Kernelized Structural SVM Learning for Supervised Object Segmentation

Luca Bertelli† Tianli Yu† Diem Vu Burak Gokturk
Google, Inc.
{lbertelli, tianliyu, diemvu, burakg}@google.com

Abstract

Object segmentation needs to be driven by top-down knowledge to produce semantically meaningful results. In this paper, we propose a supervised segmentation approach that tightly integrates object-level top down information with low-level image cues. The information from the two levels is fused under a kernelized structural SVM learning framework. We defined a novel nonlinear kernel for comparing two image-segmentation masks. This kernel combines four different kernels: the object similarity kernel, the object shape kernel, the per-image color distribution kernel, and the global color distribution kernel. Our experiments show that the structured SVM algorithm finds bad segmentations of the training examples given the current scoring function and punishes these bad segmentations to lower scores than the example (good) segmentations. The result is a segmentation algorithm that not only knows what good segmentations are, but also learns potential segmentation mistakes and tries to avoid them. Our proposed approach can obtain comparable performance to other state-of-the-art top-down driven segmentation approaches yet is flexible enough to be applied to widely different domains.

1. Introduction

Object recognition and segmentation are two interdependent processes where solving one can help the other. From the point view of segmentation, pure bottom up approaches such as mean shift or normalized cuts can only segment objects that are highly salient with respect to the background. Many interactive segmentation algorithms rely on human input to recognize and localize the object from a complex background. To create a fully automatic segmentation algorithm, high level prior knowledge needs to be infused into the segmentation process, either through customarily built models or in some cases models learned from training examples. However, these models are usually created in a domain specific fashion that is difficult to generalize. In this paper, we propose a supervised learning approach for segmentation, where the only domain specific knowledge required is to select a specific image descriptor. With that and a set of training segmentations, our algorithm will automatically learn a discriminative model to segment a new image, without any user intervention. The main contributions of this paper are:

† denotes equal contribution.

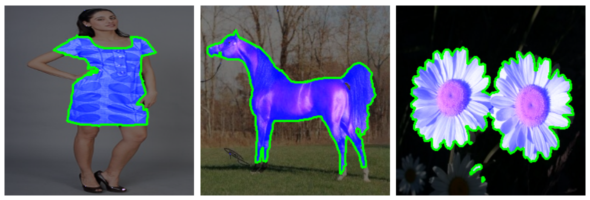
Figure 1. Segmentations produced by our learning algorithm. Our algorithm can be applied to different image domains with minimal change once a segmentation training set is provided. This flexibility is made possible by using a novel kernel that fuses top-down knowledge and low-level image cues in the structured SVM learning framework.

- We propose a kernelized structural support vector machine approach to learn discriminatively the mapping from image to a segmentation mask.
- We combine high level object similarity information (via image descriptors) with multiple low level segmentation cues into a novel kernel function used in the structured SVM framework.
- Traditional segmentation regularizations, such as the pair-wise smoothness, are preserved in our framework and explicitly enforced during the learning process. This way smoothness of the solution does not need to be "re-learned" from training examples.

Our work is built upon several important developments in computer vision: the availability of a large collections of feature descriptors, the use of large margin classifiers (e.g. SVMs) and the integration of high level prior knowledge into Markov Random Fields. Specifically, our paper is closely related to other works in these three directions:

Top Down Driven Image Segmentation Segmentation can be driven by the top-down knowledge coming from object detection [30, 31] or prior knowledge about the objects is exploited in part-based modeling to influence a bottom-up grouping process [12, 28]. Our algorithm can be considered more similar to approaches that iteratively try to accommodate bottom-up and top-down cues [16, 25, 15]. [16] learns a set of common fragments that provides a local bias to drive the Conditional Random Field (CRF) solution to reduce segmentation errors on the training set. However, the learned fragments are applied to all the training/testing images, so it works well only when the objects are all from the same class and are at roughly the same scale. In contrast, our approach becomes more flexible by incorporating an object similarity kernel that can "select" relevant training examples during training/testing. Another major difference is that structured SVM is a descriminative learner, which incorporates both positive and negative information.

Structured SVM and Max Margin MRFs Structured SVMs have found their applications in many machine learning scenarios, ranging from Natural Language Processing [22] to DNA sequence alignment [29] and to segmentation of 3D scan data [2]. Also, recently they have been used in connection with segmentation methods based on MFRs and CRFs. The energy functions that model probability distributions on these fields usually contain parameters that need to be estimated. In [21] or [18] the authors show how structured SVMs can be used to learn these parameters in a principled manner, an alternative to the pseudo-likelihood formulation used in [13], or more traditionally used cross validation and piecewise training. The main difference of our approach with previous work in structured SVMs for segmentation, consists in combining recognition with segmentation by training on a set of image-mask pairs via a kernel function that incorporates object similarity information (In [21, 18, 2] models are trained using single images, to essentially learn MRFs parameters). In addition, by using non-linear kernels ([21, 18, 2] all used linear models), we enforce a top-down knowledge that encodes complex relationships between couples of image/mask pairs.

