Learning for Structured Prediction Using Approximate Subgradient Descent with Working Sets

Aurélien Lucchi[1]* Yunpeng Li[1] Pascal Fua[1]
[1] Computer Vision Laboratory, EPFL, Lausanne

Abstract

We propose a working set based approximate subgradient descent algorithm to minimize the margin-sensitive hinge loss arising from the soft constraints in max-margin learning frameworks, such as the structured SVM. We focus on the setting of general graphical models, such as loopy MRFs and CRFs commonly used in image segmentation, where exact inference is intractable and the most violated constraints can only be approximated, voiding the optimality guarantees of the structured SVM's cutting plane algorithm as well as reducing the robustness of existing subgradient based methods. We show that the proposed method obtains better approximate subgradients through the use of working sets, leading to improved convergence properties and increased reliability. Furthermore, our method allows new constraints to be randomly sampled instead of computed using the more expensive approximate inference techniques such as belief propagation and graph cuts, which can be used to reduce learning time at only a small cost of performance. We demonstrate the strength of our method empirically on the segmentation of a new publicly available electron microscopy dataset as well as the popular MSRC data set and show state-of-the-art results.

1. Introduction

Markov random fields (MRF) [2] and conditional random fields (CRF) [11] are among the most widely used models in computer vision. They are particularly suited to low-level tasks, such as image segmentation, due to their ability to represent the interdependency between nearby pixels. The maximum-margin framework [26, 28] for learning MRFs and CRFs has gained much popularity in the recent years. As an alternative to maximum likelihood and maximum *a posteriori* learning, the max-margin criteria enjoy the advantage of avoiding the need to estimate the computationally difficult partition function and being able to optimize for many different performance metrics. A particularly successful large-margin formulation is the structured support vector machine (SSVM) [28], where the learning objective is to minimize a regularized hinge loss due to the violation of a set of soft margin constraints. This can be solved iteratively using the SSVM cutting plane algorithm [28] or by solving the equivalent unconstrained optimization problem using subgradient based methods [19, 15, 17, 30]. Both approaches require finding at each iteration the *most violated constraint*, namely the labeling that maximizes the margin-sensitive hinge loss [28], which is necessary for obtaining a valid cutting plane or a true subgradient of the objective. Finding such constraints is, however, intractable in loopy graphical models, such as the MRFs and CRFs usually used in image segmentation. Although approximate maximizers can be obtained by approximate inference, such as belief propagation [16] and graph cuts [3], and used as substitutes, the approximation can sometimes be imprecise enough to have a major impact on learning: An unsatisfactory constraint can cause the cutting plane algorithm to prematurely terminate if the new constraint does not have a higher hinge loss than all previous constraints; it can also induce erratic behavior of subgradient-based methods when the implied descent direction is too far away from any true subgradients. These phenomena therefore make the learning process more susceptible to failure.

In this paper, we propose to use a working set of constraints to increase robustness of approximate subgradient descent based learning. The resulting algorithm is particularly suited for minimizing the margin-sensitive hinge loss in the SSVM formulation [28] when the most violated constraints and hence the resulting subgradients are not exact. We show that the proposed method is able to obtain better subgradient approximations by computing them with respect to the whole working set, as opposed to existing approaches where only the last constraint is considered. Therefore, we are able to obtain sufficiently reliable approximate subgradients even when those due to individual constraints are noisy. This further enables us to replace the most violated constraints with randomly sampled labelings, thus avoiding the need to perform inference at all during

*This work was supported in part by the EU ERC Grant MicroNano

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.259

learning. The use of sampling leads to decreased learning time while still maintaining good levels of performance. We demonstrate the strength of our method on the task of learning CRF models for image segmentation.

The rest of the paper is organized as follows. We discuss the prior work in Section 2 and provide the background on the large-margin framework and learning techniques based on subgradient descent in Section 3. Section 4 describes our working set based algorithm in detail and analyze its properties. We present the experimental results in Section 5 and conclude in Section 6.

2. Related work

Maximum margin learning of CRFs was first formulated in the max-margin Markov networks (M^3N) [26], whose objective is to minimize a margin-sensitive hinge loss between the ground-truth labeling and all other labelings for each training example. This is especially appealing for learning CRFs with loopy structures, due to its more objective-driven nature and its complete bypassing of the partition function that presents a major challenge to maximum likelihood based approaches. Nevertheless, the number of constraints in the resulting quadratic program (QP) is exponential in the size of the graph, hence making it a highly non-trivial problem. In M^3N this is handled by rewriting the QP dual in terms of a polynomial number of marginal variables, which can then be solved by a coordinate descent method analogous to the sequential minimal optimization (SMO) [18]. However, solving such a QP is not tractable for loopy CRFs with high tree widths that are often needed in many computer vision tasks and even solving it approximately can become overwhelmingly expensive on large graphs.

