Learning Graph Cut Energy Functions for Image Segmentation

Marco Manfredi, Costantino Grana and Rita Cucchiara
University of Modena and Reggio Emilia
Modena, Italy
Email: {name.surname}@unimore.it

***Abstract*—In this paper we address the task of learning how to segment a particular class of objects, by means of a training set of images and their segmentations. In particular we propose a method to overcome the extremely high training time of a previously proposed solution to this problem, Kernelized Structural Support Vector Machines. We employ a one-class SVM working with joint kernels to robustly learn significant support vectors (representative image-mask pairs) and accordingly weight them to build a suitable energy function for the graph cut framework. We report results obtained on two public datasets and a comparison of training times on different training set sizes.**

I. Introduction

Many computer vision applications require the precise identification of objects within a scene, and often the segmentation and selection of a particular target class [1], [2]. This binary class-specific segmentation problem, unlike other segmentation challenges, is well posed and its performance can be accurately measured by counting the number of mislabeled pixels.

Nowadays large repositories of annotated images are available, so an interesting approach would be to have a generic segmentation algorithm, specifically trained for the object class of interest. To obtain the desired flexibility, computational complexity must also be taken into account, focusing on solutions that are fast and accurate.

Currently, s/t graph cuts [3] are considered one of the most effective techniques in image segmentation, due to the wide range of energy functions that can be minimized using efficient max-flow algorithms [4], [5]. Recently, several works addressed the problem of introducing high level information into the graph cut energy functions [6], [7], in order to obtain flexible solutions, also applied to 3D images [8], [9]. An important incentive to this field was given by the medical imaging segmentation community [10], [11], [12], where the gray-level images and the low contrast make low level approaches unfeasible. The ability to learn a suitable energy function would make the system more flexible and easily adaptable to different settings.

Bertelli *et al.* [13] faced the supervised class-specific segmentation problem using Kernelized Structural Support Vector Machines (KSSVMs), achieving good results on several datasets. Unfortunately, KSSVMs can not be applied to large scale datasets because of their complexity [14], and usually only linear kernels are employed in conjunction with Structural SVMs [15] to reduce the training phase.

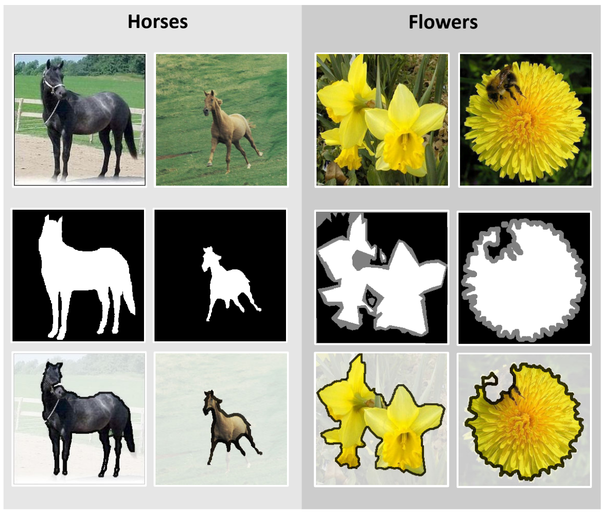

Fig. 1. Sample images from the two datasets (first row), the provided ground truth (second row) and the segmentation of the proposed approach (third row). The grey areas in the ground truth of the flower dataset are pixels labeled neither as foreground nor background.

In this paper, we focus on a generative learning technique for structured prediction, here applied to binary segmentation. Our proposal is able to dramatically reduce the training time, when compared to discriminative approaches. We exploit joint kernels on image-mask pairs, used in a one-class SVM to learn the energy function minimized in a graph cut framework. We discuss its application on two publicly available datasets, where we demonstrate that our proposal's performance is similar to the more complex and time consuming KSSVM. Fig. 1 shows some samples taken from the two datasets and the provided ground truth data.

The paper is organized as follows: in Section II we introduce the problem of structural segmentation also highlighting some limitations of one common approach, Sections III and IV describe the proposed method while in Section V and VI we give some further implementation details. In Section VII we present experimental evaluations on different datasets, and Section VIII summarizes the contributions of the paper.

