Parameter Estimation and Energy Minimization for Region-based Semantic Segmentation

M Pawan Kumar, Haithem Turki, Dan Preston, Daphne Koller

Parameter Estimation and Energy Minimization for Region-based Semantic Segmentation

M. Pawan Kumar, Haithem Turki, Dan Preston, and Daphne Koller

Abstract—We consider the problem of parameter estimation and energy minimization for a region-based semantic segmentation model. The model divides the pixels of an image into non-overlapping connected regions, each of which is to a semantic class. In the context of energy minimization, the main problem we face is the large number of putative pixel-to-region assignments. We address this problem by designing an accurate linear programming based approach for selecting the best set of regions from a large dictionary. The dictionary is constructed by merging and intersecting segments obtained from multiple bottom-up over-segmentations. The linear program is solved efficiently using dual decomposition. In the context of parameter estimation, the main problem we face is the lack of fully supervised data. We address this issue by developing a principled framework for parameter estimation using diverse data. More precisely, we propose a latent structural support vector machine formulation, where the latent variables model any missing information in the human annotation. Of particular interest to us are three types of annotations: (i) images segmented using generic foreground or background classes; (ii) images with bounding boxes specified for objects; and (iii) images labeled to indicate the presence of a class. Using large, publicly available datasets we show that our methods are able to significantly improve the accuracy of the region-based model.

1 Introduction

Semantic segmentation offers a useful representation of the scene depicted in an image by assigning each pixel to a specific semantic class (for example, 'person', 'building' or 'tree'). It is a long standing goal of computer vision, and is an essential building block of many ambitious applications such as autonomous driving, robot navigation, content-based image retrieval and surveillance.

Encouraged by their success in low-level vision applications such as image denoising and stereo reconstruction [1], earlier efforts focused on pixel-based models. Here, each pixel is assigned a label using features extracted from a regularly shaped patch around it [2], [3], or at an offset from it [4]. However, the features extracted from such patches are not reliable in the presence of background clutter. For example, a patch around a boundary pixel of a tree may contain sky or building pixels. This may inhibit the model from inferring that trees are mostly green.

To avoid the problem of pixel-based methods, researchers have started to develop region-based models. Such models divide an image into regions, where each region is a set of connected pixels. The reasoning is that if the regions are large enough to provide reliable discriminative features, but small enough so that all the pixels within a region belong to the same semantic class, then we can obtain an accurate segmentation of the image. The first region-based models [5], [6], [7] for high-level vision defined the regions of the image as the segments obtained using a standard bottom-up over-segmentation approach [8], [9]. However, since over-segmentation approaches are agnostic to the task at hand, these regions may not capture the boundaries between the scene entities accurately. To address this issue, some works have suggested heuristics for selecting a good over-segmentation [5], [6]. However, even the best over-segmentation approach is unlikely to provide regions of sufficient accuracy.

In order to obtain regions that capture the boundaries, some efforts have been made to combine multiple over-segmentations. For example, Pantofaru *et al.* [10] suggested taking the intersection of multiple over-segmentation. However, such an approach results in very small regions. Other models suggest using overlapping regions [11], [12], [13], [14]. However, the pixels within the overlap can support two contradicting hypotheses (that is, they can belong to two different semantic classes) thereby overcounting the data.

In this work, we use the region-based model of Gould *et al.* [15], which consists of two layers. The first layer assigns the image pixels to a region, where regions are restricted to be contiguous and non-overlapping. The second layer assigns each region to a unique semantic class, thereby providing the segmentation of the image. The real-valued energy function of the model provides the desirability of an overall

- *M. Pawan Kumar is with the Center for Visual Computing, Ecole Centrale Paris, Châtenay-Malabry, 92295, France. E-mail: pawan.kumar@ecp.fr*
- *Haithem Turki, Dan Preston and Daphne Koller are with the Computer Science Department, Stanford University, Stanford, CA, 94305. E-mail: {htturki,dpreston,koller}@cs.stanford.edu*

output (that is, the combined output of the first and second layers), that is, the lower the energy the better the output. Given an image, its semantic segmentation is inferred by minimizing the energy over all possible outputs. The advantage of the above formulation is two-fold: (i) in addition to the segmentation, the regions themselves would be inferred (including the number of regions), which implies that unlike other models, the regions would be task dependent; and (ii) the regions would be connected and non-overlapping, thereby avoiding data overcounting.

While the region-based model of Gould *et al.* [15] possesses several desirable qualities, its practical deployment poses two formidable problems. First, minimizing the energy associated with this model is extremely challenging due to the large number of possible pixel-to-region assignments. Second, the energy function consists of several thousand parameters that need to be estimated accurately in order to obtain good segmentations via energy minimization.

In order to address the difficulty posed by energy minimization, Gould *et al.* [15] proposed a method that constructs a large dictionary of putative regions using multiple over-segmentations obtained by changing the parameters of a bottom-up approach. Specifically, the putative regions are defined as sets of pixels obtained by merging and intersecting the segments with each other. While merging segments together provides large regions, their intersection with small segments ensures that they align well with the boundary. The set of non-overlapping regions of the image are selected from the dictionary by minimizing the energy function using a simple greedy algorithm. However, while we would expect the dictionary itself to contains accurate regions, the greedy algorithm is susceptible to getting stuck in a bad local minimum. In order to alleviate this issue, we formulate the problem of selecting the best set of regions from a large dictionary as an integer program and design an accurate linear programming (LP) relaxation for it. Furthermore, we show how the LP relaxation can be solved efficiently by suitably modifying the dual decomposition framework [16].

In order to estimate the parameters of the region-based model, Gould *et al.* [15] devised a piecewise learning approach that relies on a fully supervised training dataset. However, we argue that their approach suffers from two significant drawbacks. First, piecewise learning can result in a bad solution. Second, and more important, the collection of fully supervised data is an onerous task as reflected by the small size of the current datasets for semantic segmentation [15], [17]. In order to overcome these problems, we propose the use of diverse data, where the level of annotation varies among the training samples, from pixelwise segmentation to bounding boxes and image-level labels. We design a principled framework for learning with diverse data, with the aim of exploiting the varying degrees of information in the different datasets to the fullest. Specifically, we formulate the parameter learning problem using a latent structural support vector machine (LSVM) [18], [19], where the latent variables model any missing information in the annotation. In order to optimize the objective function of LSVM using the self-paced learning algorithm [20], we modify our LP relaxation based energy minimization algorithm such that it can complete the annotation of weakly supervised images.

We demonstrate the efficacy of our novel energy minimization and parameter estimation methods using large, publicly available datasets. Earlier versions of this article have appeared as [21], [22].

2 The Region-based Model

We begin by providing a formal description of the two-layer region-based model of Gould *et al.* [15]. To allow a reader to quickly refer to one of the various terms used throughout the paper, we summarize the notation in Table 1.

Given an image $\mathbf{X}$, the first layer of the model assigns each pixel p to a unique region R_p (indicated by an integer identifier for the region). The complete assignment of the pixels to the regions is denoted by $\mathbf{R} = \{R_p, p \in \mathbf{X}\}$. We denote the set of all regions specified by $\mathbf{R}$ as $\mathcal{R}$. Note that the number of regions $N_r = |\mathcal{R}|$ is not provided as an input, that is, it has to inferred automatically. The second layer assigns each regions r to a semantic class S_r (once again, indicated by an integer identifier for the semantic class). The complete assignment of the regions to semantic classes is denoted by $\mathbf{S} = \{S_r, r \in \mathcal{R}\}$. The semantic class S_r can either belong to the foreground class, that is, $S_r \in \mathcal{F}$, or the background class, that is, $S_r \in \mathcal{B}$. We denote the number of foreground classes as N_f and the number of background classes as N_b. The set of all $N_s = N_f + N_b$ semantic classes is denoted by $\mathcal{L} = \mathcal{F} \cup \mathcal{B}$. The output of the entire model is denoted by $\mathbf{Y} = (\mathbf{R}, \mathbf{S})$. The energy of the model consists of two types of potentials, unary and pairwise, which are described in detail in the following subsections.

2.1 Unary Potential

For each region $r \in \mathcal{R}$, we define a unary potential $\theta_r(S_r; \mathbf{X})$ for assigning it the class S_r. The value of the unary potential depends on the parameters as well as the features extracted from the image. In more detail, let $\mathbf{u}_r(\mathbf{X}, \mathbf{Y})$ denote the features extracted from the pixels belonging to the region r, which can capture shape, appearance and texture information (for example, green regions are likely to be grass or tree, while blue regions are likely to be sky). We refer the interested reader to [15] for details regarding the features. The unary potential of assigning the region r to a semantic class i is defined as

$$\theta_r(i; \mathbf{X}) = \mathbf{w}_i^\top \mathbf{u}_r(\mathbf{X}, \mathbf{Y}), \tag{1}$$

