Structured Image Segmentation Using Kernelized Features

Aurélien Lucchi[1], Yunpeng Li[1], Kevin Smith[2], and Pascal Fua[1,⋆]

[1] Computer Vision Laboratory, EPFL, Lausanne
[2] Light Microscopy Center, ETHZ, Zurich

Abstract. Most state-of-the-art approaches to image segmentation formulate the problem using Conditional Random Fields. These models typically include a unary term and a pairwise term, whose parameters must be carefully chosen for optimal performance. Recently, structured learning approaches such as Structured SVMs (SSVM) have made it possible to jointly learn these model parameters. However, they have been limited to linear kernels, since more powerful non-linear kernels cause the learning to become prohibitively expensive. In this paper, we introduce an approach to "kernelize" the features so that a linear SSVM framework can leverage the power of non-linear kernels without incurring the high computational cost. We demonstrate the advantages of this approach in a series of image segmentation experiments on the MSRC data set as well as 2D and 3D datasets containing imagery of neural tissue acquired with electron microscopes.

1 Introduction

Conditional random fields (CRFs) are a class of powerful graphical models that have been highly successful in image segmentation. In the CRF framework, the solution is typically obtained by minimizing an energy function that is the sum of *unary* and *pairwise* terms [1]. The unary term encodes the likelihood that a particular label should be assigned to a pixel based on local image features[2]. The pairwise term encodes the tendency of neighboring pixels or superpixels to share the same label, thus enforcing spatial regularity.

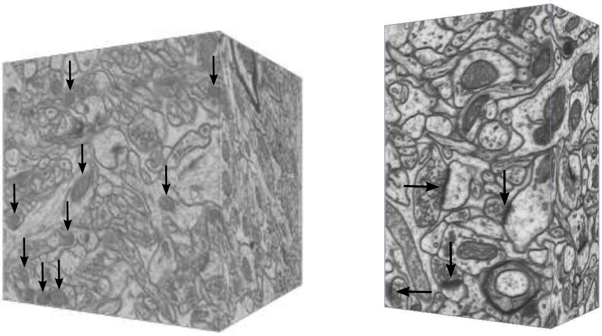
In recent years, machine learning techniques have increasingly been used to derive these terms in favor of simpler traditional models. However, this is usually done separately for each term: the unary term is optimized for labeling individual pixels while the pairwise term is optimized for labeling pixel pairs. As a result, the two terms can often be incommensurate, and an *ad hoc* weighting step is required to balance their relative influences. Instead, the parameters of the unary and pairwise terms should be learned jointly to infer the optimal labeling. This can be done using the recent structured-SVM (SSVM) framework [1]. It involves learning the unary and pairwise terms jointly through an iterative cutting-plane scheme that provably minimizes an upper bound on the empirical loss and the model complexity. Currently, SSVMs can be of practical use

[⋆] This work was supported in part by the MicroNano ERC project.

[1] Higher order terms are also possible in theory, though less commonly used in practice due to higher computational cost. The unary term is also referred to as the data term.

[2] Alternatively, groups of pixels or superpixels can be used in order to speed up computation.

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part II, LNCS 7573, pp. 400–413, 2012.



(a) mitochondria *(b)* synapses

Fig. 1. Two nearly isotropic stacks of neural tissue acquired using EM microscopy annotated for training and testing. *(a)* This stack contains 1000 images of 1024 × 1024 pixels, and was used for the task of segmenting mitochondria, indicated by arrows. *(b)* This stack contains 250 images of 655 × 429 pixels and was used to segment synaptic gaps, indicated by arrows.

only in conjunction with terms that are linear functions of the input features. This is because introducing non-linear terms would result in quadratic times for learning iterations, which quickly becomes unmanageable for regular-size 2D images and even more so for 3D data. Thus, it is usually not practical to use non-linear kernels in an SSVM framework, though they are often more powerful than their linear counterparts.

In this paper, we overcome this limitation through a two-step learning approach. We first use a regular non-linear SVM to create kernel-transformed feature vectors, each of which consists of kernel products between the input feature vector and a set of basis vectors that may not be orthogonal to each other. We then train a linear SSVM on the transformed features. This approach combines the power of non-linear kernels for individual pixel classification in the unary term with the regularizing effect of the pairwise term, while enforcing consistency between the two. This yields both improved segmentation performance and computational efficiency.

This line of research was primarily motivated by the need to more accurately segment synapses and mitochondria in electron microscopy (EM) stacks, such as those of Fig. 1. In this kind of data, unlike in popular segmentation benchmarks such as MSRC [2], global features that predict whether or not a type of object appears anywhere in the image are not useful. This is because it is known that synapses and mitochondria always appear in EM stacks, whereas it is not known if a cow or bird will appear in a particular image from a standard benchmark dataset such as MSRC. Thus, for EM applications, the CRF must rely solely on unary evidence and spatial smoothness. We demonstrate on all three datasets that our approach indeed boosts the performance of the learned CRF. Furthermore, it outperforms the previous state-of-the-art in our target applications: synapse and mitochondria segmentation.