II. Structural Segmentation

Structural prediction through SSVMs [16] proved to be effective in many computer vision tasks, such as scene recognition [17], object detection [18], tracking [19] and recently also image segmentation [20], [21], [13]. Structured segmentation

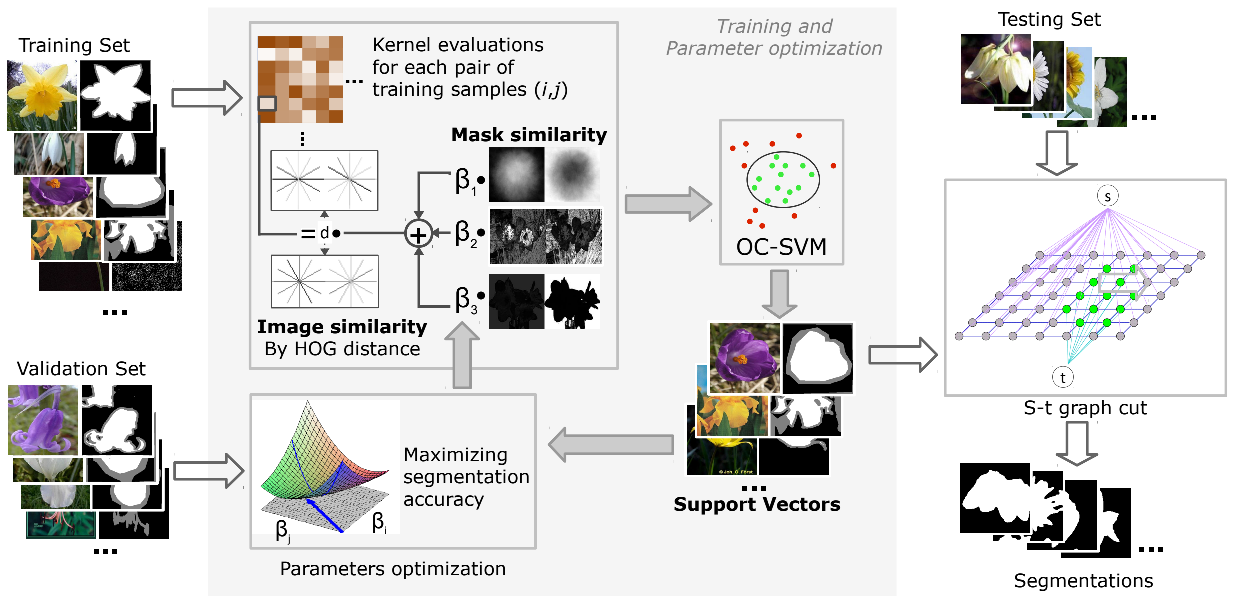
Training Set
Kernel evaluations for each pair of training samples (*i*,*j*)
Training and Parameter optimization
Testing Set
Mask similarity
β1
β2
β3
d
Image similarity By HOG distance
OC-SVM
Validation Set
Maximizing segmentation accuracy
βj
βi
Parameters optimization
Support Vectors
s
t
S-t graph cut
Segmentations

Fig. 2. A schematization of the proposed approach.

describes the problem of learning a function

$$f : \mathcal{X} \to \mathcal{Y}, \tag{1}$$

where $\mathcal{X}$ is the space of samples (images) and $\mathcal{Y}$ is the space of structured labels (binary masks). To learn f we assume that a training set of image-mask pairs $(x_1, y_1), ..., (x_n, y_n)$ is available. SSVM learns a scoring function $F(x, y)$ that matches a sample x with a label y, such that maximizing F through the label space gives the correct output label for sample x.

A common approach is to have F in the form of a linear function:

$$F(x, y, \mathbf{w}) = \mathbf{w}^\top \phi(x, y), \tag{2}$$

where $\mathbf{w} \in \mathbb{R}^n$ is a parameter vector and $\phi(x, y)$ is a joint feature vector. The definition of an explicit feature vector $\phi(x, y)$ can be very difficult, thus we will work in the dual formulation using positive definite joint kernels

$$K : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R}. \tag{3}$$

As defined in [13], the scoring function $F(x, y)$ can be written as:

$$\begin{aligned} F(x,y) = \sum_{y' \in \mathcal{W}} \alpha_{y'} \Bigg(\frac{1}{n} \sum_{i=1}^{n} [K((x,y),(x_i,y_i)) - \\ K((x,y),(x_i,y_i'))] \Bigg), \end{aligned} \tag{4}$$

where $\mathcal{W}$ is the set of the most violated constraints, and α are the weights for the support vectors that are found solving the dual problem. Given an input image x, we can find the output label by maximizing $F(x, y)$:

$$y^* = \arg\max_{y \in \mathcal{Y}} F(x, y). \tag{5}$$

This maximization can be done using graph cuts as demonstrated in [13]. Unfortunately, this formulation has two relevant performance issues:

- during training we have to construct the set of the most violated constraints $\mathcal{W}$: for each training sample, find k constraints (k depends on the desired accuracy), each with the size of the training set, and solve an inference step for each element;
- during testing we have to compare a sample x with each support vector, composed of every training sample and its most violated constraint, as in (4).

III. Energy functions modeling

The main idea behind the proposed model, summarized in Fig. 2, is to exploit one-class SVMs in a kernel space to learn a set of support vectors and their relative weights and to delete outliers from the training set, thus reducing the complexity at testing time. This idea has been firstly introduced by Lampert *et al.* [14], with the name of Joint Kernel Support Estimation, and applied to object localization and sequence labeling. Given a training set of sample-label pairs $(x_1, y_1), ..., (x_n, y_n)$ we want to model the probability density function $p(x, y)$, and use $f(x) = \arg\max p(x, y)$ for prediction. Assuming that $p(x, y)$ is high only if y is a correct label for x, we only have to find the support of $p(x, y)$. This can be effectively obtained by a one-class support vector machine (OC-SVM). We can express $p(x, y)$ as:

$$p(x, y) = \exp(\mathbf{w}^\top \phi(x, y)). \tag{6}$$

As mentioned in Section II, it is difficult to find an explicit formulation of $\phi(x, y)$, while it is easier to find a suitable joint kernel K that matches two sample-label pairs. The joint kernel can be an arbitrary Mercer kernel [22]. The output of the OC-SVM learning process becomes a linear combination of kernel evaluations with training samples, thus the prediction function can be formulated as:

$$f(x) = \arg\max_{y \in Y} \sum_{i=1}^{n} \alpha_i K\left((x,y),(x_i,y_i)\right). \tag{7}$$

The selected support vectors are the training samples that have non zero α. The learning process can be done using standard existing implementations of OC-SVM, replacing the kernel matrix within samples with the joint kernel matrix between sample-label pairs.

It is important to point out the difference between our approach and KSSVMs: in the training phase we only have to construct the joint kernel matrix between training samples, and then train a standard non linear OC-SVM, no inference steps are required during training. As a consequence, the training time does not depend on the structure of the output space, but only on the size of the training set.

IV. Defining the kernels

Joint kernels between image-mask pairs allow us to model complex relationships between samples. As proposed in [13], we choose to formulate the similarity kernel as the product of an image kernel and a mask kernel:

$$K((x_i,y_i),(x_j,y_j)) = \theta(x_i,x_j) \cdot \Omega(x_i,x_j,y_i,y_j), \tag{8}$$

where $\theta(x_i,x_j)$ measures the similarity of the objects depicted in x_i and x_j, and acts as a weight for the mask similarity kernel $\Omega(x_i,x_j,y_i,y_j)$. Consequently, if the two images are very different, the final similarity measure will be low, even if the masks are similar.

A. *Image Similarity Kernel*

The purpose of the image similarity kernel is to return high similarity values between images that contain very similar objects. We adopt a general purpose similarity measure between images, the comparison of HOG descriptors [23], although many other descriptors could be used without changing the model [24], [25]. HOGs can be compared using standard similarity measures like Bhattacharyya distance. Since we are working with images of the same category (e.g. flowers), distances within an entire dataset don't change so much; as a consequence "good" and "bad" samples are weighted similarly, leading to errors at classification time. A better choice is to employ a Gaussian kernel, capable of better distinguishing between different images, due to the parameter σ, optimized for a specific dataset. The image similarity kernel between image x_i and image x_j becomes:

$$\theta(x_i,x_j) = \exp\left(-\frac{\|\rho(x_i)-\rho(x_j)\|^2}{2\sigma^2}\right), \tag{9}$$

where $\rho(x_i)$ is the feature vector extracted from image x_i. Fig. 3 shows the different results obtained by using Bhattacharyya distance or the Gaussian kernel, to better understand the consequences at classification time. For the computation of the HOG descriptors we adopted rectangular HOG (R-HOG) [23], computing gradients on R,G, and B color channels and taking the maximum, then dividing the image with a 5×5 grid of cells (25 cells), and grouping them in 4 partially overlapped blocks of 3×3 cells each. Trilinear interpolation between histogram bins and cells was appropriately applied.