Term	Description
$\mathbf{X}$	The input image.
$\mathbf{R}$	The assignment of pixels to regions.
$\mathcal{R}$	The set of N_r regions resulting from $\mathbf{R}$.
$\mathcal{E}(\mathcal{R})$	The set of neighboring regions in $\mathcal{R}$.
$\mathcal{F}$	The set of N_f foreground semantic classes.
$\mathcal{B}$	The set of N_b background semantic classes.
$\mathcal{L}$	The set $\mathcal{F} \cup \mathcal{B}$ of all N_s semantic classes.
$\mathbf{S}$	The assignment of regions to semantic classes.
$\mathbf{Y}$	The output $(\mathbf{R}, \mathbf{S})$ of the model.
$\mathbf{u}_r(\mathbf{X}, \mathbf{Y})$	Unary features for $r \in \mathcal{R}$ in image $\mathbf{X}$.
$\mathbf{p}_{rr'}(\mathbf{X}, \mathbf{Y})$	Pairwise features for $r, r' \in \mathcal{E}(\mathcal{R})$ in image $\mathbf{X}$.
$\Psi_i(\mathbf{X}, \mathbf{Y})$	Joint unary feature for class $i \in \mathcal{L}$.
$\Psi_{ij}(\mathbf{X}, \mathbf{Y})$	Joint pairwise feature for classes $i, j \in \mathcal{L}$.
$\Psi(\mathbf{X}, \mathbf{Y})$	Joint feature vector of input $\mathbf{X}$ and output $\mathbf{Y}$.
$\mathbf{w}_i$	Unary parameters for class $i \in \mathcal{L}$.
$\mathbf{w}_{ij}$	Pairwise parameters for the classes $i, j \in \mathcal{L}$.
$\mathbf{w}$	Set of all parameters of the model.
$\theta_r(i)$	Unary potential for region r belonging to class i.
$\theta_{rr'(i,j)}$	Pairwise potential for regions r and r' belonging to classes i and j respectively.
$\mathcal{D}$	Dictionary of regions.
$\mathcal{S}$	Super-pixels, i.e. intersections of regions in $\mathcal{D}$.
$\mathcal{E}$	Set of neighboring regions in $\mathcal{D}$.
$\mathcal{L}^I$	Augmented label set $\mathcal{L} \cup \{l_0\}$.
$\overline{\theta}_r, \overline{\theta}_{rr'}$	Potentials for augmented label set.
$\mathbf{y}$	Integer variables in set $SELECT(\mathcal{D})$. Also used in the linear programming relaxation.
$(\mathcal{D}_T, \mathcal{E}_T)$	Regions and edges for first slave problem.
$\mathcal{C}(s)$	Set of regions in D covering super-pixel s, which specify the second slave problem.
$(\mathcal{D}_Q, \mathcal{E}_Q)$	Regions and edges for third slave problem.
$\mathcal{M}_Q$	Marginal polytope of valid assignments for $\mathcal{D}_Q$.
l_f, l_b	Generic foreground and background classes.
$\mathcal{F}'$	Set of foreground classes and l_b.
$\mathcal{B}'$	Set of background classes and l_f.
$\mathbf{P}$	Pixelwise segmentation using $\mathcal{F}'$ or $\mathcal{B}'$.
$\mathcal{T}$	Training set with diverse data.
$\Delta(\mathbf{P}, \mathbf{Y})$	Loss function to compare a pixelwise segmentation and the model output.
$\mathcal{Y}(\mathbf{P})$	Set of outputs consistent with $\mathbf{P}$.
$\mathbf{B}$	Bounding box annotation.

TABLE 1

Summary of the notation used in the paper. The first part corresponds to the model notation. The second part corresponds to the notation specific to energy minimization. The third part corresponds to the notation specific to parameter estimation.

where $\mathbf{w}_i$ is the unary parameter corresponding to the semantic class i. Note that the unary potential implicitly depends on the output $\mathbf{Y}$ since it is assumed that the $\mathbf{Y}$ defines the region r and assigns it to the class i. In order to avoid a cluttered notation, the dependency of the unary potential on $\mathbf{Y}$ has not been made explicit.

2.2 Pairwise Potential

For each pair of neighboring variables, whose corresponding regions share at least one boundary pixel, we define a pairwise potential $\theta_{rr'}(S_r, S_{r'}; \mathbf{X})$ for assigning classes S_r and $S_{r'}$ to r and r' respectively. Similar to the unary potential, the value of the pairwise potential depends on the parameters and the image features. In more detail, let $\mathbf{p}_{rr'}(\mathbf{X}, \mathbf{Y})$ refer to the features extracted using the pixels belonging to regions r and r', which can capture contrast and contextual information (for example, boats are likely to be above water, while cars are likely to be above road; see [15] for details). The pairwise potential for assigning the regions r and r' to semantic classes i and j respectively is defined as

$$\theta_{rr'}(i, j; \mathbf{X}) = \mathbf{w}_{ij}^\top \mathbf{p}_{rr'}(\mathbf{X}, \mathbf{Y}), \tag{2}$$

where $\mathbf{w}_{ij}$ is the pairwise parameter corresponding to the classes i and j. Once again, the pairwise potential implicitly depends on the output $\mathbf{Y}$ since it is assumed that $\mathbf{Y}$ defines the regions r and r' and assigns them to the class i and j respectively.

2.3 Energy Function

Given an image $\mathbf{X}$ and an output $\mathbf{Y}$, we define the unary feature for a semantic class i as

$$\Psi_i(\mathbf{X}, \mathbf{Y}) = \sum_{r \in \mathcal{R}} \delta(S_r = i) \mathbf{u}_r(\mathbf{X}, \mathbf{Y}), \tag{3}$$

that is, it is the sum of the features over all the regions that are assigned the class i in the output $\mathbf{Y}$. Similarly, we define the pairwise feature for the semantic classes i and j as

$$\Psi_{ij}(\mathbf{X}, \mathbf{Y}) = \sum_{(r,r') \in \mathcal{E}(\mathcal{R})} \delta(S_r = i) \delta(S_{r'} = j) \mathbf{p}_{rr'}(\mathbf{X}, \mathbf{Y}), \tag{4}$$

that is, it is the sum of the features over all pairs of neighboring regions that are assigned the classes i and j respectively in the output $\mathbf{Y}$. Here, $\mathcal{E}(\mathcal{R})$ is the set of pairs of regions that share at least one boundary pixel. The main reason for computing the pairwise features only between neighboring regions is computational efficiency. However, the methods presented in this paper can handle any arbitrary set $\mathcal{E}(\mathbf{R})$ (including the set of all pairs of regions) at the cost of additional computation. Using the unary and pairwise features, we define the joint feature vector of the input $\mathbf{X}$ and output $\mathbf{Y}$ of the model as

$$\Psi(\mathbf{X}, \mathbf{Y}) = [\Psi_i(\mathbf{X}, \mathbf{Y}), \forall i; \Psi_{ij}(\mathbf{X}, \mathbf{Y}), \forall i, j]. \tag{5}$$

In other words, the joint feature vector of the input and the output is the concatenation of the unary features for all semantic classes and the pairwise features for all pairs of semantic classes. Similarly, we define the parameter of the model as

$$\mathbf{w} = [\mathbf{w}_i, \forall i; \mathbf{w}_{ij}, \forall i, j], \tag{6}$$

that is, the parameter is the concatenation of the unary parameters for all semantic classes and the pairwise parameters for all pairs of semantic classes.

The energy of the model can then be written concisely using the parameter and the joint feature vector as follows:

$$\begin{aligned} E(\mathbf{Y}; \mathbf{X}, \mathbf{w}) &= \mathbf{w}^\top \Psi(\mathbf{X}, \mathbf{Y}) \\ &= \sum_{r \in \mathcal{R}} \theta_r(S_r) + \sum_{(r,r') \in \mathcal{E}(\mathcal{R})} \theta_{rr'}(S_r, S_{r'}). \end{aligned} \tag{7}$$

Note that we have dropped the input $\mathbf{X}$ from the notation of the individual potentials to avoid clutter.

3 Energy Minimization

For the model described above, we now consider the problem of obtaining the output $\mathbf{Y}^*$ that minimizes the energy function. This will provide us with the best set of regions for the task ($\mathbf{R}$) as well as their classes ($\mathbf{S}$). In this section, we assume that the parameters $\mathbf{w}$ have been provided. In the next section, we will describe a max-margin framework for estimating the parameters using a diverse training dataset.

As noted earlier, the main difficulty in energy minimization arises due to the fact that there are many possible assignments $\mathbf{R}$ that group pixels into regions. Specifically, for a given $H \times W$ image there can be as many as HW regions (that is, each pixel is a region). Hence the total number of possible assignments $\mathbf{R}$ is $(HW)^{(HW)}$. Furthermore, the feature vectors $\mathbf{u}_r(\mathbf{X}, \mathbf{Y})$ and $p_{rr'}(\mathbf{X})$ require the regions to be connected, which is well-known to be a difficult constraint to impose [23], [24].

In order to overcome these problems, we make use of bottom-up over-segmentation approaches. Specifically, we minimize the energy using the following two steps: (i) construct a large dictionary of connected putative regions using multiple over-segmentations; and (ii) select the set of regions that minimize the energy (that is, infer $\mathbf{R}$ and $\mathbf{S}$). We begin by describing our algorithm for selecting regions from a dictionary. We then provide details of the dictionary that we found to be effective in our experiments.

3.1 Region Selection as Optimization

Given a dictionary of regions, we wish to select a subset of regions such that: (i) the entire image is explained by the selected regions; (ii) no two selected regions overlap with each other; and (iii) the energy $E(\mathbf{Y}; \mathbf{X}, \mathbf{w})$ is minimized. Note that the dictionary itself may contain overlapping regions of any shape or size. We do not place any restrictions on it other than the assumption that it contains at least one set of disjoint regions that explains the entire image. In other words, the problem of region selection can be considered as a special case of optimization over the output $(\mathbf{R}, \mathbf{S})$ such that the regions specified by $\mathbf{R}$ are restricted to belong to a given dictionary. We formulate the above task as an integer program and provide an accurate linear programming (LP) relaxation for it.

3.1.1 Integer Programming Formulation

Before describing the integer program, we need to set up some notation. We denote the dictionary of regions by $\mathcal{D}$. The intersection of all the regions in $\mathcal{D}$ defines a set of super-pixels $\mathcal{S}$. The set of all regions that contain a super-pixel $s \in \mathcal{S}$ is denoted by $\mathcal{C}(s) \subseteq \mathcal{D}$. Finally, the set of neighboring regions is denoted by $\mathcal{E}$, where two regions r and r' are considered neighbors of each other (that is, $(r, r') \in \mathcal{E}$) if they do not overlap and share at least one boundary pixel.