2 Related Work

For tasks such as segmentation, consistent labeling of highly-correlated neighboring pixels is of great importance. Structured prediction has emerged as a powerful tool to take into account such correlations. In this section, we first discuss current approaches to structured prediction and the computational complexity issues that restrict them from use in conjunction with non-linear kernels. We then consider kernel approximation techniques that can be used to address these issues. Finally, we discuss structural kernels which, like our approach, rely on kernel functions in a structured prediction framework. But unlike our approach, they do so over the entire output space, which carries certain disadvantages.

Structured Prediction. Structured prediction methods such as conditional random fields (CRFs) [3] have been widely applied to problems with structured outputs. While traditional classifiers, such as decision trees and SVMs independently map each data instance to a single label, structured prediction methods take into account the statistical correlations between labels. This is critical for tasks such as image segmentation, where such correlations are strong between nearby pixels. Learning CRF models using large-margin methods has rapidly gained popularity in recent years. This is because they are more objective-driven and do not involve the daunting partition function that can render maximum-likelihood approaches intractable in CRFs with loopy graph structures. Compared with earlier approaches including the max-margin Markov network [4], the structured support vector machine (SSVM) [1] is especially appealing, and has since been successfully applied to many computer vision tasks, such as in [5,6,7], among others. The SSVM's appeal is due, in part, to its ability to take into account a variety of *loss functions*.

Computational Complexity. SSVMs, however, require the CRF energy function to be linear, which in turn places the same restriction on all the unary and spatial terms due to the additive nature of CRF energies. While, in principle, the linear function can be defined in some high dimensional or possibly infinite-dimensional space, reproducing this kernel Hilbert space through the use of non-linear kernels is often infeasible in practice given that the number of kernel evaluations grows quadratically with the model size and must be optimized in the dual space. Though SSVM learning techniques based on sampled cuts [8,9] have alleviated this problem to some extent, they do have to sacrifice some performance for speed and, even with this trade-off, are still generally much slower than linear SSVMs [8]. Moreover, earlier implementations of these techniques were intended for use in conjunction with the cutting-plane method to speed up training of regular non-linear SVMs. This results in a *multivariate* output space in the SSVM formulation, analogous to a CRF without edges, which is considerably less useful for image segmentation.

Kernel Approximation. Kernel Approximation is another way to improve SSVM training efficiency by seeking a lower, finite-dimensional representation of the kernel-induced feature map that lies in a higher or infinite dimensional space. This can be achieved by random sampling from the typically infinite-dimensional feature map [10,11] whose analytical form can be obtained using Fourier analysis when the kernel is homogeneous or stationary [12]. While promising results have been shown for specific additive kernels [13,12], it is less clear how this approach generalizes to non-additive kernels, such as the Gaussian RBF that is more difficult to approximate. Alternatively, the recently proposed *locally* linear SVM [14] can be used to simulate a non-linear decision boundary. However, this does not yield a *globally* linear function and, thus, does not fit into the SSVM framework. Moreover kernel approximation typically introduces additional tuning parameters such as the number of samples, which often present a performance-speed trade-off for which well-defined tuning criteria are lacking.

Structural Kernels. Finally, it is worth pointing out the difference between our approach and the recently proposed *structural kernels*, which also perform structured prediction using non-linear kernels, but in a very different setting. In [9,15], the kernels are defined on the overall output space, that is, the entire CRF, to exploit image-level "structural" information such as shape and color. While this serves to bias local labels and is useful for segmenting large dominant objects from the background, it often requires training data that completely characterizes the possible object configurations, such as binary masks. Multiple objects, or objects whose pose are not represented in the trained model will cause the approach to fail. Our approach, on the other hand, uses regular kernels defined as products between a set of basis vectors and the feature vectors extracted from individual nodes (i.e. pixels or superpixels). This has the effect of making it more "local", and robust to such failures.

3 Learning a CRF with Kernelized Features

We begin by describing our CRF model for segmentation in Sec. 3.1. We then discuss how to learn its parameters using an SSVM framework in Sec. 3.2, with specific details on how to express the CRF model in the required linear form in Sec. 3.3. In Sec. 4, we introduce a technique to create "kernelized" features, enabling us to leverage the power of non-linear kernels while the SSVM remains linear. Fig. 2 outlines our approach.

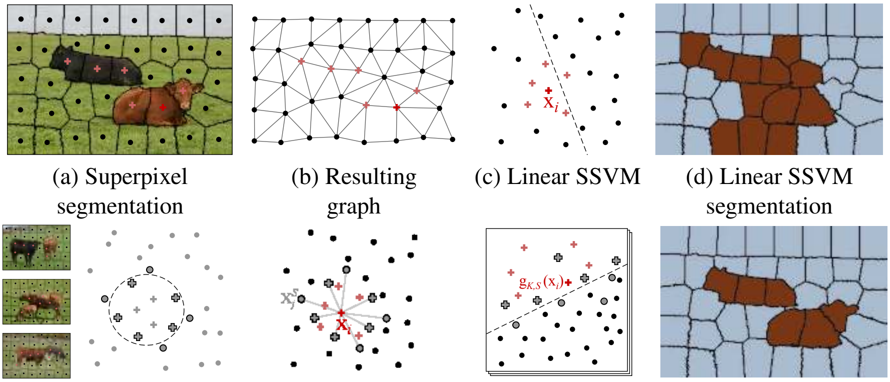
3.1 CRF for Segmentation

As a standard preprocessing step, we perform a preliminary over-segmentation of our input image into superpixels [3] using SLIC [16]. The CRF $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is thus defined so that each node $i \in \mathcal{V}$ corresponds to a superpixel and there is an edge $(i, j) \in \mathcal{E}$ between two nodes i and j if the corresponding superpixels are adjacent in the image. Let $Y = \{y_i\}$ for $i \in \mathcal{V}$ denote the labeling of the CRF which assigns a class label y_i to each node i. Its energy function can then be written as