Object Class Recognition Object recognition has experienced tremendous growth in the past 10 years, especially with the creation of the Pascal Challenge [1]. There are now many different feature descriptors available (SIFT, HOG[6], Self-similarity, *etc*.) that capture the appearance of objects in a compact, fixed-length vector form. Our approach can integrate any (or multiple) of these descriptors in our object similarity kernel. In one of the experiments, we show that even the response of part-based object detectors can be used as a feature descriptor for our object similarity kernel.

The paper is organized as follows: Sections 2 and 3 formulate the segmentation problem under the Structured SVM framework. Section 4 provides further implementation details. In Section 5 we present extensive experimental evaluation on three datasets from different domains, while we conclude and discuss future directions in Section 6.

2. Segmentation via Structured SVM Learning

We cast image segmentation as learning a *prediction* function $f : \mathcal{X} \mapsto \mathcal{Y}$ that maps the space of images $\mathcal{X}$ to the space of binary segmentation masks $\mathcal{Y}$, based on a training set of input/output pairs. Structured SVMs [23, 9] provide a framework to tackle this problem and make it tractable despite the exponential number of possible constraints.

The idea behind Structured SVM is to discriminatively learn a *scoring* function $F : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ over input/output pairs (*i.e*. over image/mask pairs). Once this function is learned, the *prediction* function f can be obtained by maximizing F over all possible $y \in \mathcal{Y}$, for a given image input:

$$y^* = f(x) = \operatorname*{argmax}_{y \in \mathcal{Y}} F(x, y) \tag{1}$$

In analogy to linear SVM, the scoring function is expressed as a linear combination of a set of basis functions $\mathbf{\Psi}(x, y)$:

$$F(x, y; \mathbf{w}) = \mathbf{w}^T \mathbf{\Psi}(x, y) \tag{2}$$

The weight vector $\mathbf{w}$ is learned through the minimization of a constrained quadratic optimization problem [9]:

$$\begin{aligned} &\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \mathbf{w}^T \mathbf{w} + C\xi \\ &s.t.\ \forall (\bar{y}_1, \ldots, \bar{y}_n) \in \mathcal{Y}^n : \\ &\frac{1}{n} \mathbf{w}^T \sum_{i=1}^{n} [\mathbf{\Psi}(x_i, y_i) - \mathbf{\Psi}(x_i, \bar{y}_i)] \geq \frac{1}{n} \sum_{i=1}^{n} \Delta(y_i, \bar{y}_i) - \xi \end{aligned} \tag{3}$$

where $\Delta(y_i, \bar{y}_i)$ is a function which measures the loss of a mask $\bar{y}_i$ if the expected mask is y_i. Intuitively, the constraints in (3) requires that for a training example pair (x_i, y_i), $F(x, y)$ has to produce a score that is higher than the score of any other pair $(x_i, \bar{y}_i)$ by at least $\Delta(y_i, \bar{y}_i)$. For segmentation purposes, we use the simple sum of pixel differences as the loss function:

$$\Delta(y_i, \bar{y}_i) = \sum_p I(y_i(p) != \bar{y}_i(p)) \tag{4}$$

where $I(\cdot)$ is an indicator function.

Although there are exponential number of constraints in (3) with respect to the dimensionality of y, [9] shows that the problem can be solved via a cutting plane algorithm, which involves constructing a working constraint set that is built by incrementally adding the most violated constraints. The search for the most violated constraint for the ith training example can be formulated as a maximization problem (similar to solving for the final segmentation):

$$\hat{y}_i = \operatorname*{argmax}_{y \in \mathcal{Y}} \Delta(y_i, y) + \mathbf{w}^T \mathbf{\Psi}(x_i, y) \tag{5}$$

which in our case can be solved via Graph Cuts as shown in Sec.4.1. The learning stops when the objective function is within a certain accuracy bound.

To adapt the Structured SVM framework to solve the segmentation learning problem, we make two important choices. First, instead of letting y vary in the entire space $\mathcal{Y}$ during the search of the argmax in (5) and (1), we restrict the search to $\mathcal{Y}_s$, subset of $\mathcal{Y}$ composed by smooth segmentation masks (we assume that noisy/unsmooth solutions are not correct, hence can be excluded from the search space). This can be accomplished by appending a regularization term to the target function, enforcing smoothness of the solution. The resulting maximization problems become:

$$y^* = f(x) = \underset{y\in\mathcal{Y}}{\operatorname{argmax}}\, F(x,y) + \Theta(x,y) \tag{6}$$

$$\hat{y}_i = \underset{y\in\mathcal{Y}}{\operatorname{argmax}}\, \Delta(y_i,y) + \mathbf{w}^T\mathbf{\Psi}(x_i,y) + \Theta(x,y) \tag{7}$$

The choice of treating regularization as a separate term (Θ defined in (15)), not explicit part of the learning model, simplifies the learning task a great deal, since it allows to focus learning on other aspects of segmentation rather than smoothness. Structured SVM does not have to learn that solutions are smooth, since it is already imposed in the model.