To formulate our problem as an integer program, we define the set $\mathcal{L}^I = \mathcal{L} \cup \{l_0\}$, which we refer to as the label set. The additional label l_0 is used to model whether a region has been selected or not. Furthermore, we define binary variables $y_r(i)$ for each region $r \in \mathcal{D}$ and $i \in \mathcal{L}^I$. These variables indicate whether a particular region is selected and if so, which class it takes. Specifically, if $y_r(0) = 1$ then the region r is not selected, else if $y_r(i) = 1$ where $i \in \mathcal{L}$ then the region r is assigned the class i. Similarly, we define binary variables $y_{rr'}(i, j)$ for all neighboring regions $(r, r') \in \mathcal{E}$ such that $y_{rr'}(i, j) = y_r(i) y_{r'}(j)$. Furthermore, we define unary and pairwise potentials corresponding to the set $\mathcal{L}^I$ as

$$\begin{aligned} \overline{\theta}_r(i) &= \begin{cases} \theta_r(i) & \text{if } i \in \mathcal{L}, \\ 0 & \text{otherwise,} \end{cases} \\ \overline{\theta}_{rr'}(i, j) &= \begin{cases} \theta_{rr'}(i, j) & \text{if } i, j \in \mathcal{L}, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{8}$$

The problem of region selection can then be formulated as the following integer program:

$$\begin{aligned} \min_{\mathbf{y} \in SELECT(\mathcal{D})} \quad & \sum_{r \in \mathcal{D}, i \in \mathcal{L}^I} \overline{\theta}_r(i) y_r(i) + \\ & \sum_{(r,r') \in \mathcal{E}, i, j \in \mathcal{L}^I} \overline{\theta}_{rr'}(i, j) y_{rr'}(i, j), \end{aligned} \tag{9}$$

where the feasible set $SELECT(\mathcal{D})$ is specified using the following constraints:

$$\begin{aligned} & y_r(i), y_{rr'}(i, j) \in \{0, 1\}, \forall r, r' \in \mathcal{D}, i, j \in \mathcal{L}^I, \\ & \sum_{i \in \mathcal{L}^I} y_r(i) = 1, \forall r \in \mathcal{D}, \\ & \sum_{j \in \mathcal{L}^I} y_{rr'}(i, j) = y_r(i), \forall (r, r') \in \mathcal{E}, i \in \mathcal{L}^I, \\ & \sum_{i \in \mathcal{L}^I} y_{rr'}(i, j) = y_{r'}(j), \forall (r, r') \in \mathcal{E}, j \in \mathcal{L}^I, \\ & \sum_{r \in \mathcal{C}(s)} \sum_{i \in \mathcal{L}} y_r(i) = 1, \forall s \in \mathcal{S}. \end{aligned} \tag{10}$$

The first set of constraints ensure that the variables $\mathbf{y}$ are binary. The second constraint implies that each region r should be assigned one label from the set $\mathcal{L}^I$. The third constraint enforces $y_{rr'}(i, j) = y_r(i) y_{r'}(j)$. The final constraint, which we call covering constraint, restricts each super-pixel to be covered by exactly one selected region.

3.1.2 Linear Programming Relaxation

Problem (9) is a generalization of pairwise energy minimization (with the additional covering constraints) and is therefore NP-hard. While we can use integer program solvers such as CPLEX to find its optimal solution, this may not be computationally feasible. To overcome this deficiency, we develop a linear programming relaxation approach, which we found to be faster empirically by a factor of nearly 2.

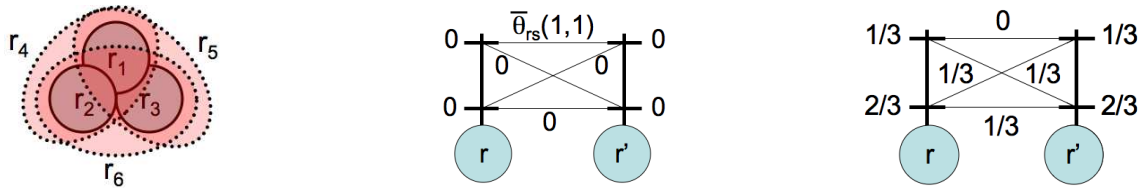

Fig. 1. **(a)** *Neighboring regions r_1, r_2 and r_3, shown using solid lines, consist of a single super-pixel. The regions shown using dashed lines are formed by two super-pixels. Specifically, $r_4 = r_1 \cup r_2$, $r_5 = r_1 \cup r_3$ and $r_6 = r_2 \cup r_3$.* **(b)** *The potentials corresponding to the clique of size 6 formed by the regions. The branches (horizontal lines) along the trellis (the vertical line on top of a region) represent the different labels that each region may take. We consider a two label case here. The unary potential $\overline{\theta}_r(i)$ is shown next to the i^{th} branch of the trellis on top of region r. The pairwise potential $\overline{\theta}_{rr'}(i,j)$ is shown next to the connection between the i^{th} branch of r and the j^{th} branch of r'. The only non-zero potential $\overline{\theta}_{rr'}(1,1) > 0$ corresponds to selecting both the regions r and r'. The optimal assignment of the clique must have an energy greater than 0 since at least two neighboring regions must be selected.* **(c)** *The optimal solution of the LP relaxation. The value of $y_r(i)$ is shown next to the i^{th} branch of r and the value of $y_{rr'}(i,j)$ is shown next to the connection between the i^{th} and j^{th} branches of r and r' respectively. Note that the solution satisfies all cycles inequalities, that is, $\sum_{(r,r')\in\mathcal{E}_C} y_{rr'}(0,0) + y_{rr'}(1,1) \geq 1$, where $\mathcal{E}_C$ is a cycle. Hence the solution lies within the feasible region of the relaxation. However, it can be easily verified that its objective function value is 0, thereby proving that the relaxation is not tight.*

The simplest approach one can take is to solve problem (9) by allowing the variables $\mathbf{y}$ to take (possibly fractional) values in the interval $[0, 1]$. The resulting LP relaxation is similar to the standard relaxation for energy minimization in pairwise random fields [25], [26], with the exception of the additional covering constraints. However, this relaxation is very weak when the pairwise potentials are not submodular (roughly speaking, when they encourage neighboring regions to take different labels) [27]. For example, consider the case where each region is either selected or not (that is, $|\mathcal{L}^I| = 2$). For two neighboring regions r and r', the pairwise potential $\overline{\theta}_{rr'}(\cdot,\cdot)$ is 0 if one or both regions are not selected and $\overline{\theta}_{rr'}(1,1)$ otherwise (as defined by equation (8)). If $\overline{\theta}_{rr'}(1,1) > 0$ then the neighboring regions are encouraged to take different labels; that is, the pairwise potentials are non-submodular. This results in frustrated cycles for which the standard LP relaxation provides a weak approximation [28] (we tested this empirically for our problem, but do not include the results due to space limitations).

There are two common ways to handle non-submodular problems in the literature: (i) applying the roof duality relaxation [28], [29]; and (ii) using message passing algorithms [30], [31], [32] based on cycle inequalities [32], [33]. Unfortunately, both these methods are not directly applicable in our case. Specifically, roof duality does not allow us to incorporate the covering constraints. Adding cycle inequalities still results in a weak approximation (for example, see Fig. 1). Instead, we consider the marginal polytope [26] of cliques that are formed by three neighboring regions along with all the regions that overlap with at least one of the three regions (for example, the clique of six regions formed in Fig. 1). At first sight, this may seem to be complicated since a large number of constraints are required to specify the marginal polytope of cliques. However, in the following subsection, we show that the overall relaxation can be solved efficiently using dual decomposition, which does not even require us to explicitly specify the clique constraints.

3.2 Solving the Relaxation

We use the dual decomposition framework that is well-known in the optimization community [16] and has recently been introduced in computer vision [34]. We begin by describing the general framework and then specify how it can be modified to efficiently solve our relaxation.

3.2.1 Dual Decomposition

Consider the following convex optimization problem: $\min_{\mathbf{z}\in\mathcal{F}} \sum_{k=1}^{M} g_k(\mathbf{z})$, where $\mathcal{F}$ represents the convex feasible region of the problem. The above problem is equivalent to the following: $\min_{\mathbf{z}_k\in\mathcal{F},\mathbf{z}} \sum_k g_k(\mathbf{z}_k)$, s.t. $\mathbf{z}_k = \mathbf{z}$. Introducing the additional variables $\mathbf{z}_k$ allows us to obtain the dual problem as

$$\max_{\boldsymbol{\lambda}_k} \min_{\mathbf{z}_k\in\mathcal{F},\mathbf{z}} \left(\sum_k g_k(\mathbf{z}_k) + \sum_k \boldsymbol{\lambda}_k(\mathbf{z}_k - \mathbf{z}) \right), \qquad (11)$$

where $\boldsymbol{\lambda}_k$ are the Lagrange multipliers. Note that if a function g_k depends on only a subset of variables belonging to the feasible region $\mathcal{F}_k \subseteq \mathcal{F}$, then we can also restrict the variables $\mathbf{z}_k \in \mathcal{F}_k$. Differentiating the dual function with respect to $\mathbf{z}$ we obtain the constraint that $\sum_k \boldsymbol{\lambda}_k = \mathbf{0}$, which implies that we can discard $\mathbf{z}$ from the above problem. The simplified form of the dual suggests the following strategy for solving it. We start by initializing $\boldsymbol{\lambda}_k$ such that $\sum_k \boldsymbol{\lambda}_k = \mathbf{0}$. Keeping the values of $\boldsymbol{\lambda}_k$ fixed, we solve the following slave problems: $\min_{\mathbf{z}_k\in\mathcal{F}} (g_k(\mathbf{z}_k) + \boldsymbol{\lambda}_k\mathbf{z}_k)$.Upon obtaining the optimal solutions $\mathbf{z}_k^*$ of the slave problems, we update the values of $\boldsymbol{\lambda}_k$ by projected subgradient descent where the subgradient with respect to $\boldsymbol{\lambda}_k$ is $\mathbf{z}_k^*$. In other words, we update $\boldsymbol{\lambda}_k \leftarrow \boldsymbol{\lambda}_k + \eta_t \mathbf{z}_k^*$ where η_t is the learning rate at iteration t. In order to satisfy the constraint $\sum_k \boldsymbol{\lambda}_k = \mathbf{0}$ we project the value of $\boldsymbol{\lambda}_k$ to $\boldsymbol{\lambda}_k \leftarrow \boldsymbol{\lambda}_k - \left(\sum_k \boldsymbol{\lambda}_k\right)/M$. By choosing η_t such that $\eta_t \to 0$ and $\sum_{t'=1}^{t} \eta_{t'} \to \infty$ as $t \to \infty$, this iterative strategy known as dual decomposition converges to

the globally optimal solution of the original convex problem. We refer the reader to [16] for details.

3.2.2 Dual Decomposition for Selecting Regions

When using dual decomposition, it is crucial to select slave problems whose optimal solutions can be computed quickly. With this in mind, we choose three types of slave problems. Below we describe each of these slave problems and justify their use by providing an efficient method to optimize them.