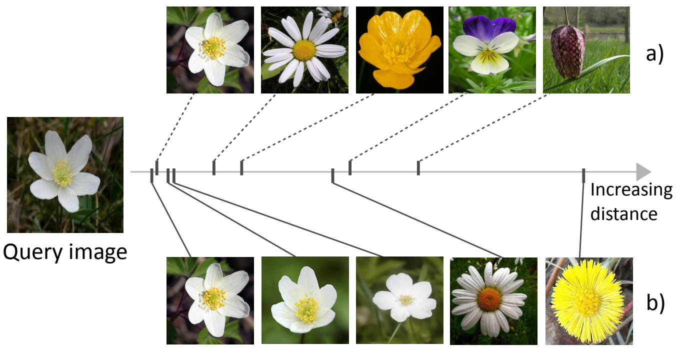

Fig. 3. HOG feature distance using Bhattacharyya (a) measure or a Gaussian kernel (b). The Gaussian kernel is able to separate with a larger gap flowers that are similar to the query from other flowers.

The HOG feature is computed using 9 bins to quantize the orientation, leading to 9 (bins) × 9 (cells per block) × 4 (blocks) = 324 features.

B. *Mask Similarity Kernel*

The mask similarity kernel takes into consideration both images and masks to extract knowledge about how comparable two segmentations are. The kernel is composed of a linear combination of three parts:

$$\Omega(x_i,x_j,y_i,y_j) = \sum_{k=1}^{3} \beta_k \Omega_k(x_i,x_j,y_i,y_j). \tag{10}$$

The first kernel $\Omega_1(y_i,y_j)$ only depends on the binary masks, and directly compares the similarity between the two, by counting the number of corresponding pixels:

$$\Omega_1(y_i,y_j) = \frac{1}{P}\sum_{p=1}^{P} \delta(y_{ip},y_{jp}), \tag{11}$$

where P is the total number of pixels in the image, y_{ip} is the p-th pixel of image y_i, and $\delta(\cdot,\cdot)$ is an indicator function defined as:

$$\delta(y_{ip},y_{jp}) = \begin{cases} 1 & \text{if} \quad y_{ip} = y_{jp} \\ 0 & \text{if} \quad y_{ip} \neq y_{jp} \end{cases}. \tag{12}$$

The second and the third kernel exploit 3D color histograms computed in the RGB space. Let's define F_i^j and B_i^j as foreground and background histograms extracted from image x_i using mask y_j, and $Pr(x_p \,|\, H)$ as the likelihood of pixel x_p to match histogram H. We use negative log-likelihoods to express the penalties to assign a pixel to foreground or to background, as firstly introduced by [3]. Negative log-likelihoods are defined as:

$$L(x_p \,|\, H) = -log(P_r(x_p|H)). \tag{13}$$

We can also define:

$$L(x_p|y_p,F,B) = \begin{cases} L(x_p \,|\, B) & \text{if} \quad y_p = \text{"}obj\text{"} \\ L(x_p \,|\, F) & \text{if} \quad y_p = \text{"}bkg\text{"}. \end{cases} \tag{14}$$

To highlight the mutual agreement of two masks, the second kernel extracts an histogram from image x_i using mask y_j and evaluates it using y_i. Having F_i^j and B_i^j:

$$\Omega_2(x_i, y_i) = \frac{1}{P}\sum_{p=1}^{P} L(x_{ip}|y_{ip}, F_i^j, B_i^j). \tag{15}$$

The third kernel exploits global features extracted from the entire training set to model the expected color distribution of foreground and background pixels. We define F_G and B_G as the global histograms extracted from training samples using their respective masks.

$$\Omega_3(x_i, x_j, y_i, y_j) = \Omega_{3i} \cdot \Omega_{3j} \tag{16}$$

where

$$\begin{array}{l} \Omega_{3i} = \frac{1}{P}\sum_{p=1}^{P} L(x_{ip}|y_{ip}, F_G, B_G) \\ \Omega_{3j} = \frac{1}{P}\sum_{p=1}^{P} L(x_{jp}|y_{jp}, F_G, B_G) \end{array}. \tag{17}$$

The histograms are quantized uniformly over the 3D color space using a fixed number of bins per channel, set to 16 by experimental evaluations (no smoothing is applied).