$$E_{\mathbf{w}}(Y) = \sum_{i \in \mathcal{V}} D_i(y_i) + \sum_{(i,j) \in \mathcal{E}} V_{ij}(y_i, y_j), \tag{1}$$

where D_i is the unary term and V_{ij} is the pairwise term. Both D_i and V_{ij} depend on the observed data and the CRF parameters $\mathbf{w}$, in addition to the labeling Y. The energy is

[3] Or supervoxels in the case of volumetric data.



(a) Superpixel segmentation (b) Resulting graph (c) Linear SSVM (d) Linear SSVM segmentation

(e) Standard RBF SVM trained with individual superpixels (f) Kernel transform (g) "Kernelized" SSVM (h) "Kernelized" SSVM segmentation

Fig. 2. Our approach. *(a)* A superpixel over-segmentation of an image where each superpixel center is marked (+/● denotes foreground/background). *(b)* The superpixel graph used to construct the CRF, where each node corresponds to a superpixel, and edges indicate adjacency in the image. *(c)* An illustration of the feature space. Each point represents a feature vector extracted from a superpixel. Because it is not linearly separable, the standard SSVM gives a poor segmentation result in *(d)*. *(e)* To address this, we train a non-structured kernel SVM on individual superpixels to obtain a set of support vectors S, indicated by outlined points. *(f)* Kernel-transformed features $\mathbf{g}_{K,S}(\mathbf{x}_i)$ are obtained for each feature vector x_i from the kernel products of x_i and S. *(g)* Data in the $|S|$-dimensional "kernelized" feature space is linearly separable, and can be used to train a linear SSVM. *(h)* The improved segmentation result.

also commonly referred to as the "cost" in the literature and we will use the two terms interchangeably. The inferred optimal labeling is simply the one that minimizes it, that is, $Y^* = \arg\min_{Y \in \mathcal{Y}} E_{\mathbf{w}}(Y)$, where $\mathcal{Y}$ denotes the set of all possible labelings. While the exact minimization of the energy function is generally intractable on loopy CRFs, good approximate solutions can be found efficiently using techniques such as graph cuts [17] and belief propagation [18]. In our case, we use graph cuts when the energy function is submodular [19] and belief propagation otherwise.

3.2 Learning the CRF Using SSVM

Structured SVM (SSVM) is a large-margin method for learning the parameters of models with structured outputs, such as the CRF model we use for segmentation. The SSVM uses the ground truth training data to learn the CRF parameters so that the inferred labeling of the CRF is "close" to that of the training data, defined as yielding a low *loss*. More specifically, given a set of N training examples with ground truth labelings $(Y^{(1)}, \ldots, Y^{(N)})$, the SSVM [4] optimizes a quadratic objective function of the parameters $\mathbf{w}$ subject to a set of linear soft margin constraints

[4] Here, we use the margin-rescaling variant of SSVM [1].

$$\min_{\mathbf{w},\boldsymbol{\xi}\geq\mathbf{0}} \frac{1}{2}\,||\mathbf{w}||_2^2 + \frac{C}{N}\sum_{n=1}^{N}\xi_n \tag{2}$$
$$\text{s.t. } \forall n, Y \in \mathcal{Y}_n \backslash Y^{(n)} : \delta E_{\mathbf{w}}(Y) \geq \Delta(Y^{(n)}, Y) - \xi_n$$

where $\mathcal{Y}_n$ is the set of all possible labelings for example n, ξ_n are the slack variables, and $\delta E_{\mathbf{w}}(Y)$ is shorthand for $E_{\mathbf{w}}(Y) - E_{\mathbf{w}}(Y^{(n)})$. The constant C controls the tradeoff between margin and training error, and the loss function Δ measures the closeness of a labeling Y to the ground truth $Y^{(n)}$.

A natural choice for Δ is the per-superpixel 0-1 loss $\Delta(Y^{(n)}, Y) = \sum_{i\in\mathcal{V}} \Delta(y_i^{(n)}, y_i)$ with $\Delta(y_i^{(n)}, y_i) = I(y_i \neq y_i^{(n)})$, which penalizes all errors equally. However, in image segmentation, it is common for certain classes to occur much more frequently than others. To ensure good performance across all classes, we adopt a loss function that weighs errors for a given class inversely proportional to the frequency with which it appears in the training data

$$\Delta(y_i^{(n)}, y_i) = \begin{cases} \frac{1}{\text{frequency}(y_i^{(n)})} & \text{, if } y_i \neq y_i^{(n)} \\ 0 & \text{, otherwise.} \end{cases} \tag{3}$$

Since the total number of constraints grows exponentially with the CRF size, they cannot be exhaustively enumerated in most cases. The SSVM solves this by employing an iterative cutting-plane algorithm, which finds the most violated constraint for each example n

$$\hat{Y} = \arg\min_{Y\in\mathcal{Y}_n} E_{\mathbf{w}}(Y) - \Delta(Y^{(n)}, Y) \tag{4}$$

at every iteration and adds it to the working set of constraints. As with inferring the optimal labeling, finding the most violated constraint is intractable on loopy CRFs. However, the approximate most violated constraints can be found efficiently using the same kind of energy minimization techniques as in inference, and this approach has proven effective in practice [20,5].