The first type of slave problems is similar to the one used for energy minimization in pairwise random fields. Specifically, each slave problem is defined using a subset of regions $\mathcal{D}_T \subseteq \mathcal{D}$ and edges $\mathcal{E}_T \subseteq \mathcal{E}$ that form a tree. For each such graph $(\mathcal{D}_T, \mathcal{E}_T)$ we define the following problem:

$$\begin{aligned} \min_{\mathbf{y} \geq 0} \quad & \sum_{r \in \mathcal{D}_T, i} \left(\frac{\overline{\theta}_r(i)}{n_r} + \lambda_r^1(i) \right) y_r(i) + \\ & \sum_{(r,r') \in \mathcal{E}_T, i, j} \left(\frac{\overline{\theta}_{rr'}(i,j)}{n_{rr'}} + \lambda_{rr'}^1(i,j) \right) y_{rr'}(i,j), \\ \text{s.t.} \quad & \mathbf{y} \geq 0, \sum_{i \in \mathcal{L}^I} y_r(i) = 1, \\ & \sum_{j \in \mathcal{L}^I} y_{rr'}(i,j) = y_r(i), \sum_{i \in \mathcal{L}^I} y_{rr'}(i,j) = y_{r'}(j), \end{aligned} \tag{12}$$

where n_r and $n_{rr'}$ are the total number of slave problems involving $r \in \mathcal{D}$ and $(r, r') \in \mathcal{E}$ respectively. Note that the LP relaxation has an integral optimal solution for trees [25], [26]. In fact, the above problem can be solved efficiently using belief propagation [35].

The second type of slave problems correspond to the covering constraints. Specifically, for each $s \in \mathcal{S}$ we define:

$$\begin{aligned} \min_{\mathbf{y} \geq 0} \quad & \sum_{r \in \mathcal{C}(s), i} \left(\frac{\overline{\theta}_r(i)}{n_r} + \lambda_r^2(i) \right) y_r(i) \\ \text{s.t.} \quad & \sum_{r \in \mathcal{C}(s)} \sum_{i \in \mathcal{L}} y_r(i) = 1, \sum_{i \in \mathcal{L}^I} y_r(i) = 1. \end{aligned} \tag{13}$$

Similar to problem (13), the above problem has an integral optimal solution since its constraint matrix is totally unimodular. It can be solved by finding the region r^* and its class i^* in $O(N_s|\mathcal{C}(s)|)$ time such that

$$(r^*, i^*) = \operatorname*{argmin}_{r \in \mathcal{C}(s), i \in \mathcal{L}} \left(\frac{\overline{\theta}_r(i)}{n_r} + \lambda_r^2(i) \right). \tag{14}$$

The optimal solution of problem (13) is obtained by setting $y_{r^*}(i^*) = 1$ and the rest of the variables to 0.

The third type of slave problems correspond to the clique constraints defined in §3.1. Specifically, for every clique defined by $\mathcal{D}_Q \subseteq \mathcal{D}$ and $\mathcal{E}_Q \subseteq \mathcal{E}$, we specify:

$$\begin{aligned} \min_{\mathbf{y} \in \mathcal{M}_Q} \quad & \sum_{r \in \mathcal{D}_Q, i} \left(\frac{\overline{\theta}_r(i)}{n_r} + \lambda_r^3(i) \right) y_r(i) + \\ & \sum_{(r,r') \in \mathcal{E}_Q, i, j} \left(\frac{\overline{\theta}_{rr'}(i,j)}{n_{rr'}} + \lambda_{rr'}^3(i,j) \right) y_{rr'}(i,j), \end{aligned} \tag{15}$$

where $\mathcal{M}_Q$ is the marginal polytope of the clique. Note that since $\mathcal{M}_Q$ is the convex hull of all valid integral assignments and the objective function is linear, it follows that the above problem has an integral optimal solution. In order to solve the above problem, we first determine the number of valid integral assignments. To this end, let r_1, r_2 and r_3 be the three regions that define the clique $\mathcal{D}_Q$ (for example, see Fig. 1). Let $\mathcal{C}(r_i)$ by the set of regions overlapping with the region r_i (including r_i itself). In order for an assignment to be valid, we can only select one region from each set $\mathcal{C}(r_i)$ (otherwise the regions will overlap with each other). Thus, the above problem reduces to picking one region from each set $\mathcal{C}(r_i)$ and assigning it to one of N_s semantic classes, which has a total complexity of $O(N_s^3|\mathcal{C}(r_1)||\mathcal{C}(r_2)||\mathcal{C}(r_3)|)$.

We iteratively solve the above slave problems and update $\boldsymbol{\lambda}$. Upon convergence, we obtain the primal solution (that is, the subset of regions and their labels) in a similar manner to [30], [34], [36]. Briefly, this involves sequentially considering each super-pixel (in some arbitrary order) and picking the best region for it according to the values of the dual variable $\boldsymbol{\lambda}$. Typically, we obtain a very small primal-dual gap using our approach.

3.3 Generating the Dictionaries

Ideally, we would like to form a dictionary that consists of all the regions obtained by merging and intersecting the segments provided by a bottom-up approach. However, this will clearly result in a very large dictionary that cannot be handled even using our efficient algorithm. Instead, we iteratively search over the regions using the following strategy. We initialize our dictionary with the regions obtained from one (very coarse) over-segmentation. In the subsequent iterations, we consider two different types of dictionaries (similar to [15]). The first dictionary consists of the current set of regions $\mathcal{D}_{cur}$ (those that have provided the best explanation of the image until now, in terms of the energy) as well as all the regions obtained by merging two neighboring regions in $\mathcal{D}_{cur}$. The second type of dictionary uses multiple over-segmentations to define the regions. Specifically, in addition to $\mathcal{D}_{cur}$, it also consists of all the regions obtained by merging every segment from the over-segmentations with all its overlapping and neighboring regions in $\mathcal{D}_{cur}$. Similarly, it also contains all the regions obtained by intersecting every segment with all its overlapping regions in $\mathcal{D}_{cur}$. While using the first dictionary results in larger regions (by merging neighboring regions together), the second dictionary allows us to correct any mistakes (merging two incompatible regions) by considering intersections of segments and regions.

In order to speed-up the overall energy minimization algorithm, we make use of a shrinking [37] and

a move-making strategy [38], which are explained in detail in the appendices.

4 Parameter Estimation

Given a training dataset that consists of images with different types of ground-truth annotations, our goal is to learn accurate parameters for the region-based model. To this end, we design a principled framework for learning with diverse data, with the aim of exploiting the varying degrees of information in the different datasets to the fullest: from the cleanliness of pixelwise segmentations, to the vast availability of bounding boxes and image-level labels. Specifically, we formulate the parameter learning problem using a latent structural support vector machine (LSVM) [18], [19], where the latent variables model any missing information in the annotation. For this work, we focus on three types of missing information: (i) the specific class of a pixel labeled using a generic foreground or background class (for example, images from the VOC segmentation dataset [17] or the Stanford background dataset [15]); (ii) the segmentation of an image annotated with bounding boxes of objects (for example, images from the VOC detection dataset [17]); and (iii) the segmentation of an image labeled to indicate the presence of a class (for example, images from the ImageNet dataset [39]). We show how the energy minimization algorithm described in the previous section can be suitably modified to obtain an approximate but accurate solution of the optimization problem corresponding to LSVM using the self-paced learning algorithm [20].

4.1 Learning with Generic Classes

To simplify the discussion, we first focus on the case where the ground-truth annotation of an image specifies a pixelwise segmentation that includes generic foreground or background labels. As will be seen in subsequent subsections, the other cases of interest, where the ground-truth only specifies bounding boxes for objects or image-level labels, will be handled by solving a series of LSVM problems that deal with generic class segmentations.

4.1.1 Notation

As in the previous section, we denote an image by $\mathbf{X}$ and an output of the model by $\mathbf{Y}$. The joint feature vector of the input and the output is denoted by $\Psi(\mathbf{X}, \mathbf{Y})$. The energy of an output is equal to $\mathbf{w}^\top \Psi(\mathbf{X}, \mathbf{Y})$, where $\mathbf{w}$ are the parameters of the model. The best segmentation of an image is obtained using energy minimization, that is, $\min_{\mathbf{Y}} \mathbf{w}^\top \Psi(\mathbf{X}, \mathbf{Y})$.

We denote the labels corresponding to the generic foreground and the generic background as l_f and l_b respectively. Let $\mathcal{F}' = \mathcal{F} \cup \{l_b\}$ denote the augmented foreground label set, which includes the generic background, and let $\mathcal{B}' = \mathcal{B} \cup \{l_f\}$ denote the augmented background label set, which includes the generic foreground. Note that the sets $\mathcal{F}'$ and $\mathcal{B}'$ are used to specify the ground-truth pixelwise segmentation of the VOC segmentation dataset and the Stanford background dataset respectively. The ground-truth pixelwise segmentation of an image $\mathbf{X}$ provided by the user will be denoted by $\mathbf{P}$. In other words, a pixel $p \in \mathbf{X}$ is assigned to a label $P_p \in \mathcal{F}' \cup \mathcal{B}'$ in the segmentation. We denote the training dataset as $\mathcal{T} = \{(\mathbf{X}_k, \mathbf{P}_k), k = 1, \cdots, n\}$, where $\mathbf{X}_k$ is an image and $\mathbf{P}_k$ is the corresponding segmentation.

Given an output $\mathbf{Y}$ of the model, it would be convenient to denote the semantic class assigned to a pixel p as Y_p. In order to simplify the subsequent discussion, we define the set of outputs consistent with a pixelwise segmentation $\mathbf{P}$, denoted by $\mathcal{Y}(\mathbf{P})$. An output $\mathbf{Y}$ belongs to $\mathcal{Y}(\mathbf{P})$ if and only if the following conditions hold for every pixel $p \in \mathbf{X}$: (i) if $P_p \in \mathcal{L}$, then $Y_p = P_p$; (ii) if $P_p = l_f$, then $Y_p \in \mathcal{F}$ (that is, if a pixel belongs to the generic foreground class, then it must be assigned a specific foreground class by the output); and (iii) if $P_p = l_b$, then $\mathbf{Y}_p \in \mathcal{B}$. Note that for any given pixelwise segmentation $\mathbf{P}$, the set $\mathcal{Y}(\mathbf{P})$ is very large as it consists of all outputs with pixel-to-region assignments $\mathbf{R}$ that are consistent with the given segmentation (that is, the boundaries of the regions are restricted to lie within the same semantic class, or along the segmentation boundaries). This is true even when the pixelwise segmentation does not use the generic foreground and background labels.