The other choice, that makes our work different from most of the prior work, consists in using kernel functions so that we could work in the dual formulation. The advantage is that an explicit expression for the feature vector $\mathbf{\Psi}(x,y)$ (which could be very complex) is not required, since everything can be written in terms of Mercer kernels:

$$K\big((x_i,y_i),(x_j,y_j)\big) = \langle \mathbf{\Psi}(x_i,y_i), \mathbf{\Psi}(x_j,y_j)\rangle^{1} \tag{8}$$

These kernels are easier to describe analytically, since they express the relationship between two image-mask pairs. The solution of the dual problem gives a set of weights α^* for the support vectors. The final scoring function $F(x,y)$ can be written as:

$$F(x,y) = \mathbf{w}^{*T}\mathbf{\Psi}(x,y) = \tag{9}$$
$$\sum_{\bar{\mathbf{y}}\in\mathcal{W}} \alpha^*_{\bar{\mathbf{y}}}\left(\frac{1}{n}\sum_{i=1}^{n}\left[K\big((x,y),(x_i,y_i)\big) - K\big((x,y),(x_i,\bar{y}_i)\big)\right]\right)$$

where $\bar{\mathbf{y}} = (\bar{y}_1,\ldots,\bar{y}_n)$ are the most violated constraints found by Structured SVM learning. This scoring function will be used in (6) to obtain the segmentation mask of x.

3. Designing a Kernel for Segmentation

One of the main contributions of this work is the design of the kernel functions in (8). In contrast with most of the work in structured SVM for segmentation, where the feature vector $\mathbf{\Psi}$ is directly modeled (see for instance [21, 18, 2]) akin conventional linear SVMs, we chose to instead model the kernel function that encodes the mutual agreement between two image-mask pairs. This choice has two main advantages: as in non-linear SVMs it does not require the explicit knowledge of the non-linear mapping to higher dimensional space and it also allows to encode complex relationships between two training samples (image-mask pairs). In particular the kernel will evaluate the quality of the mutual match between image-mask pairs: if the images are similar, the masks have to be similar as well. In case either the images are significantly different or the masks are not matching, the kernel response has to be low. To satisfy these general conditions, we choose our kernel to be the product of the image kernel and mask kernel.

$$K\big((x_i,y_i),(x_j,y_j)\big) = \Lambda(x_i,x_j)\cdot\Omega(y_i,y_j;x_i,x_j) \tag{10}$$

where $\Lambda(x_i,x_j)$ measures the image/object similarity and $\Omega(y_i,y_j)$ measures the mask similarity. [23] proves that such multiplicative kernel is equivalent in the primal to the tensor product of the feature spaces produced by each individual kernel. [8] uses a similar multiplicative kernel between input and output variables, although their output kernel does not depend on the input.

3.1. Object Similarity Kernel

The object similarity kernel $\Lambda(x_1,x_2)$ is used in our framework to modulate the other kernels, which are comparing two image-mask pairs. The presence of this term essentially selects only pairs for which there is a good similarity measure between the images/objects. At classification time, this fact has an important effect: the testing sample is essentially compared with the support vectors for which this similarity measure is sufficiently high. We constructed Λ as a gaussian kernel over the distance between feature vectors computed from the two images:

$$\Lambda(x_i,x_j) = \exp\left(-\frac{||\phi(x_i)-\phi(x_j)||^2}{2\sigma^2}\right) \tag{11}$$

where $\phi:\mathcal{X}\mapsto\mathbb{R}^n$ is a n-dimensional feature vector computed form the image content. In this paper we considered two different types of descriptors. The first one is of general applicability and consists of computing HOG features [6] at fixed grid locations. The second one, of higher discriminative power but object-class specific, uses the deformable parts model object detector in [7] and forms a descriptor using the coordinates of the bounding boxes correspondent to the different parts. The details can be found in Section 4.3.

3.2. Mask Similarity Kernel

The mask similarity kernel is designed to be a linear combination of several individual kernels, each one capturing a specific segmentation feature:

$$\Omega(y_i,y_j;x_i,x_j) = \sum_{l}^{L}\beta_l\Omega_l(y_i,y_j;x_i,x_j) \tag{12}$$

[1]In our specific case this is a generalized inner product between two different spaces that yields an asymmetric kernel. More details in Sec. 3.

In this paper we use three different mask similarity kernels.

Shape Kernel. The shape kernel, measures the degree of agreement of the two masks. Denoting with y_{ip} the value of the mask y_i at position p and assuming y_{ip} a binary random variable we can then write:

$$\Omega_S = \frac{1}{P}\sum_{p=1}^{P} \left(y_{ip}y_{jp} + (1-y_{ip})(1-y_{jp})\right)$$

where P is the total number of pixels in the image.