3.3 Linearizing the CRF Energy Function

Since the SSVM operates by solving a quadratic program (QP), all the constraints in Equation 2 must be linear [1]. This requires that the energy function $E_{\mathbf{w}}$ be expressible as an inner product between the parameter vector and a feature map. Since the energy is the sum of individual unary and pairwise terms, this implies that D_i and V_{ij} also must be expressible as

$$D_i(y_i) = \left\langle \mathbf{w}^D, \psi_i^D(y_i) \right\rangle \tag{5}$$

and

$$V_{ij}(y_i, y_j) = \left\langle \mathbf{w}^V, \psi_{ij}^V(y_i, y_j) \right\rangle, \tag{6}$$

where $\psi_i^D(y_i)$ and $\psi_{ij}^V(y_i, y_j)$ are feature maps dependent on both the observed data and the labels, and where

$$\mathbf{w} = ((\mathbf{w}^D)^T, (\mathbf{w}^V)^T)^T \tag{7}$$

is the vector of parameters that define the functions D_i and V_{ij} respectively.

If we let $\Psi^D(Y) = \sum_{i \in \mathcal{V}} \psi_i^D(y_i)$ and $\Psi^V(Y) = \sum_{(i,j) \in \mathcal{E}} \psi_{ij}^V(y_i, y_j)$, then $\Psi(Y) = (\Psi^D(Y)^T, \Psi^V(Y)^T)^T$. allowing the CRF energy to now be written linearly as

$$E_{\mathbf{w}}(Y) = \langle \mathbf{w}, \Psi(Y) \rangle . \tag{8}$$

Let $\mathbf{x}_i$ be a feature vector associated with node i extracted from the observed data. We can define the data feature map as

$$\psi_i^D(y_i) = (I(y_i = 1)\mathbf{x}_i^T, \ldots, I(y_i = K)\mathbf{x}_i^T)^T , \tag{9}$$

where K is the number of possible labels, i.e., $y_i \in \{1, \ldots, K\}$. If we write $\mathbf{w}^D = (\mathbf{w}_1^D, \ldots, \mathbf{w}_K^D)^T$, the unary term becomes the inner product

$$D_i(y_i) = \left\langle \mathbf{w}_{y_i}^D, \mathbf{x}_i \right\rangle , \tag{10}$$

which represents the energy of node i taking on label y_i. Similarly, if we define the pairwise feature map as

$$\psi_{ij}^V(y_i, y_j) = (I(y_i = a, y_j = b))_{(a,b) \in \{1,\ldots,K\}^2} \tag{11}$$

with the corresponding parameters $\mathbf{w}^V = (w_{ab})_{(a,b) \in \{1,\ldots,K\}^2}$, then the pairwise term

$$V_{ij}(y_i, y_j) = w_{y_i y_j} \tag{12}$$

reflects the transition cost between nodes i and j from label y_i to label y_j. Although the above definition depends only on the labels y_i and y_j, the pairwise term can, in fact, be made data-aware (as in [2,7]). For instance, it can be made gradient-adaptive by including parameters for each discretized gradient level. In a similar fashion, it can be made to consider geometric relationships such as "sky should appear above grass"[5].

4 Kernel-Transformed Features

As shown in the previous section, standard SSVMs require an energy function that is linear in the parameters and features. Since unary terms based on non-linear SVMs are often more powerful and produce better results [21,7], this constitutes a major limitation. It should be possible, in principle, to learn non-linear unary terms within the SSVM framework by implicitly defining $\mathbf{w}$ and $\mathbf{x}_i$ of Eq. 10 in a high-dimensional space through kernels. In practice, however, these kernels are very high- or even infinite-dimensional, making such an approach computationally intractable.

Our approach aims at circumventing this problem. It starts from the observation that a non-linear binary (+1/-1 label) SVM classifier always takes the form

$$\text{Score}(\mathbf{x}) = \sum_j \alpha_j y_j^S K(\mathbf{x}_j^S, \mathbf{x}) , \tag{13}$$

[5] We incorporate both geometric and gradient context in our model. They are omitted from the notation for brevity, as the extension follows naturally from above.

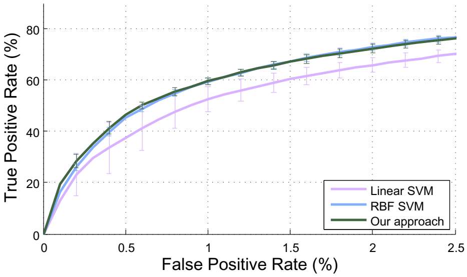


Fig. 3. Comparison of a linear SVM, RBF-SVM, and a linear SVM trained on feature vectors kernelized using the support vectors of the RBF-SVM. For classification of individual superpixels (ignoring structure), training a linear SVM on kernelized feature vectors yields a similar performance to a standard RBF-SVM. Error bars indicate standard deviation over 10 experiments

where $\mathbf{x}_j^S \in S$ are the support vectors with corresponding labels y_j^S. Extended to multiclass labels ($y_i \in \{1, \ldots, K\}$) for a general unary term, it becomes

$$D_i(y_i) = -\sum_j \alpha_j c(y_j^S, y_i) K(\mathbf{x}_j^S, \mathbf{x}_i) \; , \tag{14}$$

where $c(y_j^S, y_i)$ is 1 if $y_i = y_j^S$ and -1 otherwise. Note that, although the function is non-linear in the input features $\mathbf{x}_i$, because of the non-linear kernel K, it is linear in the kernel products $K(\mathbf{x}_j^S, \mathbf{x}_i)$.