V. Graph construction

It is worth noting that the previously defined kernels compare two image-mask pairs, while at testing time the test mask is obviously missing. Kernels must thus be reformulated so to return pixel-wise potentials, in order to perform the maximization reported in (7). This maximization is done using s/t graph cuts [3].

The problem can be formulated as a maximum a posterior estimation of a Markov Random Field, minimizing the energy function:

$$E(y) = R(y) + \lambda B(y), \tag{18}$$

where y is a binary vector of pixel labels and $R(y)$ is the unary term expressing the cost of assigning a pixel to the foreground or to the background. $B(y)$ is the smoothness term, formulated as proposed by Rother *et al.* [26]:

$$B(y) = \sum_{p,q\in\mathcal{N}} \delta(y_p, y_q) \frac{1}{dist(p,q)} \exp\left(-\frac{\|x_p - x_q\|^2}{2\sigma^2}\right) \tag{19}$$

where $\mathcal{N}$ is the set of neighboring pixels (8-connected), $\delta(\cdot,\cdot)$ is the indicator function defined in (12), $dist(p,q)$ is the distance between pixels and σ is the expectation of the euclidean distance in color space $\|x_p - x_q\|^2$. At classification time we have to compute the foreground and background potentials P_f and P_b corresponding to the unary term in the graph cut framework. They are the result of a linear combination of potentials P_{f_i} and P_{b_i} obtained from the comparison of the testing image x_j with each support vector (x_i, y_i), weighted by the corresponding α_i. The potentials at position p are:

$$\begin{array}{l} P_{f\,ip} = \theta(x_i, x_j)(\beta_1 P_{f\,ip}^{1} + \beta_2 P_{f\,ip}^{2} + \beta_3 P_{f\,ip}^{3}) \\ P_{bip} = \theta(x_i, x_j)(\beta_1 P_{bip}^{1} + \beta_2 P_{bip}^{2} + \beta_3 P_{bip}^{3}) \end{array} \tag{20}$$

where $\theta(x_i, x_j)$ is the image similarity kernel defined in (9). The first kernel is strictly related to the mask y_i:

$$\begin{array}{l} P_{f\,ip}^{1} = y_{ip} \\ P_{bip}^{1} = 1 - y_{ip} \end{array} \tag{21}$$

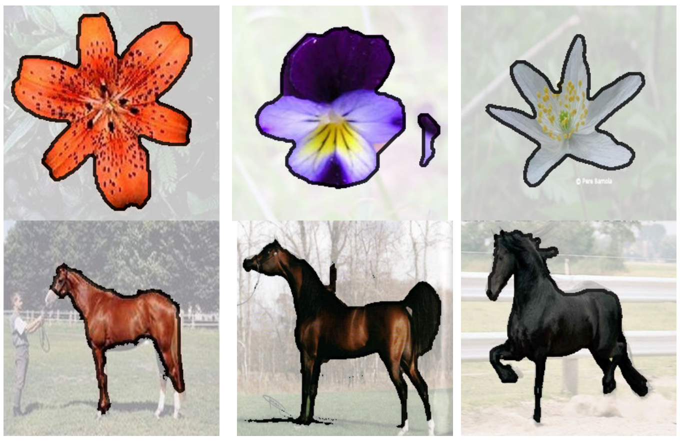
Fig. 4. Segmentation results on the flowers and the horses datasets.