Structured SVMs (SSVM) [28] optimize the same kind of objective as M^3N, while allowing for a more general class of loss functions. It employs a cutting plane algorithm to iteratively solve a series of increasingly larger QPs, which makes learning more scalable. However, the cutting plane algorithm requires the computation of the most violated constraints, namely the labeling maximizing the hinge loss [28]. This involves performing the *loss augmented inference* [25], which makes it intractable on loopy CRFs. Though approximate constraints can be used [5], they make the cutting plane algorithm susceptible to premature termination and can lead to catastrophic failure. Moreover, solving the QP can become slow as the set of constraints grows larger after each iteration, especially when the dimensionality of the feature space is also high.

An alternative to solving the quadratic program deterministically is to employ stochastic gradient or subgradient descent. This class of methods has been studied extensively for non-structured prediction problems [21]. In the context of structured prediction, learning can be achieved by finding a convex-concave saddle-point and solving it with a dual extra-gradient method [27]. In [19] max-margin learning is solved as an unconstrained optimization problem and subgradients are used to approximate the gradient in the resulting non-differentiable problem. This method trades optimality for a lower complexity, making it more suitable for large-scale problems. The approach of [15] proposes a perceptron-like algorithm based on an update whose expectation is close to the gradient of the true expected loss. However, the soundness of these methods heavily depends on the assumption that a valid subgradient is obtained at each iteration. Hence they become much less reliable when the subgradients are noisy due to inexact inference, as is the case for loopy CRFs.

The recently proposed SampleRank [29] avoids performing inference altogether during learning; Instead it samples labelings at random using Markov chain Monte Carlo (MCMC). At each step, parameters are updated with respect to a pair of sampled labelings. Though achieving notable speed improvement, the method does not in fact optimize the actual hinge loss but rather a loose upper bound on it. Hence, unlike our method, it solves a problem that is substantially different from the original max-margin formulation.

3. Max-margin Learning of CRFs

Conditional random fields (CRF) [11] are graphical models used to encode relationships between a set of input and output variables. Given its parameters $\mathbf{w}$, a CRF predicts the labeling Y for a given input X by maximizing some score function $S_{\mathbf{w}} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, i.e.,

$$\hat{Y} = \arg\max_{Y \in \mathcal{Y}} S_{\mathbf{w}}(Y) = \arg\max_{Y \in \mathcal{Y}} \mathbf{w}^T \Psi(X, Y) \quad (1)$$

The score is usually expressed as a linear function of $\mathbf{w}$ and can be written as $\mathbf{w}^T \Psi(X, Y)$, where the vector $\Psi(X, Y)$ is the *feature map* corresponding to the input X and the labeling Y. The fundamental properties of random fields imply that the feature map $\Psi(X, Y)$ and hence the score $S_{\mathbf{w}}$ decompose into sums over individual nodes and edges for any pairwise CRFs [2]. For a comprehensive review of CRF models, we refer readers to [11] and [24].

3.1. Discriminative Learning

Discriminative learning uses the labeled training data to learn the CRF parameters so that the inferred labeling of the CRF is "close" to that of the ground truth, defined as yielding a low *loss*. More specifically, given a set of N training examples $\mathcal{D} = ((X^1, Y^1), \ldots, (X^N, Y^N))$ where $X^i \in \mathcal{X}$ is an input example, such as the image or features associated to it, and $Y^i \in \mathcal{Y}$ is the associated labeling, the learning task consists in finding model parameters $\mathbf{w}$ that

achieve low empirical loss subject to some regularization. In other words, we seek

$$\begin{aligned}\mathbf{w}^* &= \arg\min_{\mathbf{w}} \mathcal{L}(\mathcal{D}, \mathbf{w}) \\ &= \arg\min_{\mathbf{w}} \sum_{(X^n, Y^n)\in\mathcal{D}} l(X^n, Y^n, \mathbf{w}) + R(\mathbf{w}), \quad (2)\end{aligned}$$

where l is the surrogate loss function and $R(\mathbf{w})$ is the regularizer (typically the L2 norm). The most common choice of l is the hinge loss, as used in [26, 28], which will be described later on in this section. Note that the definition of the surrogate loss l depends on the score function $S_{\mathbf{w}}$, since the goal of learning is to make the maximizer of $S_{\mathbf{w}}$ a desirable output for the given input.

3.2. Max-margin Formulation

The max-margin approach is a particular instance of discriminative learning, where parameter learning is formulated as a quadratic program (QP) with soft margin constraints [28]:

$$\begin{aligned}&\min_{\mathbf{w},\boldsymbol{\xi}\geq\mathbf{0}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n \quad (3)\\ &\text{s.t. } \forall n : S_{\mathbf{w}}(Y^n) \geq \max_{Y\in\mathcal{Y}_n}(S_{\mathbf{w}}(Y) + \Delta(Y^n, Y)) - \xi_n,\end{aligned}$$

where $\mathcal{Y}_n$ is the set of all possible labelings for example n, the constant C controls the trade-off between margin and training error, and the task loss Δ measures the closeness of any inferred labeling Y to the ground truth labeling Y^n.