4.1.2 Learning as Risk Minimization

Given the dataset $\mathcal{T}$, we learn the parameters $\mathbf{w}$ by training an LSVM. Briefly, an LSVM minimizes an upper bound on a user-specified risk, or loss, $\Delta(\mathbf{P}, \hat{\mathbf{Y}})$. Here, $\mathbf{P}$ is the ground-truth segmentation and and $\hat{\mathbf{Y}}$ is the predicted output for a given set of parameters. In this work, we specify the loss using the overlap score, which is the measure of accuracy for the VOC challenge [17]. For an image labeled using the set $\mathcal{F}'$, we define the loss function as

$$\Delta(\mathbf{P}, \hat{\mathbf{Y}}) = 1 - \frac{1}{|\mathcal{F}'|} \sum_{i \in \mathcal{F}'} \frac{\left| \mathcal{P}_i(\mathbf{P}) \cap \mathcal{P}_i(\hat{\mathbf{P}}) \right|}{\left| \mathcal{P}_i(\mathbf{P}) \cup \mathcal{P}_i(\hat{\mathbf{Y}}) \right|}, \tag{16}$$

where the function $\mathcal{P}_i(\cdot)$ returns the set of all the pixels labeled using class i. Note that when i is the generic background, then $\mathcal{P}_i(\hat{\mathbf{Y}})$ is the set of all pixels labeled using any specific background class. A similar loss function can be defined for images labeled using the set $\mathcal{B}'$. Formally, the parameters are learned by solving the following non-convex optimization problem:

$$\begin{aligned} \min_{\mathbf{w}, \xi_k \geq 0} \quad & \frac{\lambda}{2} ||\mathbf{w}||^2 + \frac{1}{n} \sum_{k=1}^{n} \xi_k, \\ \text{s.t.} \quad & \mathbf{w}^\top \Psi(\mathbf{X}_k, \overline{\mathbf{Y}}_k) - \min_{\mathbf{Y} \in \mathcal{Y}(\mathbf{P}_k)} \Psi(\mathbf{X}_k, \mathbf{Y}) \\ & \geq \Delta(\mathbf{P}_k, \overline{\mathbf{Y}}_k) - \xi_k, \forall \overline{\mathbf{Y}}_k. \end{aligned} \tag{17}$$

Intuitively, the problem requires that for every image the energy of the best output that is consistent with the ground-truth segmentation should be less than the energy of all other outputs. The desired margin between the energy of the two outputs is proportional to the value of the loss.

4.1.3 Learning an LSVM

Algorithm 1 describes the main steps of learning an LSVM using our recently proposed self-paced learning (SPL) algorithm [20]. Unlike the concave convex procedure (CCCP) [18], [19] used in previous works, which treats all images equally, SPL automatically chooses a set of easy images at each iteration (in step 4), and uses only those images to update the parameters.

Algorithm 1 *The self-paced learning algorithm for* LSVM. *The training dataset consists of a set of images along with their ground-truth pixelwise segmentations that can label a pixel using a generic foreground or a generic background class. The hyperparameter σ_0 is chosen such that half the training samples are considered easy, that is, $v_k = 1$, at the first iteration. The hyperparameters $\mu = 1.3$ and $\epsilon = 10^{-3}$.*

input $\mathcal{T}$, $\mathbf{w}_0$, σ_0, μ, ϵ.

1: $\mathbf{w} \leftarrow \mathbf{w}_0$, $\sigma \leftarrow \sigma_0$.
2: **repeat**
3: Impute the best output consistent with the pixelwise segmentation as

$$\mathbf{Y}_k = \underset{\mathbf{Y} \in \mathcal{Y}(\mathbf{P}_k)}{\operatorname{argmin}} \mathbf{w}^\top \Psi(\mathbf{X}_k, \mathbf{Y}). \tag{18}$$

4: Compute the slack variables ξ_k as

$$\xi_k = \max_{\overline{\mathbf{Y}}_k} \quad \Delta(\mathbf{P}_k, \overline{\mathbf{Y}}_k) - \mathbf{w}^\top \Psi(\mathbf{X}_k, \overline{\mathbf{Y}}_k) + \mathbf{w}^\top \Psi(\mathbf{X}_k, \mathbf{Y}_k). \tag{19}$$

Using ξ_k, define variables $v_k = \delta(\xi_k \leq \sigma)$, where $\delta(\cdot) = 1$ if its argument is true and 0 otherwise.
5: Update the parameters by solving the following problem that only considers easy images:

$$\begin{aligned} \min_{\mathbf{w}, \xi_k \geq 0} \quad & \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{1}{n}\sum_{k=1}^{n} v_k \xi_k, \\ \text{s.t.} \quad & \mathbf{w}^\top \left(\Psi(\mathbf{X}_k, \overline{\mathbf{Y}}_k) - \Psi(\mathbf{X}_k, \mathbf{Y}_k)\right) \\ & \geq \Delta(\mathbf{P}_k, \overline{\mathbf{Y}}_k) - \xi_k, \forall \overline{\mathbf{Y}}_k. \end{aligned} \tag{20}$$

6: Change the threshold $\sigma \leftarrow \sigma\mu$.
7: **until** Decrease in objective (17) is below tolerance ϵ and all images have been labeled as easy.

Following our earlier work [20], we choose the initial threshold σ_0 such that half the images are considered easy in the first iteration, and set the annealing factor $\mu = 1.3$. These settings have been shown to work well in practice for a large number of applications [20]. Ideally, we would like to be able to optimize over $\mathbf{Y}_k$ and $\mathbf{w}$ simultaneously. However, this is not possible since the optimization problem (17) is non-convex. Instead, SPL alternates between optimizing one set of variables while keeping the other fixed. This process is guaranteed to obtain a local minimum or saddle point solution for problem (17). In order to avoid learning from images whose outputs are never imputed correctly, we measure the accuracy of the model after each iteration using a validation set (different from the test set). We report test results using the parameters that are the most accurate on the validation set.

Let us take a closer look at what is needed to learn the parameters of our model. The first step of SPL requires us to impute the output of each training image $\mathbf{X}$, given the ground-truth segmentation $\mathbf{P}$, by solving problem (18). In other words, this step completes the output (finds the pixel-to-region assignment, and labels all pixels using a specific foreground or background class in the set $\mathcal{L}$) to provide us with a positive example. We call this segmentation-consistent inference. The second step requires us to find an output with low energy but high loss (a negative example) by solving problem (19). We call this loss-augmented inference. The third step requires solving the convex optimization problem (20). In this work, we solve it using stochastic subgradient descent [40], where the subgradient for a given image $\mathbf{X}$ is the joint feature vector $\Psi(\mathbf{X}, \mathbf{Y}^*\})$, where $\mathbf{Y}^*$ is obtained by solving problem (19). We refer the interested reader to [40] for details.

To summarize, in order to learn from generic class segmentations, we require two types of inference algorithms—segmentation-consistent inference and loss-augmented inference. Unfortunately, both the inference problems are NP-hard, and thus, cannot be solved optimally by a computationally efficient algorithm. However, the structured SVM and LSVM frameworks have been shown to perform well empirically when using approximate inference algorithms [41], [42]. Inspired by their success, we modify our energy minimization algorithm described in the previous section for approximate but accurate inference.

4.1.4 Segmentation-Consistent Inference

The goal of segmentation-consistent inference is to impute the output that minimizes the energy under the constraint that it does not contradict the ground-truth segmentation, which specifies a pixelwise segmentation using generic classes. In other words, a pixel marked as a specific class must belong to a region labeled as that class. Furthermore, a pixel marked as generic foreground (background) must be labeled using a specific foreground (background) class.

For a given dictionary of regions $\mathcal{D}$, segmentation-consistent inference is equivalent to the following integer program (IP):

$$\min_{\overline{\mathbf{y}} \in SELECT(\mathcal{D})} \boldsymbol{\theta}^\top \overline{\mathbf{y}} \qquad \text{s.t.} \qquad \Delta(\mathbf{P}, \overline{\mathbf{y}}) = 0. \tag{21}$$

The set $SELECT(\mathcal{D})$ refers to the set of all valid assignments to $\overline{\mathbf{y}}$, as specified in equation (10). The

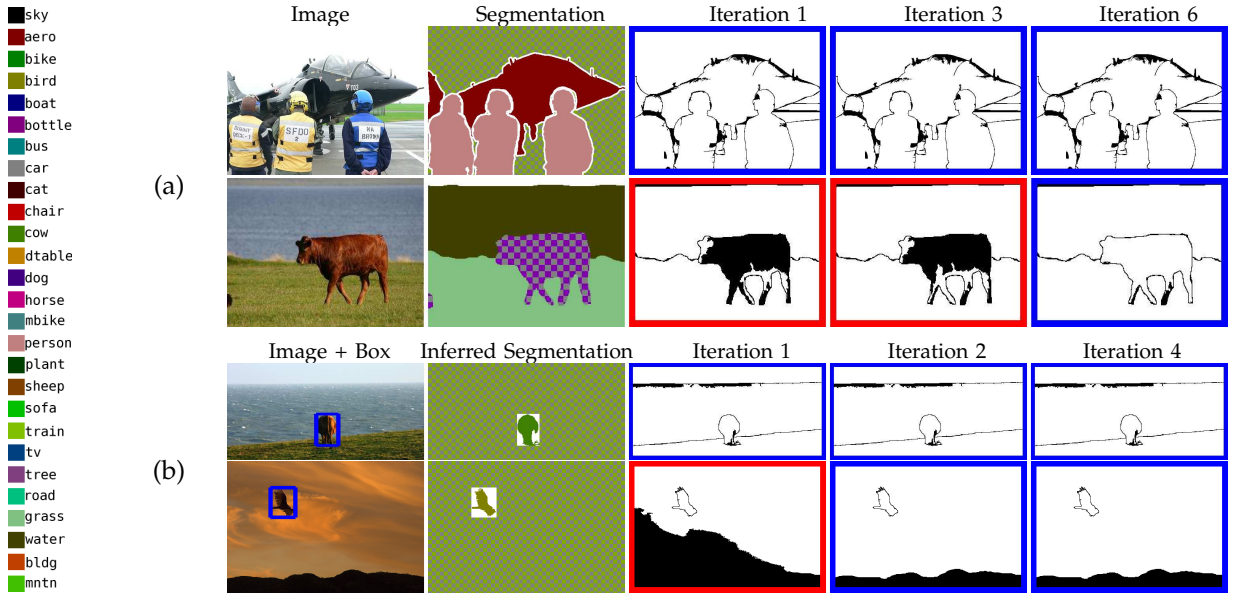

Fig. 2. *Outputs obtained using segmentation-consistent inference during different iterations of* SPL. **(a)** *Images annotated with generic classes. Column 2 shows the segmentation (where the checkered patterns indicate generic classes). In columns* $3-5$*, pixels labeled using the correct specific-class by segmentation-consistent inference are shown in white, while pixel labeled using the wrong specific-class are shown in black (we used the ground-truth specific-class segmentations of these images only for the purpose of illustration; these ground-truth segmentation were not used during training). A blue surrounding box on the output implies that the example was selected as easy by* SPL*, while a red surrounding box indicates that is wasn't selected during the specified iteration. Note that* SPL *discards the image where the cow (row 2) is incorrectly labeled.* **(b)** *Images annotated using bounding boxes. Column 2 shows the segmentation obtained using bounding box inference. The objects have been accurately segmented. Furthermore,* SPL *discards the image where the sky (row 2) is incorrectly labeled.*