The second kernel expresses the cost of assigning a pixel to foreground or to background, according to the histograms F_j^i and B_j^i, defined in Section IV-A:

$$\begin{array}{l} P_{f\,ip}^{2} = L(x_{jp}\,|\,B_j^i) \\ P_{bip}^{2} = L(x_{jp}\,|\,F_j^i) \end{array} \tag{22}$$

The third kernel expresses the cost of assigning a pixel to foreground or to background, given the global histograms F_G , B_G calculated on the training set:

$$\begin{array}{l} P_{f\,ip}^{3} = L(x_{jp}|B_G) \cdot \frac{1}{P}\gamma(x_i, y_i) \\ P_{bip}^{3} = L(x_{jp}|F_G) \cdot \frac{1}{P}\gamma(x_i, y_i) \end{array} \tag{23}$$

where

$$\gamma(x_i, y_i) = \sum_{p=1}^{P} L(x_{ip}|y_{ip}, F_G, B_G). \tag{24}$$

VI. Parameter optimization

Some parameters of the proposed approach have been optimized on a validation set, maximizing the segmentation accuracy. These are the weights of the mask kernels β_1,β_2 and β_3, the ν of OC-SVM, and the λ of the graph cut framework. The parameter ν is used in the OC-SVM to specify an upper bound to the percentage of outliers, the higher the ν, the higher the percentage of training samples that can be ignored by the OC-SVM, introducing robustness against outliers. The parameter λ of the graph cut weights the importance of the smoothness term in the final energy function and must be optimized because the unary potential $R(y)$ changes when the kernel weights change.

The optimization is done iteratively, by changing one parameter at a time. Each parameter has an optimization range and a step size. The best value within the range is searched and the range is recentered on it. If the current center does not change, then the step size is halved and the optimization process moves to the next parameter. This operation is repeated for a fixed number of iterations (set to 4 by experimental evaluation). For all the parameters that affect the training phase (β_1, β_2, β_3 and ν), the OC-SVM is retrained to test each new value. A detailed discussion on the parameter values on the different datasets is given in Section VII-A.

TABLE I. PERFORMANCE COMPARISON ON THE WEIZMANN HORSES AND OXFORD FLOWERS DATASETS.

Flower Dataset	$S_a(\%)$	$S_o(\%)$
KSSVM + Hog feature [13]	97.66	92.33
Our method	**97.17**	**92.14**
Flower shape model [28]	-	94
Horses Dataset	$S_a(\%)$	$S_o(\%)$
KSSVM + Hog feature	93.9	77.9
Our method	**93.04**	**76.32**
GrabCut init. with BB	69.53	50.39
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

VII. EXPERIMENTAL EVALUATION

We tested the proposed method on two publicly available datasets, containing images of flowers and horses. The first is the Weizmann horse dataset [27], that contains 328 images of horses with strong differences in background, contrast and pose. The second is the Oxford flower dataset [28], composed of 849 images of flowers belonging to different species.

All the images are resized to the same dimension to allow the kernel computation, the chosen size is 256×256 for both the datasets. We split each dataset in three parts and trained our method on the first one, optimized the parameters on the second one, and tested the system on the third one; eventually we exchanged the parts and averaged the results (three tests are conducted for each experiment).

We used two metrics to evaluate the segmentation performance: the pixel-wise accuracy S_a, that measures the percentage of correctly labeled pixels, and the intersection-over-union metric S_o, defined as the intersection of the output mask and the ground truth mask divided by the union of the two masks.

$$\begin{aligned} S_a &= \frac{1}{P}\sum_{p=1}^{P}\delta(M_p, M_{GTp}) \\ S_o &= \frac{\sum_{p=1}^{P} M_p=\text{“}obj\text{”} \wedge M_{GTp}=\text{“}obj\text{”}}{\sum_{p=1}^{P} M_p=\text{“}obj\text{”} \vee M_{GTp}=\text{“}obj\text{”}} \end{aligned} \tag{25}$$

where M is the output mask of the system, M_{GT} is the ground truth mask and $\delta(\cdot,\cdot)$ is the indicator function defined in (12). We firstly compared our solution with the one proposed in [13] on the flower dataset (Table I). As a comparison we also report the results obtained by [28]. Here it is important to note that the method proposed in [28] is strictly related to the domain of flowers, because it exploits a flower shape model made of center and petals. Moreover, our method is capable to obtain the same results of the KSSVM while dramatically reducing the computational complexity; this quickly becomes a key feature when dealing with larger datasets.

On the horses dataset we compared our method with the KSSVM framework and with the GrabCut framework [26] as a baseline. We tested different automatic initialization strategies for GrabCut: the first is done with the bounding boxes coming from the part based detector by Felzenszwalb *et al.* [29], [30], the others employ the average of the masks of the k nearest images found with Eq. (9). KSSVMs perform slightly better, probably due to their discriminative nature that allows to put aside wrong (but feasible) segmentations, exploiting the most violated constraints.