If we define $\mathbf{g}_{K,S}(\mathbf{x}_i)$ as the vector of kernel products

$$\mathbf{g}_{K,S}(\mathbf{x}_i) = (K(\mathbf{x}_1^S, \mathbf{x}_i), \ldots, K(\mathbf{x}_{|S|}^S, \mathbf{x}_i))^T, \tag{15}$$

and $\mathbf{w'}_{y_i}$ as their coefficients

$$\mathbf{w'}_{y_i} = (-\alpha_1 c(y_1^S, y_i), \cdots, -\alpha_{|S|} c(y_{|S|}^S, y_i))^T, \tag{16}$$

then the unary term can be re-expressed as

$$D_i(y_i) = \langle \mathbf{w'}_{y_i}, \mathbf{g}_{K,S}(\mathbf{x}_i) \rangle \; , \tag{17}$$

which is of the finite-dimensional linear form needed for learning within the SSVM framework, as discussed in the previous section.

This suggests a simple, 2-step learning approach to incorporate kernels into an SSVM, illustrated in Fig. 2. First, we train a standard non-structured non-linear kernel SVM using feature vectors extracted from individual superpixels to obtain a set of support vectors S. S are then used as *basis vectors* to create a set of *kernel-transformed* (or "kernelized") feature vectors $\mathbf{g}_{K,S}(\mathbf{x}_i)$, which are provided to train the linear SSVM.

Although our formulation is not equivalent to a non-linear kernel SSVM[6], nor can it be shown to approximate a non-linear SSVM as kernel approximation methods

[6] The parameters $\mathbf{w}'$ now correspond to the primal variables of a linear SSVM instead of the dual variables of a non-linear SSVM.

do [10,13,12], it does produce models with the same functional form as those learned using a kernel SSVM and, importantly, performs well in practice. To demonstrate the principle that a linear SVM trained using kernel-transformed features performs similarly to a non-linear SVM that uses the same kernel, we conducted a simple experiment. The goal was to classify individual superpixels, ignoring structure. We compare the performance of a standard linear SVM, an SVM trained with an RBF kernel, and a simplification of our approach in which feature vectors kernelized using the support vectors obtained from the RBF-SVM are used to train a standard linear SVM. The results appearing in Fig. 3 support our intuition – so long as we transform the original feature vector using the right set of basis vectors, learning the new coefficients under a different objective function (i.e., as primal instead of dual variables) yields performance similar to a non-linear kernel SVM.

5 Results

Our primary motivation for developing the technique presented here was to segment synaptic gaps and mitochondria from images acquired with an electron microscope, such as the ones depicted in Fig. 1. For such data, it is known *a priori* that particular cellular structures will appear in the image. Consequently, the use of global features and/or priors which has proved to be essential when the presence of a particular category is uncertain, as is the case for segmentation benchmarks such as MSRC [2], is of no particular benefit. Because EM segmentation methods can not use such global features, they have to more heavily rely on smoothness terms. Nevertheless, we demonstrate that our approach boosts performance of the learned CRF model on the MSRC dataset as well as the EM data. In subsection 5.2 we are able to match state-of-the-art results on the MSRC dataset by incorporating global features. For the EM data in subsections 5.3 and 5.4, our approach significantly outperforms the state-of-the-art.

5.1 Competing Methods

To highlight the relative importance of the various components of our approach, we compare against the following variations which we treat as baselines.

- **Linear SVM** – A standard linear SVM trained on the original feature vectors x_i. Each superpixel is classified independently (i.e., without CRF).
- **Linear SSVM/CRF** – As described in Sec. 3.2, a linear SSVM on the original features is used to learn the parameters of a CRF, which is used for segmentation.
- **RBF SVM** – A non-linear SVM trained on the original feature vectors using an RBF kernel. Each superpixel is classified independently.
- **RBF+SSVM/CRF** – Instead of being kernelized versions of the original feature vectors as in our approach, transformed feature vectors contain the per-class scores of an RBF SVM trained on the original features. A linear SSVM learns the parameters of a CRF using the transformed features.
- **Codebook SVM** – The original feature vectors are transformed using a "kernel codebook" approach inspired by [22]. It relies on k-means clustering to create a set of basis vectors (codewords) instead of discriminatively learned SVM support vectors. We used as many codewords as support vectors for all our experiments. The resulting features are classified independently with a linear SVM.