Local Color Model Kernel. The Local Model kernel, tries to measure how well mask i fits to image i via a foreground/background color model built using image i and mask j. Before defining the expression of the kernel, we need to introduce a quantizer function $h(x_p) : \mathbb{R}^M \mapsto \{0,1\}^Q$ that maps feature points to binary indicator vectors (where M is the dimension of the feature space and Q is the number of quantization cells). We also introduce:

$$\mathcal{F}_i^j = \frac{\sum_{p=1}^{P} h(x_{ip})y_{jp}}{\sum_{p=1}^{P} y_{jp}} \tag{13}$$

as the histogram of the foreground computed on image x_i using mask y_j. Similarly the background histogram is:

$$\mathcal{B}_i^j = \frac{\sum_{p=1}^{P} h(x_{ip})(1-y_{jp})}{\sum_{p=1}^{P}(1-y_{jp})} \tag{14}$$

These two models are built using the features from image x_i and the mask information from mask y_j. We can then write the kernel expression as:

$$\Omega_{LM} = \frac{1}{P}\sum_{p=1}^{P}\left(\mathcal{F}_i^{jT} h(x_{ip})y_{ip} + \mathcal{B}_i^{jT} h(x_{ip})(1-y_{ip})\right)$$

Note that this kernel is not symmetric, *i.e.* $K((x_i,y_i),(x_j,y_j)) \neq K((x_j,y_j),(x_i,y_i))$. Recently non-symmetric kernels have gained popularity since they allow to compare data pairs from two different input spaces (*i.e.* query to documents in in web ranking or items to users in recommendation engines). This significantly simplifies the inference problem as explained in Section 4.1. In [24, 27] the authors show that learning via asymmetric kernels can be treated in the same way as symmetric kernels, provided that the kernel is positive definite [2]. In other words, the SVM optimization retains its convexity property. In our specific scenario, we notice that the final kernel (consider for now only the contribution of the local color model) is given by $K = \Lambda\Omega_{LM}$. We can easily show that, by choosing σ in (11) small enough the matrix K becomes diagonally dominant (and therefore positive definite since the entries on the diagonal are positive). Choosing a small σ also benefits the learning process itself, since the matching by similarity becomes stricter and hence features like shape become more meaningful.

Global Color Model Kernel. This kernel is constructed in similar fashion to Ω_{LM}, but measures how well each image-mask pair fits a global color model built using all training samples. Denoting with $\mathcal{F}_G$ and $\mathcal{B}_G$ global foreground and background color histograms, we can write:

$$\Omega_{GM} = \left[\tfrac{1}{P}\sum_{p=1}^{P}\left(\mathcal{F}_G^T h(x_{ip})y_{ip} + \mathcal{B}_G^T h(x_{ip})(1-y_{ip})\right)\right] \cdot \left[\tfrac{1}{P}\sum_{p=1}^{P}\left(\mathcal{F}_G^T h(x_{jp})y_{jp} + \mathcal{B}_G^T h(x_{jp})(1-y_{jp})\right)\right]$$

Differently from the local color model kernel, we make Ω_{GM} symmetric by multiplying the terms from example i and example j. This won't affect the complexity of the inference because neither $\mathcal{F}_G$ nor $\mathcal{B}_G$ depend on y_i or y_j.

4. Implementation Details

4.1. The smoothness regularization term

The mask smoothness term we use in (6) and (5) is commonly used in many MRF based segmentation algorithms:

$$\Theta(x,y) = \mu\frac{1}{P}\sum_{p=1}^{P}\sum_{q\in\mathcal{N}_p} B(y_p,y_q|x) \tag{15}$$

Here p and q are pixels in the image, $\mathcal{N}_p$ is a neighborhood of pixel p, μ is a constant weight, $B(y_p,y_q|x)$ is a standard image-dependent binary potential:

$$B(y_p,y_q|x) = \frac{|y_p-y_q|}{d_{pq}}\exp\left(-\frac{(x_p-x_q)^2}{2\sigma_x^2}\right) \tag{16}$$

where with d_{pq} we denoted the euclidean distance between pixel p and pixel q. In our implementation, μ is always kept proportional to the sum of all α^* in (9). After adding this term, both (6) and (5) can be re-conducted as a special case of a generic unary+binary energy function, so that Graph Cuts algorithm can be used to efficiently find the global optimal solution [11, 5]:

$$E = \sum_{p=1}^{P} U(y_p) + \sum_{p=1}^{P}\sum_{q\in\mathcal{N}_p} B(y_p,y_q) \tag{17}$$

The binary term is standard in Graph Cuts based segmentation. The unary is a linear combination of the unary potential coming from different kernels (see (9), (10) and (12)). They all depend linearly on y_{ip} (y_i is the unknown mask, while y_j is the mask corresponding the to support vector). The benefit of designing an asymmetric kernel is now evident. In fact, by making the kernel symmetric, one would introduce a non-linear dependency on y_{ip} (in the denominator of (13) and (14)), which would prevent the use of graph cuts to solve the inference problem.