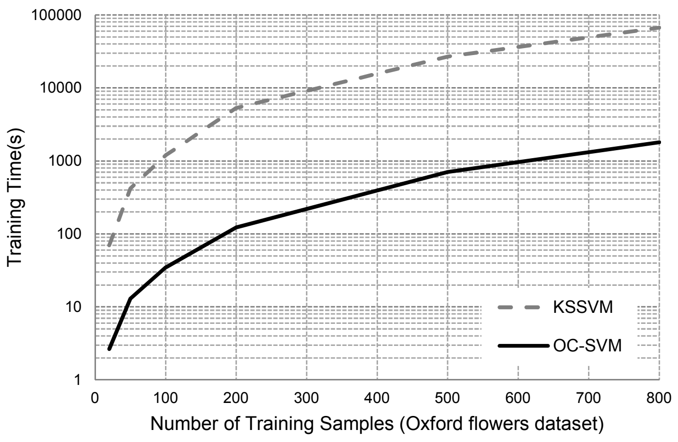

Fig. 5. Comparison of the time required to train the system with KSSVM and our proposal.

A. Parameter optimization

To understand the meaning of the parameters involved in the method, we summarized in Table II their optimized values. The optimization leads to different values for the two datasets, and some observations can be made:

- the first mask kernel is more important for the horses, probably due to the pose homogeneity through the dataset;
- the second mask kernel, which exploits information from both images and masks, is the most important, and receives the higher weight in both the datasets;
- the third mask kernel, which employs global color information learned from training, is more important for the flowers, and this is certainly related to the homogeneous background of flower images, that often depict grass or soil;
- parameter ν (for details see Section VI) is higher in the flowers dataset and this means that is convenient to ignore a certain percentage of training samples (about 45%), that do not provide important information.

B. Training Time Comparison

To highlight the difference in terms of training time between our approach and KSSVMs, we chose the largest dataset at our disposal, that is the Oxford flowers dataset, and compared training performance increasing the number of training samples from 20 to 800. Given that the code for KSSVMs is not publicly available, we used our implementation, based on the LaRank classifier [31]. In Fig. 5 a comparison between the two methods is reported.

TABLE II. OPTIMIZED PARAMETERS ON THE TWO DATASETS.

Parameter	Flowers	Horses
β_1	0.2	0.28
β_2	1.0	1.0
β_3	0.16	0.05
λ	0.24	0.18
ν	0.45	0.2

Although the training time of our approach increases in a non-linear manner due to the exponential number of kernel computations needed, KSSVM training time is one or two order of magnitude higher, and increases in a non-linear manner too.

VIII. Conclusions

We proposed a novel segmentation approach based on one-class SVMs and joint kernels between image-mask pairs. The method exploits the ability of OC-SVMs to identify and ignore outliers in the training set, while reducing the number of kernel computations needed at classification time. The characteristics of this generative learning algorithm allow faster training and testing phases when compared to discriminative approaches like KSSVMs while reaching comparable performance.

References

[1] M. Manfredi, C. Grana, S. Calderara, and R. Cucchiara, "A complete system for garment segmentation and color classification," *Machine Vision and Applications*, vol. 25, no. 4, pp. 955–969, May 2014.

[2] C. Grana, D. Borghesani, and R. Cucchiara, "Automatic segmentation of digitalized historical manuscripts," *Multimedia Tools and Applications*, vol. 55, no. 3, pp. 483–506, Dec. 2011.

[3] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images," in *Proceedings of the 8th IEEE International Conference on Computer Vision*, vol. 1, 2001, pp. 105–112.

[4] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 147–159, Feb. 2004.

[5] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.

[6] N. Vu and B. Manjunath, "Shape prior segmentation of multiple objects with graph cuts," in *Proceedings of the 21th IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008, pp. 1–8.

[7] O. Veksler, "Star Shape Prior for Graph-Cut Image Segmentation," in *Proceedings of the 8th European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, 2008, vol. 5304, pp. 454–467.

[8] V. Lempitsky, A. Blake, and C. Rother, "Branch-and-Mincut: Global Optimization for Image Segmentation with High-Level Priors," *Journal of Mathematical Imaging and Vision*, vol. 44, pp. 315–329, 2012.