constraint $\Delta(\mathbf{P}, \overline{\mathbf{y}}) = 0$ (where we have overloaded Δ for simplicity to consider $\overline{\mathbf{y}}$ as its argument) ensures that the imputed latent variables are consistent with the ground-truth. Similar to the energy minimization problem discussed in the previous section, the above IP is solved approximately by relaxing the elements of $\overline{\mathbf{y}}$ to take values between 0 and 1 and introducing the clique constraints, which results in a tight linear program (LP).

Fig. 2(a) shows examples of the segmentation obtained using the above segmentation-consistent inference over different iterations of SPL. Note that the segmentations obtained are able to correctly identify the specific classes of pixels labeled using generic classes. The quality of the segmentation, together with the ability of SPL to select correct images to learn from, results in an accurate set of parameters.

4.1.5 Loss-Augmented Inference

The goal of loss-augmented inference is to find an output that minimizes the energy while maximizing the loss (as shown in problem (19)), which can be formulated as the following IP:

$$\min_{\overline{\mathbf{y}} \in SELECT(\mathcal{D})} \quad \boldsymbol{\theta}^{\top} \overline{\mathbf{y}} - \Delta(\mathbf{P}, \overline{\mathbf{y}}). \tag{22}$$

Unfortunately, relaxing $\overline{\mathbf{y}}$ to take fractional values in the interval $[0, 1]$ for the above problem does not result in an LP. This is due to the dependence of Δ on the output $\overline{\mathbf{y}}$ in both its numerator and denominator (see equation (16)). We address this issue by adopting a two stage strategy: (i) replace Δ by another loss function that results in an LP relaxation; and (ii) using the solution of the first stage as an accurate initialization, solve problem (22) via iterated conditional modes (ICM). For the first stage, we use the macro-average error over all classes as the loss function. Since macro-average error is is linear in $\overline{\mathbf{y}}$, it can be handled by using our LP relaxation, which allows for fractional values of the variables and introduces clique constraints. In our experiments, ICM converged within very few iterations (typically less than 5) when initialized in this manner. Since both the LP relaxation and the ICM algorithm are not guaranteed to provide the global optimum solution, the subgradients computed by solving problem (22) will be approximate. However, as will be seen in section 5, the approximate subgradients are sufficient to obtain an accurate set of parameters.

4.2 Learning with Bounding Boxes

We now focus on learning specific-class segmentation from training images with user-specified bounding boxes for instances of some classes. To simplify our description, we make the following assumptions: (i) the image contains only one bounding box, which provides us with the location information for an instance of a specific foreground class; and (ii) all the pixels that lie outside the bounding box belong to the generic background class. We note that our approach can be trivially extended to handle cases where the above assumptions do not hold true (for example, bounding boxes for background or multiple boxes per image).

The major obstacle in using bounding box annotations is the lack of a readily available loss function

that compares bounding boxes $\mathbf{B}$ to the model output $\mathbf{Y}$. Note that it would be unwise to use a loss function that compares two bounding boxes (the ground-truth and the predicted one that can be derived from the output), as this function would not be compatible with the overlap score loss used in the previous section. In other words, minimizing such a loss function would not necessarily improve the segmentation accuracy. We address this issue by adopting a simple, yet effective, strategy that solves a series of LSVM problems for generic classes. Our approach consists of three steps:

- Given an image $\mathbf{X}$ and its bounding box annotation $\mathbf{B}$, we infer an output of the model using the current set of parameters (say, the parameters learned using generic class segmentation data). The segmentation is obtained by minimizing an objective function that augments the energy of the model with terms that encourage the segmentation to agree with the bounding box (see details below).
- Using the above output, we define a generic class segmentation $\mathbf{P}$ of the image (see details below).
- Segmentation $\mathbf{P}$ is used to learn the parameters.

The new parameters then update the segmentation. The entire process can be repeated until convergence (that is, until the segmentations stop changing). In our experiments, we stop the process when the accuracy of the validation set stops increasing, which is typically after 2 to 3 iterations. Note that the third step simply involves learning an LSVM as described in the previous section. We describe the first two steps in more detail.

4.2.1 *From Bounding Box to Output.*

We assume that the specified bounding box is tight (a safe assumption for most datasets) and penalize any row and column of the bounding box that is not covered by the corresponding class. Here, a row (column) is said to be covered if a region that overlaps with the row (column) has been selected and assigned the class associated with the given bounding box.

Formally, given a bounding box $\mathbf{B}$ of class c, we define a pixelwise segmentation $\mathbf{P}'$ such that all the pixels p inside the bounding box have no label specified in $\mathbf{P}'$ (denoted by $\mathbf{P}'_p = l_0$) and all the pixels outside the bounding box have a generic background label (that is, $\mathbf{P}'_p = l_b$, which is consistent with our assumption). Furthermore, for each row (column) of $\mathbf{B}$ indexed by q, we define a penalty κ_q, which is imposed if q is not covered. The penalty κ_q has a high value κ_{max} for the center row and center column, and decreases at a linear rate to κ_{min} at the boundary. In our experiments, we set $\kappa_{max} = 10\kappa_{min}$ and cross-validated the value of κ_{min} using a small set of images. We found that our method produced very similar segmentations for a large range of κ_{min}.

Given a dictionary of regions $\mathcal{D}$, we obtain the output of the model by solving the LP relaxation of the following IP:

$$\min_{\overline{\mathbf{y}} \in SELECT(\mathcal{D}), \overline{y}_q \in \{0,1\}} \boldsymbol{\theta}^\top \overline{\mathbf{y}} + \sum_q \kappa_q (1 - \overline{y}_q)$$
$$\text{s.t.} \quad \Delta(\mathbf{P}', \overline{\mathbf{y}}) = 0, \overline{y}_q \leq \sum_{r \in \mathcal{C}(q)} \overline{y}^r_c. \qquad (23)$$

Here, $\mathcal{C}(q)$ is the set of all regions in $\mathcal{D}$ that overlap with q. The variable $\overline{y}_q$ is an indicator that takes the value 1 if q is covered, and 0 otherwise. The loss function Δ is measured over all pixels that lie outside the bounding box, which are assumed to belong to the generic background class. We refer to the above problem as bounding-box inference. In order to obtain an approximate solution for bounding-box inference, we relax the variables to take fractional values and introduce clique constraints. The resulting LP is solved using dual decomposition. Fig. 2(b) shows some example outputs obtained from bounding-box inference, together with the results of segmentation-consistent inference during different iterations of SPL. The quality of the segmentations and the ability of SPL to select good images ensures that our model is trained without noise.

Lempitsky *et al.* [43] have suggested a method to obtain a binary segmentation of an image with a user-specified bounding box. However, our setting differs from theirs in that, unlike the low-level vision model used in [43] (likelihoods from RGB values, contrast dependent penalties), we use a more sophisticated high-level model which encodes information about specific classes and their pairwise relationship using a region-based representation. Hence, we can resort to a much simpler optimization strategy and still obtain accurate segmentations.

4.2.2 *From Output to Segmentation.*

Let $\mathbf{Y}^B$ denote the output obtained from bounding-box inference. Using $\mathbf{Y}^B$ we define a pixelwise segmentation $\mathbf{P}$ as follows. For each pixel p inside the bounding box that was labeled as class c, that is, $\mathbf{Y}^B_p = c$, we define $\mathbf{P}_p = c$. For pixels p inside the bounding box such that $\mathbf{Y}^B_p \neq c$, we define $\mathbf{P}_p = l_0$. In other words, during segmentation-consistent inference these pixels can belong to any class, foreground or background. The reason for specifying $\mathbf{P}_p$ in this manner is that, while we are fairly certain that the pixels labeled $\mathbf{Y}^B_p = c$ do belong to the class c, due to the lack of information in the annotation we are not sure of which class the other pixels belong to. Not labeling such pixels prevents using the mistakes made in bounding-box inference to learn the parameters. Finally, for all the pixels outside the bounding box we set $\mathbf{P}_p = l_b$, that is, they are labeled as generic background. The segmentation $\mathbf{P}$ can then be used to update the parameters by solving an LSVM as described in the previous subsection.

4.3 Learning with Image-Level Labels

We use a similar three step process to the one described above for bounding boxes in order to take advantage of the numerous images with image-level labels, which indicate the presence of a class. In more detail, given an image containing class c, we define a segmentation $\mathbf{P}'$ that does not specify a label for any pixel of the image (that is, $\mathbf{P}'_p = 0$ for all l_p).

Given a dictionary $\mathcal{D}$, we obtain an output by solving the LP relaxation of the following IP:

$$\min_{\overline{\mathbf{y}} \in SELECT(\mathcal{D}), \overline{y} \in \{0,1\}} \boldsymbol{\theta}^\top \overline{\mathbf{y}} - \kappa_{max} \overline{y}, \text{ s.t. } \overline{y} \leq \sum_{r \in \mathcal{D}} \overline{y}^r_c. \quad (24)$$

The indicator variable $\overline{y}$ is 1 if at least one selected region is assigned to the class c, and 0 otherwise. We refer to the above problem as image-level inference. In order to an approximate solution for image-level inference, we relax the variables to take fractional values and introduce clique constraints. The resulting P is solved using dual decomposition.