[2]For an asymmetric matrix $\mathbf{A}$, positive definite means the symmetric counterpart $\mathbf{A} + \mathbf{A}'$ is positive definite

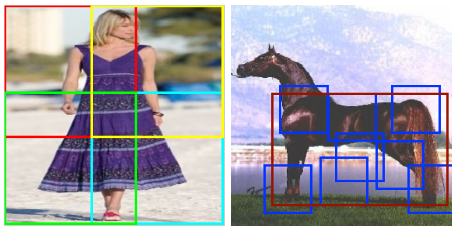

Figure 2. Left: the location of the 2x2 overlapping HOG descriptors. The four HOG descriptors are concatenated to form a 216 dim. HOG Grid feature. Right: Horse detector bounding boxes generated by [7], the coordinates of the 9 bounding boxes are concatenated to create a 36 dim. Detector Response feature.

4.2. Learning the Relative Weights between Kernels

The relative weights β_l between kernels need to be learned as well. In Multiple Kernel Learning methods [3, 14, 26], the kernel weights and the support vector weights are optimized in an alternating fashion, since one can prove that the joint optimization problem retains the convexity property [19]. However, in structured SVM, the objective is to maximize the margin of the scoring function constraints, not the segmentation accuracy. So even if MKL is applied, the result might not translate into improved segmentation performance. This is in contrast with regular SVMs, where the objective of the maximization is directly correlated to the classification accuracy. Throughout our experimental evaluation, we found that a much more effective approach consists in the joint optimization of the parameters over a validation set, via steepest descent minimization of the empirical segmentation loss. The validation set also serves to search the optimal value of other parameters like the C in (3) and σ in (11).

4.3. Object Feature Descriptors

Due to the complexity of the backgrounds in different datasets, it is very important to choose object features that correlate well with the position and pose of the object. We tried two different types of object features: HOG [6] Grid feature and the horse detector response feature (see Fig. 2).

HOG Grid features are HOG descriptor extracted on a fixed 2×2 grid on the image and the 4 descriptors (each has a dimention of 54) are concatenated to form a feature vector. We use the code provided by Pedro Felzenszwalb *et al.* [7] to generate the horse detector response feature. To ensure that we always get a response vector, we ignore the detection threshold and always pick the highest response. The detector returns 9 bounding boxes (1 box for the overall horse location, and 8 boxes for the locations of different parts). We then concatenate the coordinates of these bounding boxes into a vector and normalize them with respect to the size of the image. Note that the detector does not always return an accurate bounding box. This will have less effect on our algorithm since we are not directly using the bounding box to constrain the segmentation. The bounding box is only used as a feature vector for comparison.

4.4. Time Complexity During Training

We are using structured SVM's one slack dual formulation with margin rescaling constraints. One disadvantage of working in the kernel space for structured SVM is that the time complexity in training scales quadratically (comparing to linearly in the linear structured SVM case) with respect to the number of training examples [9]. This could pose a problem for large scale datasets. [9] proposes to use a low rank approximation to the real kernel matrix to reduce the complexity. In our specific case, another possibility is to truncate the object similarity kernel in (11), such that it will return 0 for two images with sufficiently different descriptors. This means that the kernel matrix, as defined in (10), can be made sparse enough to need only a number of kernel evaluations linear in the number of examples.

5. Experimental Results

We trained/tested our segmentation algorithms on three different datasets: the Dresses dataset (from like.com inventory), the Weizmann horses dataset [4], and the Oxford 17 cateogry flower dataset [17]. Each dataset is split into three subsets. Accuracy for each set is computed using the model trained (and tuned by searching the best parameters) on the other two. The accuracies are then averaged across the dataset. The Dresses dataset is composed of 600 images, whose manual segmentations are available. The complexity of this dataset comes from the fact that different type of dresses are also appearing in several different configurations: dresses worn by a model, dresses placed on mannequins or dresses occupying the full image. The goal of the segmentation is to separate the dress from the background, people and mannequins. The Weizmann horse dataset has 328 horse images, with different poses and backgrounds, along with ground truth segmentation masks. The flower dataset consists of 849 flower images and their corresponding trimaps. To ensure the fixed length input/output to the structured SVM framework, all the images are resized to 256×256, regardless of their aspect ratios.

We measured our segmentation performance using two metrics: the overall pixel accuracy S_a and the foreground overlapping ratio S_o. We can write S_a as:

$$S_a = \sum_p I[L(p) = L_{GT}(p)]/A \qquad (18)$$

where A is the area of the image, L and L_{GT} are the forground/background label map. It is a good error metric if both foreground and background are equally important, but it becomes less accurate for datasets whose object of interest has limited area. S_o is defined as the intersection of the

Table 1. Segmentation performance against GrabCut [20] with different initializations.