Table 1. MSRC segmentation results. We follow the standard reporting procedure. For each category, the pixel-wise classification rate is provided. Global pixel-wise accuracy and average per-category scores provide measures for overall performance. The first part of the table shows baseline methods that ignore global information, while the second part compares our approach to state-of-the-art methods that consider global information. Bold entries are used for each part separately to indicate best performance.

	building	grass	tree	cow	sheep	sky	airplane	water	face	car	bicycle	flower	sign	bird	book	chair	road	cat	dog	body	boat	**Global**	**Average**
Linear SVM	**59**	79	78	60	63	70	78	61	46	61	76	57	64	59	58	63	70	52	55	68	31	68	62
RBF SVM	47	83	**80**	83	62	76	74	62	72	55	73	59	**86**	40	54	62	73	66	34	60	34	70	64
Linear SSVM/CRF	53	81	74	76	72	73	85	63	59	60	81	54	77	62	68	72	76	66	**68**	**71**	6	71	65
RBF+SSVM/CRF	54	**84**	70	75	69	80	86	**66**	69	**77**	87	46	74	**66**	65	69	65	77	58	68	18	71	68
Kernelized SSVM/CRF	41	77	79	**87**	**91**	**86**	**92**	65	**86**	65	**89**	**61**	76	48	**77**	**91**	**77**	**82**	32	48	**39**	**73**	**70**
G-Kernelized SSVM/CRF	59	90	**92**	82	83	94	91	80	85	**88**	96	89	73	48	96	62	81	87	33	44	30	82	76
Shotton et al. [23]	49	88	79	**97**	**97**	78	82	54	**87**	74	72	74	36	24	93	51	78	75	35	**66**	18	72	67
Ladicky et al. [24]	**80**	**96**	86	74	87	**99**	74	**8**7	86	87	82	**97**	**95**	30	86	31	**95**	51	**69**	**66**	09	**86**	75
Gonfaus et al. [21]	60	78	77	91	68	88	87	76	73	77	**93**	**97**	73	**57**	95	**81**	76	81	46	56	46	77	75
Lucchi et al. [7]	50	83	87	81	84	90	**97**	72	75	79	90	95	79	52	**97**	**81**	80	**89**	51	64	**60**	79	**78**

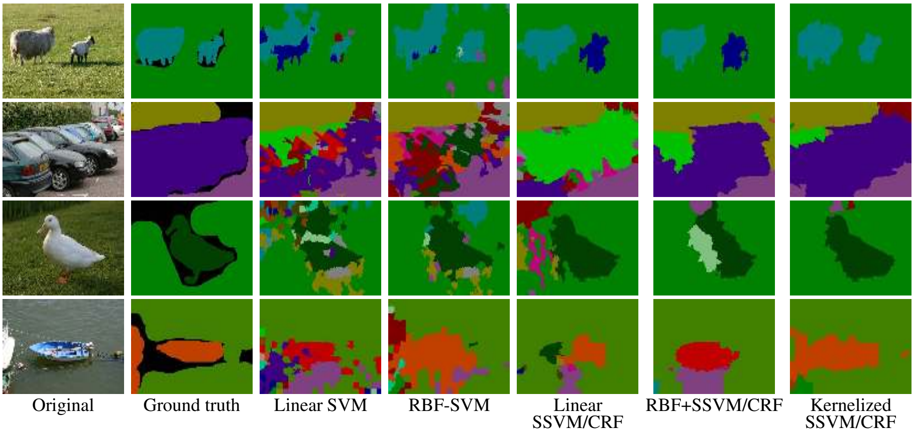


Fig. 4. Example segmentations from the MSRC dataset

- **Codebook SSVM/CRF** – Features are constructed in the same manner as for Codebook SVM, but a linear SSVM is used to learn a CRF for structured prediction.
- **Kernelized SSVM/CRF** – Our method. A linear SSVM learns the parameters of a CRF for structured prediction, using kernel-transformed features $\mathbf{g}_{K,S}(\mathbf{x}_i)$, as described in Sec. 4.

In addition to these, we also compare to state-of-the-art methods, [23,24,21,7] for MSRC and [25] for the EM dataset. All SSVM methods used Joachims' *SVM-struct* software [1] for training.

5.2 MSRC Dataset

The MSRC-21 dataset is a popular multi-class object segmentation benchmark dataset which contains 591 images with objects from 21 categories.

We extract feature vectors from each image by first over-segmenting the image using SLIC superpixels [16]. We then extract SIFT descriptors and color histograms from image patches surrounding each superpixel centroid. We also include location information as in [24]. This information is converted to a bag-of-words descriptor using a nearest-neighbor search[7]. The resulting descriptor serves as the feature vector x_i used to train the various methods. Training and testing is done using the standard split of the dataset [2]. Note that the Codebook SVM and Codebook SSVM methods are not reported since x_i already contains a bag-of-words descriptor. In addition to the baseline methods described above, we compare state-of-the-art approaches [23,24,21,7] to our approach incorporating global features:

- **G-Kernelized SSVM/CRF** – Our method as described in Sec. 4 using an augmented feature vector. In addition to SIFT descriptors and color histograms, the feature vector contains the global features described in [21].

Table 1 summarizes the segmentation performance of the various approaches and example segmentations appear in Fig. 4. The top of Table 1 compares our method to the baselines introduced in Sec. 5.1, demonstrating that structured learning tends to yield higher segmentation performance than unstructured approaches. It is also clear that non-linear models outperform linear ones for both structured and instance-based learning. Because our approach combines structured prediction with the power of non-linear kernels, it yields superior performance over the baselines in nearly every category.