To obtain a pixelwise segmentation $\mathbf{P}$ from the above output, we label all pixels p belonging to class c as $\mathbf{P}_p = c$. For the rest of the pixels, we define $\mathbf{P}_p = l_0$. The segmentation obtained in this manner are used to refine the parameters and the entire process is repeated until convergence.

5 Experiments

We now demonstrate the efficacy of our energy minimization and parameter estimation approaches using large, publicly available datasets.

5.1 Energy Minimization

We compare our LP relaxation based approach for energy minimization with three baselines: (i) regions obtained by the intersection of multiple over-segmentations of an image; (ii) regions defined by the segments of the best single over-segmentation (in terms of the energy of the model); and (iii) regions selected from a dictionary similar to ours by the method of [15] (using the code provided by the authors). For each method, we use the same set of parameters, which are learned by the closed loop learning (CLL) technique of [15].

5.1.1 Dataset

We use the publicly available Stanford background dataset [15]. It consists of 715 images (collected from standard datasets such as PASCAL, MSRC and geometric context) whose pixels have been labeled as belonging to one of seven background classes or a generic foreground class. For each image we use three over-segmentations obtained by employing different kernels for the standard mean-shift algorithm [9]. Similar to [15], we split the dataset into 572 images for training and 143 images for testing. We report results on four different splits.

	Energy	Accuracy
Pixel	-	76.65 ± 1.20
Intersection	16150 ± 2005	76.84 ± 1.34
Segmentation	6796 ± 833	77.85 ± 1.50
[15]	4815 ± 592	78.52 ± 1.40
Our Method	**1630 ± 306**	**79.42 ± 1.41**

TABLE 2

The mean and standard deviation of the energy and the pixel-wise accuracy over four folds is shown. The first row corresponds to a pixel-based model [15]. The second row uses regions obtained by intersecting the three over-segmentations. The third row shows the results obtained by using the best single over-segmentation (in terms of the energy function). The fourth row corresponds to the method described in [15]. The fifth row shows our method's results. Using multiple over-segmentations to define an accurate dictionary and selecting the regions by our LP relaxation based method results in a lower energy output that provides better accuracy.

5.1.2 Results

We evaluate the accuracy of the output obtained by measuring the percentage of pixels whose labels matched the ground-truth. Here, the label of a pixel is the label assigned to the region to which it belongs. Table 2 shows the average energy and accuracy for the different approaches. Note that all region-based methods outperform the pixel-based approach described in [15] (that provides comparable results to [4]). However, the choice of the regions greatly affects the value of the energy and hence the accuracy of the segmentation. By using large dictionaries and an accurate LP relaxation, our approach provides a statistically significant improvement (using paired t-test with $p = 0.05$) over other methods, both in terms of energy and accuracy. Our algorithm takes less than 10 minutes per image on average on a 2.4 GHz processor. A majority of the computational time (80%) is taken up by dual decomposition, while the rest of the time is required for refining the dictionary. Within dual decomposition, the most computationally expensive slave problem corresponds to the cliques (that is, third type, which takes over 50% of the total time). Note that new optimization frameworks such as alternating direction method of multipliers (ADMM) [44] may allow for further speed-ups over dual decomposition.

5.2 Parameter Estimation

Despite obtaining a substantial improvement in terms of the energy, our LP relaxation approach only results in a small improvement in terms of the segmentation accuracy. This is due to the fact that the parameters learned using CLL are not suited to our inference approach. To alleviate this, we use our LSVM formulation to estimate the parameters using large, publicly available datasets that specify varying levels of annotation for the training images.

	avg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow
CLL	**24.7**	32.1	16.3	9.5	5.9	25.0	43.7	40.2	17.1	4.6	26.6
LSVM	**26.9**	41.4	19.0	9.6	6.9	28.1	45.4	41.4	16.4	5.2	26.0
BOX	**28.3**	43.0	19.9	10.9	8.0	30.8	47.4	39.0	16.5	3.8	26.6
LABELS	**28.8**	44.5	20.9	10.7	7.8	31.5	45.6	40.0	15.7	3.5	28.1
CCCP	**24.7**	39.4	15.6	6.5	6.5	27.7	41.0	40.7	16.3	5.5	29.2
[11]	**34.5**	67.1	26.6	30.3	31.6	30.0	44.5	41.6	25.2	5.9	27.8
[12]	**24.8**	48.3	6.7	19.1	10.0	16.6	32.7	38.1	25.3	5.5	9.4
[14]	**36.3**	64.3	21.8	21.7	32.0	40.2	57.3	49.4	38.8	5.2	28.5

	d-table	dog	horse	m-bike	person	plant	sheep	sofa	train	tv	bg
CLL	16.8	10.3	32.7	31.3	35.7	8.9	27.7	11.1	29.7	23.6	69.6
LSVM	18.7	10.3	34.4	33.5	37.3	7.6	32.7	10.7	34.1	31.3	75.2
BOX	16.9	11.4	36.9	36.2	41.2	7.7	37.4	11.5	35.8	33.8	79.4
LABELS	16.0	12.3	37.7	40.4	42.2	8.3	35.6	12.9	35.8	34.6	80.0
CCCP	12.8	8.5	31.4	39.6	32.7	8.2	30.1	12.3	27.9	18.5	67.9
[11]	11.0	23.1	40.5	53.2	32.0	22.2	37.4	23.6	40.3	30.2	80.2
[12]	25.1	13.3	12.3	35.5	20.7	13.4	17.1	18.4	37.5	36.4	79.6
[14]	22.0	19.6	33.6	45.5	33.6	27.3	40.4	18.1	33.6	46.1	83.9

TABLE 3

Accuracies for the VOC2009 *test set. First row shows the results obtained using* CLL *[15] with a combination of* VOC2009 *and* SBD *training images. The second row shows the results obtained using* SPL *for* LSVM *with the same set of the training images. The third row shows the results obtained using an additional 1564 bounding box annotations. The fourth row shows the results obtained by further augmenting the training dataset with 1000 image-level annotations. The best accuracy for each class is underlined. The fifth row shows the results obtained when the* LSVM *is learned using* CCCP *on the entire dataset. The last three rows show the results obtained by other models. Note that, in contrast to our approach, these models are only concerned with segmentation using the label set $\mathcal{F}'$, and are therefore trained using a different objective function.*

	avg	sky	tree	road	grass	water	bldg	mntn	fg
CLL	**53.1**	77.7	48.4	70.1	73.5	55.6	62.5	0.0	36.0
LSVM	**54.3**	79.1	48.2	75.5	76.0	55.1	61.4	0.0	39.1
BOX	**54.8**	78.3	48.6	75.4	76.0	59.9	60.8	0.0	39.6
LABELS	**55.3**	78.1	49.5	75.5	76.1	60.1	62.0	0.0	41.3
CCCP	**53.8**	75.4	48.7	70.0	74.0	59.9	62.5	0.0	39.9

TABLE 4

Accuracies for the SBD *test set. See caption of Table 3 for an explanation of the various methods.*

5.2.1 Generic Class Segmentations

Comparison. We show the advantage of the LSVM formulation over CLL, which was specially designed for the region-based model, for the problem of learning a specific-class segmentation model using generic class segmentations.

Datasets. We use two datasets: (i) the VOC2009 segmentation dataset, which provides us with segmentations consisting of 20 specific foreground classes and a generic background; and (ii) SBD, which provides us with segmentations consisting of 7 specific background classes and a generic foreground. Thus, we consider 27 specific classes, which results in a harder learning problem compared to methods that use only the VOC2009 segmentation dataset or SBD. The total size of the dataset is 1846 training (1274 from VOC2009 and 572 from SBD), 278 validation (225 from VOC2009 and 53 from SBD) and 840 test (750 from VOC2009 and 90 from SBD) images. For CLL, the validation set is used to learn the pairwise potentials and several hyper-parameters while for LSVM it is used for early stopping (see section 4.1).

Results. Tables 3 and 4 (rows 1 and 2) show the accuracies obtained for SBD and VOC2009 test images respectively. Since we used the overlap loss during training, we measure the accuracies using the overlap score, that is, $1-\Delta(\mathbf{P}, \hat{\mathbf{Y}})$, where $\mathbf{P}$ is the ground-truth segmentation and $\hat{\mathbf{Y}}$ is the predicted output of the model. While both CLL and LSVM produce specific-class segmentations of all the test images, we use generic classes while measuring the performance due to the lack of specific-class ground-truth segmentations. For each method, we used 30 cores of 2.4 GHz processors. The CLL method is significantly faster and takes less than one day to finish training, while the LSVM converges after 2.5 days of training. However, in terms of accuracy, LSVM outperforms CLL on nearly all the object classes in VOC2009 (17 of 21 classes). For SBD, LSVM provides a significant boost in performance for 'sky', 'road', 'grass' and 'foreground'. With the exception of 'building', the accuracies for other classes is comparable. The reason for poor performance in the 'mountain' class is that several 'mountain' pixels are labeled as 'tree' in SBD (which confuses both the learning algorithms). Our results convincingly demonstrate the advantage of using LSVM.

5.2.2 Bounding Box Annotations

Comparison. We now compare the model learned using only generic class segmentations with the model that is learned by also considering bounding box

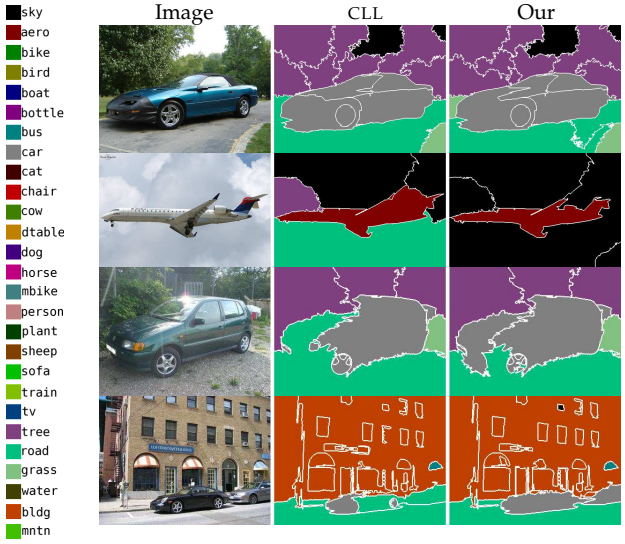

Fig. 3. *The first two rows show the results obtained for images from the* VOC2009 *test set. Note that, unlike our approach that learns the parameters using* LSVM *on a large dataset,* CLL *mislabels background pixels into the wrong specific classes. The last two rows show images from the* SBD *test set. While our approach is able to identify most of the foreground pixels correctly,* CLL *mislabels them as background.*

annotations. In keeping with the spirit of SPL, we use the previous LSVM model (learned using *easier* examples) as initialization for learning with additional bounding boxes.