Dresses Dataset	S_a (%)	S_o (%)
KSSVM Seg + HOGGrid	**93.48**	**82.36**
GrabCut init. with AveMask	83.90	68.47
GrabCut init. with 1-NN mask	88.96	75.52
GrabCut init. with 5-NN masks	90.31	77.74
GrabCut init. with 10-NN masks	90.28	77.44

Horses Dataset	S_a (%)	S_o (%)
KSSVM Seg + Detector Feature	**94.63**	**80.08**
GrabCut init. with Bounding Box	69.53	50.39
GrabCut init. with AveMask	85.52	61.39
GrabCut init. with 1-NN mask	85.66	62.34
GrabCut init. with 5-NN masks	86.93	63.83
GrabCut init. with 10-NN masks	86.46	63.20

result mask and groundtruth mask devided by the union of the two masks.

$$S_o = A(M \cap M_{GT})/A(M \cup M_{GT}) \qquad (19)$$

where M is the mask in the input image, while M_{GT} is the mask in the ground truth. It measures more directly the quality of foreground masks.

The first experiment we present consists in a validation of the proposed algorithm against a baseline to illustrate the performance gain introduced by the learning algorithm. Since, after the structural learning process, inference is solved using graph cuts with a unary potential learned via examples, we compared, as a baseline, the accuracy of our algorithm with segmentation via graph cuts but with a different unary potential. We use OpenCV 2.1 GrabCut code [20] and initialize its unary potential in two different ways: averaging all the mask in the training set or averaging just the masks of the K nearest neighbors in the training set. For the horses dataset, we used the output of the object detector to initialize GrabCut with the bounding box provided by the detector. The results, presented in Table 1, clearly demonstrate the increased accuracy in the segmentation using the structural learning algorithm described in this paper.

Figure 3 shows how the structured SVM finds bad segmentations of the training examples given the kernel based scoring function. These segmentations, or most violated constraints, are then punished during the learning process, while segmentations similar to the ground truth masks are promoted. In Figure 3 we can see how these bad segmentation carry some meaning: one (third column) is always the inverse of the ground truth mask (second column), while the others (columns 4 to 6) highlight potentially bad way of segmenting the input image (*i.e.* including heads or legs, or segmenting out part of the background).

We also present some qualitative results from the Dresses dataset in Figure 4. Notice that, despite the complex backgrounds and the presence of distractors (such as other people besides the main model), the algorithm is able to reliably separate the dress of interest from the background and from other parts of the model's body, such as arms, legs and head.

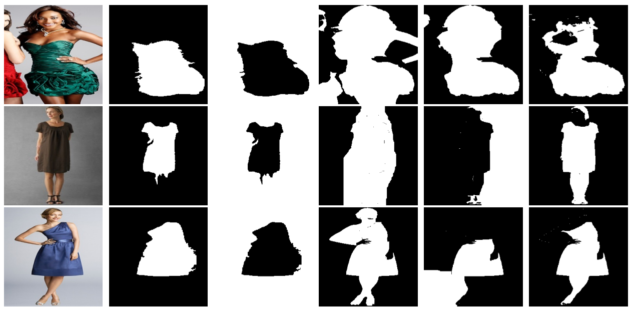
Figure 3. Column 1: images from the Dresses dataset. Column 2: ground truth masks. Columns 3 to 6: examples of most violated constraints (*i.e.* bad segmentations) learned by the algorithm.

Table 2. Segmentation performance comparison with other state of the art algorithms on the horse dataset. †Obj Cut only reported accuracy on a small fraction of the horse dataset. ‡Cosegmentation (reported accuracy on multiple 30 image sets) is a weakly supervised algorithm, which does not need segmentation masks in training, hence lower performance is expected.

	S_a(%)	S_o(%)
KSSVM Seg + HOGGrid	93.9	77.9
KSSVM Seg + Detector Feature	94.6	80.1
Levin & Weiss [16]	95.5	N/A
Obj Cut† [12]	96.0	N/A
Cosegmentation‡ [10]	80.1	N/A

Table 2 summarizes the segmentation metrics we obtained with different image feature descriptors and also the comparison with some of the state-of-the-art results on the horse dataset. Note that in [16] , the authors assume that a prefect bounding box detection of the horse is available and use that to scale and crop the images so that all horses are in the center and with the same scale. In fact, even the state of the art object detector will not give consistent bounding boxes. We get comparable accuracy on the original horse dataset, without any cropping or scaling. To further visualize the contributions of each individual kernel, we plot the foreground potential computed from each kernel on several test images in Figure 5. We can see the effect of the image similarity kernel on the shape potential, which tends to draw information from similar shapes in the training examples.Also, in many cases the three kernels complement each other to generate the final segmentation mask.

Figure 6 lists a few failure cases, the horse's legs are usually hard to segment because of the thin and elongated structure, unless they have a very distinctive color from the background. The other major issues consist in poses that are not present in the training set, which means that the closest support vectors will not generalize well into these examples.