The bottom of Table 1 compares our approach to state-of-the-art methods. As observed in [7], global features are necessary to obtain state-of-the-art performance on datasets such as MSRC. Therefore, when we do not use them our performance is lower but by incorporating the features of [21] into our approach (G-Kernelized SSVM/CRF), we are able to match state-of-the-art performance and out-perform all other methods in several categories.

5.3 Synaptic Gap Dataset

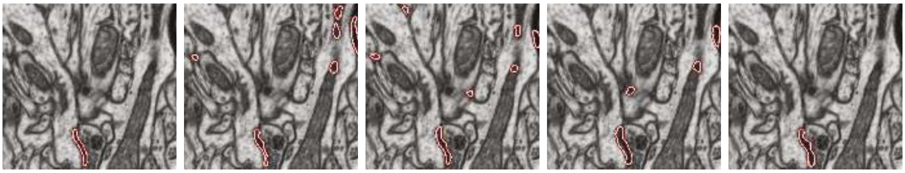
Segmenting the synaptic gaps from EM images of neural tissue shown in Fig. 1*(b)* is challenging due to the large amount of clutter including vesicles, mitochondria, and various cellular membranes that exhibit a variety of distracting shapes and textures. The dataset of Fig. 1*(b)* contains 250 images of 655×429 pixels and a total of 24 synapses. Each pixel was labeled by an expert as either synaptic or non-synaptic. The dataset was then split into 2 parts for training and testing.

We begin by over-segmenting each image using SLIC superpixels [16]. Each image contained an average of 4000 superpixels and 11400 edges. At the location of each superpixel we extract a feature vector consisting of intensity histograms and steerable filter responses. The later is computed by convolving a patch extracted at the center of each superpixel with a set of steerable filters at 3 different scales ($\sigma = \{2, 5, 6\}$). An SVM with an RBF kernel trained on 40K randomly sampled superpixels provides the support vectors for the kernel transform in our approach.

[7] A dictionary containing 1,000 words for SIFT features and 400 words for color histograms is constructed using k-means on extracted features.

Table 2. Segmentation results for the synaptic gap EM dataset. We measure segmentation performance using the *Jaccard index*, the ratio of the correctly segmented area to the union of the segmentation and the ground truth.

Linear SVM	Linear SSVM/CRF	RBF SVM	RBF+ SSVM/CRF	Codebook SVM	Codebook SSVM/CRF	**Kernelized SSVM/CRF**
58%	60%	61%	64%	60%	63%	**66%**



Ground truth Linear SVM RBF SVM Codebook SSVM/CRF Kernelized SSVM/CRF

Fig. 5. Segmentation results on the synaptic gap dataset. The kernelized SSVM correctly segments the synapse in this example, while the baseline methods include spurious segments.

A summary of the segmentation performance is provided in Table 2, and examples appear in Fig. 5. As with the MSRC baselines, we observe a trend in which structured learning yields higher segmentation performance than unstructured approaches and non-linear models outperform linear models. Our approach, which combines structured learning with non-linear models outperforms all other methods.

5.4 Mitochondria Dataset

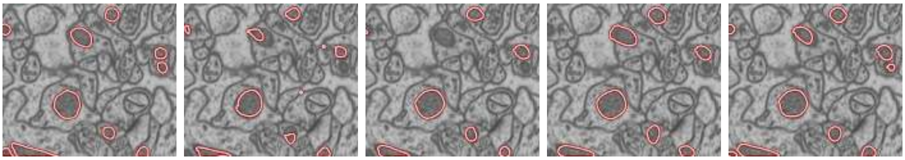
Here, we perform mitochondria segmentation in 3D using large image stacks from Fig. 1*(a)*. This greatly increases the scale of the problem since the image stacks are orders of magnitude larger than most 2D images.

We begin by over-segmenting the volume using SLIC supervoxels [16]. For each supervoxel, we extract a feature vector that captures local shape and texture information using Ray descriptors [27,25] and intensity histograms. Those feature vectors x_i are used to train each baseline methods, as well as our model. Due to the high cost of labeling such large volumes, our experiments are restricted to two subvolumes containing $1024 \times 768 \times 165$ voxels. The first subvolume, containing 42 mitochondria, was used to train the various methods; the second, containing 45 mitochondria, was used for testing. Each subvolume contains $\sim$13K supervoxels. The resulting graphs have $\sim$91K edges. An SVM with an RBF kernel trained on 4K randomly sampled supervoxels provides the support vectors for the kernel transform in our approach.

A summary of the segmentation performance is provided in Table 3. Example segmentations are provided in Fig. 6. Once again, we observe the trend in which structured learning yields higher performance than unstructured approaches and non-linear models outperform linear models. Comparing to [25], we can see that our approach of jointly learning the data and pairwise parameters in an SSVM framework is superior to learning them independently and then using CRF energy minimization for inference.

Table 3. Segmentation results for the mitochondria EM dataset. We measure segmentation performance using the *Jaccard index*.

Linear SVM	Linear SSVM/CRF	RBF SVM	RBF+ SSVM/CRF	Codebook SVM	Codebook SSVM/CRF	Lucchi et al. [25]	**Kernelized** **SSVM/CRF**
73%	79%	75%	80%	75%	80%	80%	**84%**



Ground truth Linear SVM RBF SVM Codebook SSVM/CRF Kernelized SSVM/CRF

Fig. 6. Segmentation results on the EM dataset. The kernelized SSVM correctly segments all mitochondria in this example, while other methods fail to detect some mitochondria, poorly delineate certain boundaries, or erroneously insert extra regions. Note that the data and segmentations are 3-dimensional – the images above correspond to slices through the test volume.