Datasets. In addition to VOC2009 and SBD, we use some of the bounding box annotations that were introduced in the VOC2010 detection dataset. Our criteria for choosing the images is that (i) they were not present in the VOC2009 detection dataset (which were used to obtain detection-based image features for the model); and (ii) none of their bounding boxes overlapped with each other. This provides us with an additional 1564 training images that have previously not been used to learn a segmentation model.

Results. Training the model with the above dataset takes 2.5 days to converge when using 30 cores of 2.4 GHz processors. Tables 3 and 4 (row 3) show the accuracies obtained for VOC2009 and SBD respectively. Once again, we observe an improvement in the accuracies for nearly all the VOC2009 classes (18 of 21 classes) compared to the LSVM trained using only generic class segmentations. For SBD, we obtain a significant boost for 'tree', 'water' and 'foreground', while the accuracies of 'road', 'grass' and 'mountain' remain (almost) unchanged.

5.2.3 Image-Level Annotations

Comparison. We compare the model learned using generic class and bounding box annotations with the model that is learned by also considering image-level labels. Once again, following the idea of SPL closely, we use the model learned in the previous subsection as an initialization for learning with the additional image-level labels.

Datasets. In addition to SBD, VOC2009 segmentation dataset and VOC2010 detection dataset, we use a subset of the ImageNet [39] dataset that provides tags indicating the presence of an object class in an image. Due to time limitations, we restrict ourselves to 1000 randomly chosen images from ImageNet. Our only criterion for choosing the images was that they must contain at least 1 of the 20 foreground classes from the VOC datasets.

Results. Training the model with the above dataset takes 2 days to converge. Once again, we used 30 cores of 2.4 GHz processors. Tables 3 and 4 (row 4) show the accuracies obtained for VOC2009 and SBD respectively. For the VOC2009 segmentation test set, the final model learned from all the training images provides the best accuracy for 12 of the 21 classes. Compared to the model learned using generic class labels and bounding boxes, we obtain a significant improvement for 13 classes by incorporating image-level annotations. Of the remaining 8 classes, the accuracies are comparable for 'bird', 'boat', 'chair' and 'train'. For the SBD test set, the model trained using all the data obtains the highest accuracy for 5 of the 8 classes. Fig. 3 shows examples of the specific-class segmentation obtained using our method. Note that the parameters learned using our approach on a large dataset are able to correctly identify the specific classes of pixels.

5.2.4 SPL vs. CCCP

Comparison. While it is not the central hypothesis of the paper, we also tested the CCCP algorithm to determine whether there is any benefit to using the SPL training method.

Datasets. We use the entire dataset consisting of strongly supervised images from the SBD and VOC2009 segmentation datasets and weakly supervised images from the ImageNet and VOC2010 detection datasets.

Results. The CCCP algorithm is significantly faster than SPL as it takes only 3 days in total to converge. However, SPL outperforms CCCP in terms of the accuracy of the segmentations (tables 3 and 4, row 5). Specifically, CCCP does not provide any improvement over CLL, which is trained using only the strongly supervised images, in terms the average overlap score for VOC2009 . While the overlap score improves for the SBD dataset, the improvement is significantly better when using SPL (row 4). These results convincingly demonstrate that, unlike CCCP, SPL is able to handle the noise inherent in the problem of learning with diverse data.

6 Discussion

In order to aid the use of the region-based model [15], we proposed an energy minimization algorithm that avoids the bad local minima, and a principled LSVM parameter estimation framework using diverse data.

Our energy minimization algorithm can be used for other applications such as object discovery. Our parameter estimation approach can be adapted to handle other types of annotations.

A limitation of our work is the amount of time required to train and test the model. In particular, during training, a large number of stochastic subgradient descent iterations are required to update the parameters. During testing, dual decomposition also runs for several iterations. These problem may be resolved using ADMM [44].

Acknowledgments: This work is partially funded by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013)/ERC Grant agreement number 259112, INRIA-Stanford associate team SPLENDID, NSF grant IIS 0917151, MURI contract N000140710747, and the Boeing company.

REFERENCES

[1] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparitive study of energy minimization methods for Markov random fields with smoothness-based priors," *PAMI*, 2008.
[2] X. He, R. Zemel, and M. Carriera-Perpinan, "Multiscale conditional random fields for image labeling," in *CVPR*, 2004.
[3] S. Konishi and A. Yuille, "Statistical cues for domain specific image segmentation with performance analysis," in *CVPR*, 2000.
[4] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," in *ECCV*, 2006.
[5] D. Larlus and F. Jurie, "Combining appearance models and Markov random fields for category level object segmentations," in *CVPR*, 2008.
[6] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie, "Objects in context," in *ICCV*, 2007.
[7] A. Saxena, M. Sun, and A. Ng, "Make3D: Learning 3D scene structure from a single still image," *PAMI*, 2008.
[8] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in *CVPR*, 2009.
[9] D. Comaniciu and P. Meer, "Mean shift analysis and applications," in *ICCV*, 1997.
[10] C. Pantofaru, C. Schmid, and M. Hebert, "Object recognition by integrating multiple segmentations," in *ECCV*, 2008.
[11] J. Gonfaus, X. Boix, J. Van de Weijer, A. Bagdanov, J. Serrat, and J. Gonzalez, "Harmony potentials for joint classification and segmentation," in *CVPR*, 2010.
[12] P. Kohli, L. Ladicky, and P. Torr, "Robust higher order potentials for enforcing label consistency," in *CVPR*, 2008.
[13] L. Ladicky, C. Russell, P. Kohli, and P. Torr, "Associative hierarchical CRFs for object class image segmentation," in *ICCV*, 2009.
[14] F. Li, J. Carreira, and C. Sminchisescu, "Object recognition as ranking holistic figure-ground hypotheses," in *CVPR*, 2010.
[15] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in *ICCV*, 2009.
[16] D. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.
[17] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *IJCV*, 2010.
[18] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *CVPR*, 2008.
[19] C.-N. Yu and T. Joachims, "Learning structural SVMs with latent variables," in *ICML*, 2009.
[20] M. P. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *NIPS*, 2010.
[21] M. P. Kumar and D. Koller, "Efficiently selecting regions for scene understanding," in *CVPR*, 2010.
[22] M. P. Kumar, H. Turki, D. Preston, and D. Koller, "Learning specific-class segmentation from diverse data," in *ICCV*, 2011.
[23] S. Nowozin and C. Lampert, "Global connectivity potentials for random field models," in *CVPR*, 2009.
[24] S. Vicente, V. Kolmogorov, and C. Rother, "Graph cut based image segmentation with connectivity priors," in *CVPR*, 2008.
[25] C. Chekuri, S. Khanna, J. Naor, and L. Zosin, "A linear programming formulation and approximation algorithms for the metric labelling problem," *Disc. Math.*, 2005.
[26] M. Wainwright, T. Jaakkola, and A. Willsky, "MAP estimation via agreement on trees: Message passing and linear programming," *IEEE Info. Theory*, 2005.
[27] P. Hammer, "Some network flow problems solved with pseudo-Boolean programming," *Operations Research*, 1965.
[28] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer, "Optimizing binary MRFs via extended roof duality," in *CVPR*, 2007.
[29] E. Boros and P. Hammer, "Pseudo-Boolean optimization," *Discrete Applied Mathematics*, 2002.
[30] N. Komodakis and N. Paragios, "Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles," in *ECCV*, 2008.
[31] M. P. Kumar and P. Torr, "Efficiently solving convex relaxations for MAP estimation," in *ICML*, 2008.
[32] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss, "Tightening LP relaxations for MAP using message passing," in *UAI*, 2008.
[33] F. Barahona and A. Mahjoub, "On the cut polytope," *Mathematical Programming*, 1986.
[34] N. Komodakis, N. Paragios, and G. Tziritas, "MRF optimization via dual decomposition: Message-passing revisited," in *ICCV*, 2007.
[35] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kauffman, 1998.
[36] N. Komodakis and N. Paragios, "Beyond pairwise energies: Efficient optmization for higher-order MRFs," in *CVPR*, 2009.
[37] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods*. MIT Press, 1999.
[38] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *PAMI*, 2001.
[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarhical image database," in *CVPR*, 2009.
[40] S. Shalev-Shwartz, Y. Singer, and N. Srebro, "Pegasos: Primal estimated sub-gradient solver for SVM," in *ICML*, 2009.
[41] T. Finley and T. Joachims, "Training structural SVMs when exact inference is intractable," in *ICML*, 2008.
[42] H. Wang, S. Gould, and D. Koller, "Discriminative learning with latent variables for cluttered indoor scene understanding," in *ECCV*, 2010.
[43] V. Lempitsky, P. Kohli, C. Rother, and T. Sharp, "Image segmentation with a bounding box prior," in *ICCV*, 2009.
[44] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, 2011.

M. Pawan Kumar is an Assistant Professor at Ecole Centrale Paris. He completed his PhD in 2008 at Oxford Brookes University. He was a postdoctoral researcher at the University of Oxford in 2008 and at Stanford University from 2009 to 2011.

Haithem Turki is an Engineer at Palantir Technologies. He completed his undergraduate studies at Stanford University in 2012 with a major in Artificial Intelligence and a minor in Management Science and Engineering.

Dan Preston is the CEO of MetroMile. Previously, he was the CTO of AisleBuyer LLC. He completed his undergraduate studies at Brandeis University in 2007, and his masters studies at Stanford University in 2012.

Daphne Koller is the CEO and co-founder of Coursera. Until 2014, she was the Rajeev Motwani Professor at Stanford University. She completed her PhD at Stanford University in 1993, and was a postdoctoral researcher at the Computer Science Division of UC Berkeley. She did her masters and undergraduate degrees at the Hebrew University of Jerusalem, Israel.