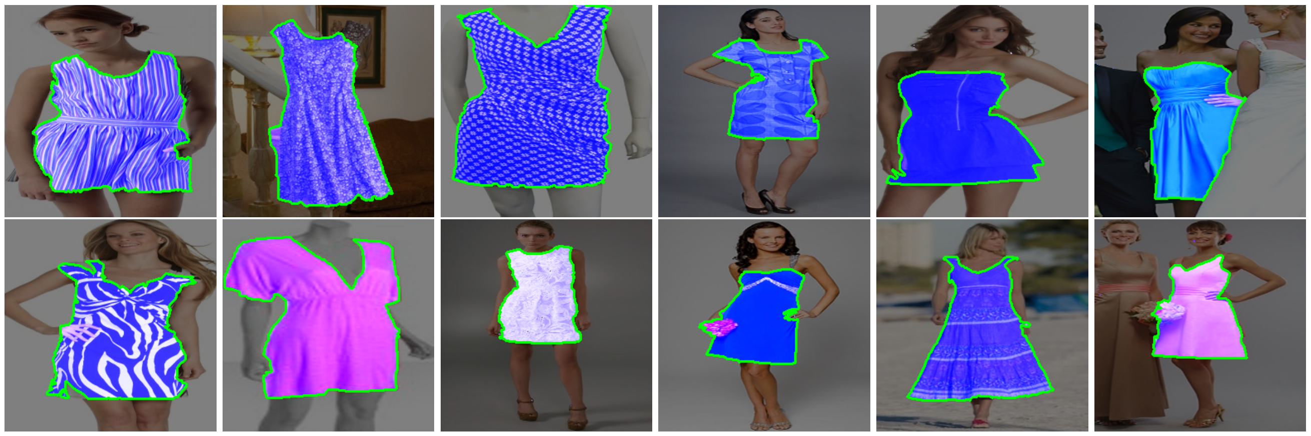
Figure 4. Some segmentation results on the Dresses dataset. Despite the presence of complex backgrounds and distractors (such as other people besides the main model), the proposed algorithm shows promising performances.

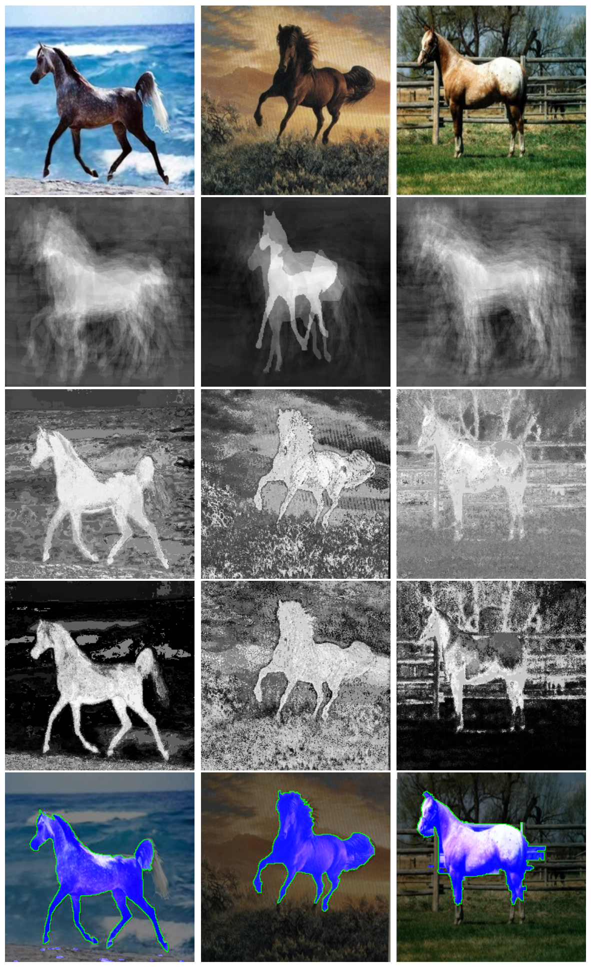
Figure 5. 1st row: Different horse input images. 2nd, 3rd and 4th rows: foreground potentials generated by the shape kernel, local image kernel and global image kernel respectively. 5th row: final segmentation results given by the learned structured SVM model.

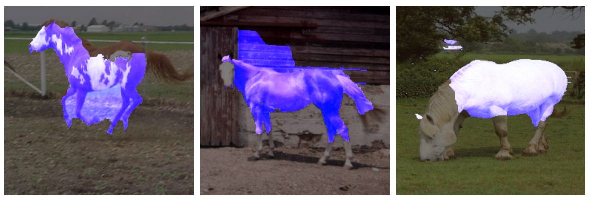
Figure 6. Some failure cases of the learned model

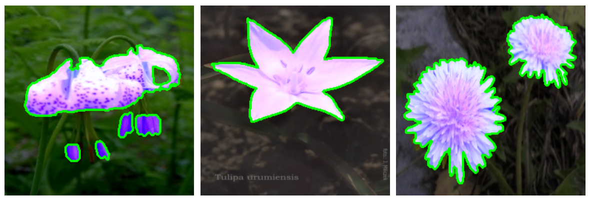
Figure 7. Segmentation results on the Oxford Flower Dataset.

Finally we tested our algorithm on the Oxford Flower Dataset [17]. Our algorithm achieves $S_a = 97.66\%$ and $S_o = 92.33\%$ in the labeled region, which is comparable with $S_o = 94\%$ reported in [17]. It has to be pointed out that their method is very domain specific, relying on a flower shape model that consists of flower center and petals, while our method is more generally applicable. Figure 7 shows some qualitative results.