5.5 Discussion

The expense of training is an important consideration for any classification-based approach. As previously noted, an SSVM can, in principle, be trained using a non-linear kernel. However, this is not feasible in practice as the number of kernel evaluations grows quadratically with the size of the graph. We keep the cost linear by applying a kernel-transform to the features. But to do so, we must first train a standard non-linear SVM to obtain the support vectors used in the kernel transform. The training time of this step could be a potential cause for concern, as it incurs the same quadratic cost. Fortunately, it can be kept in check using known techniques such as randomly sampling the data or iteratively mining for hard examples [26]. However, these techniques cannot be applied to directly speed up the SSVM because they disregard the structure of the graph, which is essential for learning. Note that while we only used Gaussian radial basis functions in our experiments, they are not required by design and one could easily substitute them with other kernels such as Polynomial or Hyperbolic tangent.

6 Conclusion

In this work, we introduced a technique to leverage the power of non-linear kernels in a structured prediction framework by applying a kernel transform to the feature vectors. Results on three different datasets demonstrate the advantages of our approach. Although this work focuses on segmentation, the concept should generalize to other structured prediction problems such as gene sequencing and natural language parsing.

References

1. Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: ICML (2004)

2. Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context. IJCV 81 (2009)
3. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML (2001)
4. Taskar, B., Guestrin, C., Koller, D.: Max-margin Markov networks. In: NIPS (2003)
5. Szummer, M., Kohli, P., Hoiem, D.: Learning CRFs Using Graph Cuts. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 582–595. Springer, Heidelberg (2008)
6. Nowozin, S., Gehler, P.V., Lampert, C.H.: On Parameter Learning in CRF-Based Approaches to Object Class Image Segmentation. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part VI. LNCS, vol. 6316, pp. 98–111. Springer, Heidelberg (2010)
7. Lucchi, A., Li, Y., Boix, X., Smith, K., Fua, P.: Are Spatial and Global Constraints Really Necessary for Segmentation? In: ICCV (2011)
8. Yu, C.N.J., Joachims, T.: Training structural svms with kernels using sampled cuts. In: KDD (2008)
9. Aliaksei, M., Severyn, A.: Fast support vector machines for structural kernels. In: ECML (2011)
10. Rahimi, A., Recht, B.: Random features for large-scale kernel machines. In: NIPS (2007)
11. Balcan, M.F., Blum, A., Vempala, S.: Kernels as features: On kernels, margins, and low-dimensional mappings. Mach. Learn. 65, 79–94 (2006)
12. Vedaldi, A., Zisserman, A.: Efficient additive kernels via explicit feature maps. In: PAMI (2011)
13. Maji, S., Berg, A.: Max-Margin Additive Classifiers for Detection. In: ICCV (2009)
14. Ladicky, L., Torr, P.: Locally Linear Support Vector Machines. In: ICML (2011)
15. Bertelli, L., Yu, T., Vu, D., Gokturk, B.: Kernelized Structural Svm Learning for Supervised Object Segmentation. In: CVPR (2011)
16. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Suesstrunk, S.: SLIC Superpixels Compared to State-of-the-art Superpixel Methods. PAMI (in press, 2012)
17. Boykov, Y., Veksler, O., Zabih, R.: Fast Approximate Energy Minimization via Graph Cuts. PAMI 23 (2001)
18. Murphy, K.: Bayesian Map Learning in Dynamic Environments. In: NIPS, pp. 1015–1021 (1999)
19. Kolmogorov, V., Zabih, R.: What Energy Functions Can Be Minimized via Graph Cuts? PAMI 26, 147–159 (2004)
20. Finley, T., Joachims, T.: Training structural SVMs when exact inference is intractable. In: ICML (2008)
21. Gonfaus, J., Boix, X., Weijer, J., Bagdanov, A., Serrat, J., Gonzalez, J.: Harmony Potentials for Joint Classification and Segmentation. In: CVPR, pp. 3280–3287 (2010)
22. van Gemert, J.C., Geusebroek, J.-M., Veenman, C.J., Smeulders, A.W.M.: Kernel Codebooks for Scene Categorization. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 696–709. Springer, Heidelberg (2008)
23. Shotton, J., Johnson, M., Cipolla, P.: Semantic Texton Forests for Image Categorization and Segmentation. In: CVPR (2008)
24. Ladicky, L., Russell, C., Kohli, P., Torr, P.H.S.: Associative Hierarchical CRFs for Object Class Image Segmentation. In: ICCV (2009)
25. Lucchi, A., Smith, K., Achanta, R., Knott, G., Fua, P.: Supervoxel-Based Segmentation of Mitochondria in Em Image Stacks with Learned Shape Features. TMI 31, 474–486 (2011)
26. Felzenszwalb, P., Mcallester, D., Ramanan, D.: A Discriminatively Trained, Multiscale, Deformable Part Model. In: CVPR (2008)
27. Smith, K., Carleton, A., Lepetit, V.: Fast Ray Features for Learning Irregular Shapes. In: ICCV, pp. 397–404 (2009)