[9] X. Chen, J. Udupa, U. Bagci, Y. Zhuge, and J. Yao, "Medical Image Segmentation by Combining Graph Cuts and Oriented Active Appearance Models," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 2035–2046, Apr. 2012.

[10] D. Freedman and T. Zhang, "Interactive graph cut based segmentation with shape priors," in *Proceedings of the 18th IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, Jun. 2005, pp. 755–762.

[11] S. Andrews, C. McIntosh, and G. Hamarneh, "Convex multi-region probabilistic segmentation with shape prior in the isometric log-ratio transformation space," in *Proceedings of the 13th International Conference on Computer Vision*, 2011, pp. 2096–2103.

[12] A. Lucchi, K. Smith, R. Achanta, G. Knott, and P. Fua, "Supervoxel-Based Segmentation of Mitochondria in EM Image Stacks With Learned Shape," *IEEE Transaction on Medical Imaging*, vol. 31, no. 2, pp. 474–486, 2012.

[13] L. Bertelli, T. Yu, D. Vu, and B. Gokturk, "Kernelized structural SVM learning for supervised object segmentation," in *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011, pp. 2153–2160.

[14] C. Lampert and M. Blaschko, "Structured prediction by joint kernel support estimation," *Machine Learning*, vol. 77, pp. 249–269, 2009.

[15] A. Lucchi, Y. Li, K. Smith, and P. Fua, "Structured Image Segmentation Using Kernelized Features," in *Proceedings of 12th European Conference on Computer Vision*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7573, pp. 400–413.

[16] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.

[17] S. Parizi, J. Oberlin, and P. Felzenszwalb, "Reconfigurable models for scene recognition," in *Proceedings of the 25th IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 2775–2782.

[18] R. B. Girshick, P. F. Felzenszwalb, and D. A. McAllester, "Object Detection with Grammar Models," in *Neural Information Processing Systems*, 2011, pp. 442–450.

[19] S. Hare, A. Saffari, and P. Torr, "Struck: Structured output tracking with kernels," in *Proceedings of the 13th International Conference on Computer Vision*, Nov. 2011, pp. 263–270.

[20] M. Szummer, P. Kohli, and D. Hoiem, "Learning CRFs Using Graph Cuts," in *Proceedings of the 8th European Conference on Computer Vision*, 2008, vol. 5303, pp. 582–595.

[21] S. Nowozin, P. Gehler, and C. Lampert, "On Parameter Learning in CRF-Based Approaches to Object Class Image Segmentation," in *Proceedings of the 10th European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, 2010, vol. 6316, pp. 98–111.

[22] V. Vapnik, *Statistical learning theory*. Wiley, 1998.

[23] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 18th IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, Jun. 2005, pp. 886–893.

[24] M. Bertini, A. Del Bimbo, G. Serra, C. Torniai, R. Cucchiara, C. Grana, and R. Vezzani, "Dynamic pictorially enriched ontologies for video digital libraries," *IEEE MultiMedia*, vol. 16, no. 2, pp. 41–51, Apr. 2009.

[25] G. Serra, C. Grana, M. Manfredi, and R. Cucchiara, "Modeling local descriptors with multivariate gaussians for object and scene recognition," in *Proceedings of the 21th ACM International Conference on Multimedia*, Barcelona, Spain, Oct. 2013, pp. 709–712.

[26] C. Rother, V. Kolmogorov, and A. Blake, ""GrabCut": interactive foreground extraction using iterated graph cuts," in *ACM SIGGRAPH 2004 Papers*, ser. SIGGRAPH '04, 2004, pp. 309–314.

[27] E. Borenstein, E. Sharon, and S. Ullman, "Combining top-down and bottom-up segmentation," in *IEEE International Conference on Computer Vision and Pattern Recognition Workshops, 2004. CVPRW '04.*, vol. 4, 2004, pp. 46–54.

[28] M.-E. Nilsback and A. Zisserman, "Delving deeper into the whorl of flower segmentation," *Image and Vision Computing*, vol. 28, no. 6, pp. 1049–1062, Jun. 2010.

[29] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester, "Discriminatively Trained Deformable Part Models, Release 5," http://people.cs.uchicago.edu/ rbg/latent-release5/.

[30] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1627–1645, 2010.

[31] A. Bordes, N. Usunier, and L. Bottou, "Sequence labelling svms trained in one pass," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, vol. 5211, pp. 146–161.