6. Conclusions and Future Work

We have presented a segmentation approach that uses a kernelized structural SVM to learn discriminatively, from a training set of examples, to generate the most appropriate segmentation mask of an unseen image. By designing novel non-linear kernel functions, we combine high level object similarity information (via object feature descriptors) with multiple low level segmentation cues in an unified and principled framework. The result is a supervised learning approach for segmentation of general applicability. The only domain specific knowledge requirement is the design of the object descriptors, used in the object similarity kernel. Experimental evaluation suggests that our proposed approach compares favorably with other state-of-the-art methods on complex image segmentation benchmarks.

As potential future work, experimenting with other descriptors in the object similarity kernel will be interesting. Our approach does not yet model the boundary curves very well, they are mostly driven by low-level cues. Incorporating curve based shape knowledge could offer additional benefits. Instead of relying on a single global object similarity kernel, dividing the kernel into a parts-based representation could solve some of our current difficulties with the horse dataset and other objects with deformations and articulations. On the learning theory side, it would be interesting to establish a theoretical connection between the complexity of the top-down models the algorithm can learn and the number of segmentations needed in the training set.

Acknowledgments The authors would like to acknowledge Mark Taylor, Murali Venkatasubramanian, Jacquie Phillips, Konstantina Tzovolos and all the other colleagues at Like.com/Google for their help and support.

References

[1] The Pascal Visual Object Classes challenge. http://pascallin.ecs.soton.ac.uk/challenges/voc/. 2154

[2] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *IEEE CVPR*, pages 167–176, 2005. 2154, 2155

[3] F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *International Conference on Machine Learning*, 2004. 2157

[4] E. Borenstein. Weizmann Horses Database. http://www.msri.org/people/members/eranb/. 2157

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1222–1239, Nov 2001. 2156

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE CVPR*, 2005. 2154, 2155, 2157

[7] P. Felzenszwalb, R. Girshick, and D. McAllester. Discriminatively trained deformable part models. 2155, 2157

[8] C. Ionescu, L. Bo, and C. Sminchisescu. Structural SVM for visual localization and continuous state estimation. In *ICCV*, 2009. 2155

[9] T. Joachims, T. Finley, and C. N. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009. 2154, 2157

[10] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *IEEE CVPR*, 2010. 2158

[11] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:147–159, 2004. 2156

[12] M. Kumar, P. Torr, and A. Zisserman. Obj Cut. In *IEEE CVPR*, 2005. 2154, 2158

[13] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *ICCV*, 2003. 2154

[14] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. Jordan. Learning the kernel matrix with semidefinite programming. *JMLR*, 5:27–72, 2004. 2157

[15] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *ECCV Workshop on statistical learning in computer vision*, 2004. 2154

[16] A. Levin and Y. Weiss. Learning to combine bottom-up and top-down segmentation. *International Journal of Computer Vision*, 81(1):105–118, 2009. 2154, 2158

[17] M. E. Nilsback and A. Zisserman. Delving deeper into the whorl of flower segmentation. *Image and Vision Computing*, 2009. 2157, 2159

[18] S. Nowozin, P. Gehler, and C. Lampert. On parameter learning in CRF-based approaches to object class image segmentation. In *ECCV*, 2010. 2154, 2155

[19] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008. 2157

[20] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*, 2004. 2158

[21] M. Szummer, P. Kohli, and D. Hoiem. Learning CRFs using graph cuts. In *ECCV*, 2008. 2154, 2155

[22] B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. Max-margin parsing. *Empirical Methods in Natural Language Processing*, 2004. 2154

[23] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005. 2154, 2155

[24] K. Tsuda. Support vector classifier with asymmetric kernel. In *European Symposium on Artificial Neural Networks*, pages 183–188, 1999. 2156

[25] Z. Tu, X. Chen, A. Yuille, and S. C. Zhu. Image parsing: Unifying segmentation, detection, and object recognition. *IJCV*, 63(2):113–140, 2005. 2154

[26] M. Varma and D. Ray. Learning the discriminative power-invariance trade off. In *ICCV*, 2007. 2157

[27] W. Wu, j. Xu, H. Li, and S. Oyama. Asymmetric kernel learning. Technical report, Microsoft Research, 2010. 2156

[28] Y. Yang, S. Hallman, D. Ramanan, and C. Fowlkes. Layered object detection for multi-class segmentation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2010. 2154

[29] C. Yu, T. Joachims, R. Elbert, and J. Pillardy. Support vector training of protein alignment models. In *International Conference in Research on Computational Molecular Biology*, pages 253–267, 2007. 2154

[30] S. Yu, R. Gross, and J. Shi. Concurrent object recognition and segmentation by graph partitioning. In *Neural Information Processing Systems (NIPS)*, pages 1407–14, 2003. 2153

[31] S. Yu and J. Shi. Object-specific figure-ground segregation. In *IEEE CVPR*, 2003. 2153