape, and perspective. For each image, we also estimate the color of each class within that image, based on an initial estimate of the geometry likelihoods. Our pairwise interaction terms model texture and color gradients between neighboring pixels and the relative positions. In total there are 72 interaction terms (4 directions, including diagonal, 9 pairwise labelings, and two types of gradients). We could learn, for example, that sky is unlikely to appear directly beneath vertical regions (although it can happen with overhanging objects) or that certain types of gradients are indicative of a boundary for particular geometric classes.

What we are chiefly interested in here, however, is whether our learning algorithm can do anything useful on this large-scale, multi-class learning problem.

Table 2. Accuracy on the geometry estimation problem

Objective	Unary only	Rescaled Margin	Rescaled Slack (Loss)	Rescaled Margin (Loss)
Accuracy	84.6%	87.7%	86.8%	87.0%

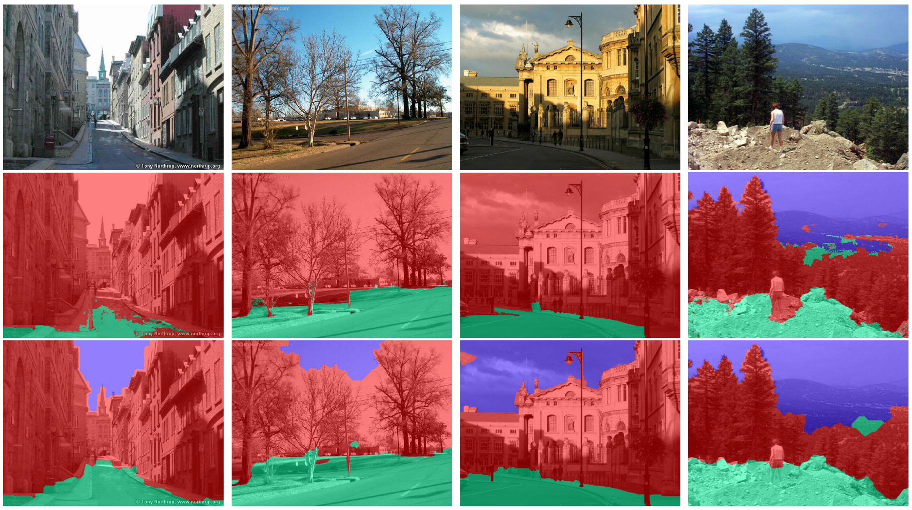

Fig. 5. Results of geometry ("ground", "vertical", or "sky") estimation based on our CRF learning method. Input images (top row), results based on unary energies alone (middle) and the learned CRF (bottom).

To handle the case of multiple classes, we replace our regular graph cut inference step with alpha-expansion graph cuts. Our results, shown in Figure 5 and Table 2 show that we are able to learn parameters that significantly outperform the unary classifier. Despite the large number of parameters, the learning required only about fifty iterations for each objective, indicating that our method is suitable for large-parameter problems.

5 Discussion and Conclusion

We have presented an efficient algorithm to train random field models (MRFs and CRFs) for images. These models are not tractable to train exactly using maximum likelihood. Instead, we start from a standard large-margin framework, and then leverage graphcut to perform inference and thereby arrive at a truly practical algorithm for the computer vision field.

Intuitively, for a learning problem involving many parameters a large set of minimum energy solutions will be needed to constrain the feasible parameter space of our convex program. We note that some experiments required up to 184 iterations, as our parameter learning algorithm only adds one solution to the solution set in each iteration. The rate of convergence of the learning algorithm can be improved by adding multiple samples from the posterior labeling distribution to the solution set in each iteration of the algorithm. We plan to explore this in future work.

References

1. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE Trans. Pattern Anal. Mach. Intell. 26(2), 147–159 (2004)
2. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. 23(11), 1222–1239 (2001)
3. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In: ICCV, pp. I: 105–112 (2001)
4. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut": interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics 23(3), 309–314 (2004)
5. Barbu, A., Zhu, S.C.: Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. IEEE Trans. Pattern Anal. Mach. Intell. 27(8), 1239–1253 (2005)
6. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M.F., Rother, C.: A comparative study of energy minimization methods for Markov random fields. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 16–29. Springer, Heidelberg (2006)
7. Kumar, S., August, J., Hebert, M.: Exploiting inference for approximate parameter learning in discriminative fields: An empirical study. In: EMMCVPR, pp. 153–168 (2005)
8. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. Jrnl. Machine Learning Research 6, 1453–1484 (2005)
9. Anguelov, D., Taskar, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., Ng, A.Y.: Discriminative learning of Markov random fields for segmentation of 3D scan data. In: CVPR, pp. 169–176 (2005)
10. Taskar, B., Chatalbashev, V., Koller, D., Guestrin, C.: Learning structured prediction models: a large margin approach. In: ICML, pp. 896–903 (2005)
11. Boros, E., Hammer, P.L.: Pseudo-boolean optimization. Discrete Applied Mathematics 123(1–3), 155–225 (2002)
12. Hoiem, D., Efros, A.A., Hebert, M.: Geometric context from a single image. In: ICCV (2005)
13. Finley, T., Joachims, T.: Training structural SVMs when exact inference is intractable. In: Proc. Intl. Conf. on Machine Learning (ICML), pp. 304–311 (2008)
14. Komodakis, N., Tziritas, G., Paragios, N.: Fast, approximately optimal solutions for single and dynamic MRFs. In: CVPR (2007)
15. Collins, M.: Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In: ACL-02 conf. Empirical methods in natural lang. processing (EMNLP), pp. 1–8 (2002)
16. Taskar, B., Chatalbashev, V., Koller, D.: Learning associative Markov networks. In: ICML (2004)
17. Kohli, P., Torr, P.H.S.: Measuring uncertainty in graph cut solutions - efficiently computing min-marginal energies using dynamic graph cuts. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 30–43. Springer, Heidelberg (2006)