The QP can be converted to an unconstrained optimization problem by incorporating the soft constraints directly into the objective function, yielding:

$$\begin{aligned}&\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \quad (4)\\ &\min_{\mathbf{w}} \frac{1}{2C}||\mathbf{w}||^2 + \sum_{n=1}^{N}[S_{\mathbf{w}}(Y^*) + \Delta(Y^n, Y^*) - S_{\mathbf{w}}(Y^n)]_+,\end{aligned}$$

where

$$Y^* = \arg\max_{Y\in\mathcal{Y}_n}(S_{\mathbf{w}}(Y) + \Delta(Y^n, Y)). \quad (5)$$

It is easy to see that Eq. 4 has the same form as Eq.2 where the hinge loss is used as the surrogate loss l, i.e.,

$$l(Y^n, Y^*, \mathbf{w}) = [S_{\mathbf{w}}(Y^*) + \Delta(Y^n, Y^*) - S_{\mathbf{w}}(Y^n)]_+ \quad (6)$$

For most existing approaches, a key challenge to solving Eq. 4 effectively is the loss-augmented inference, i.e., finding Y^*.

3.3. Stochastic Subgradient Descent

The objective function of Eq. 4 can be minimized via stochastic subgradient descent, similar to [19, 15]. This class of methods iteratively computes and steps in the opposite direction of a subgradient vector with respect to a example X^n chosen by picking an index $n \in \{1 \dots N\}$ uniformly at random. We then replace the objective in Eq. 4 with an approximation based on the training example (X^n, Y^n), yielding:

$$f(\mathbf{w}, n) = l(Y^n, Y^*, \mathbf{w}) + \frac{1}{2C}||\mathbf{w}||^2. \quad (7)$$

A subgradient of the convex function $f : \mathcal{W} \to \mathbb{R}$ at $\mathbf{w}$ is defined as a vector $\mathbf{g}$, such that

$$\forall \mathbf{w}' \in \mathcal{W}, \mathbf{g}^T(\mathbf{w}' - \mathbf{w}) \leq f(\mathbf{w}') - f(\mathbf{w}). \quad (8)$$

The set of all subgradients at $\mathbf{w}$ is called the *subdifferential* at $\mathbf{w}$ and is denoted $\partial f(\mathbf{w})$. The subdifferential is always a non-empty convex compact set.

A valid subgradient $g(Y^n, Y^*, \mathbf{w})$ with respect to the parameter $\mathbf{w}$ can always be computed as the gradient of $f(\mathbf{w})$ at Y^*. Hence for the hinge loss, it can be computed as:

$$\frac{\partial f(Y^n, Y^*, \mathbf{w})}{\partial \mathbf{w}} = \psi(Y^*) - \psi(Y^n) + \frac{\mathbf{w}}{C}. \quad (9)$$

This results in a simple algorithm that iteratively computes and steps in the direction of the negative subgradient. In order to guarantee convergence, the step size $\eta^{(t)}$ has to satisfy the following conditions :

$$\lim_{T\to+\infty}\sum_{t=1}^{T}\eta^{(t)} = \infty \quad \text{and} \lim_{T\to+\infty} \quad \sum_{t=1}^{T}(\eta^{(t)})^2 < \infty. \quad (10)$$

For loopy CRFs, however, true subgradients of the hinge loss cannot always be obtained due to the intractability of loss-augmented inference. This can lead to erratic behavior due to noisy subgradient estimates and loss of performance.

4. Estimating Subgradient Using Working Sets

Our algorithm aims at better estimating an approximate subgradient of $f(Y^n, Y, \mathbf{w})$ by using working sets of constraints, denoted $\mathcal{A}^n$, for learning loopy CRFs where exact inference is intractable. The algorithm we propose is outlined in Algorithm 1. It first solves the loss-augmented inference to find a constraint Y^* and add it to the working set $\mathcal{A}^n$. It then steps in the opposite direction of the approximate subgradient computed as an average over the set of violated constraints belonging to $\mathcal{A}^n$.

Hence unlike dual averaging methods [17, 30] that aggregate over all previous subgradients, our algorithm only considers the subset of active, namely violated, constraints

Algorithm 1

INPUTS :
  $\mathcal{D}$: Training set of N examples.
  β : Learning rate parameter.
  $\mathbf{w}^{(1)}$: Arbitrary initial values, e.g., 0.
OUTPUT : $\mathbf{w}^{(T+1)}$
Initialized $\mathcal{A}^n \leftarrow \emptyset$ for each $n = 1 \dots N$
for $t = 1 \dots T$ **do**
  Pick some example (X^n, Y^n) from $\mathcal{D}$
  $Y^* = \arg\max_{Y \in \mathcal{Y}_n} (S_{\mathbf{w}}(Y) + \Delta(Y^n, Y))$
  $\mathcal{A}^n \leftarrow \mathcal{A}^n \cup \{Y^*\}$
  $\mathcal{A}^{n'} \leftarrow \{Y \in \mathcal{A}^n \mid l(Y, Y^n, \mathbf{w}^{(t)}) > 0\}$
  $\eta^{(t)} \leftarrow \frac{\beta}{t}$
  $\mathbf{g}^{(t)} \leftarrow \frac{1}{|\mathcal{A}^{n'}|} \sum_{Y \in \mathcal{A}^{n'}} \frac{\partial f(Y^n, Y, \mathbf{w}^{(t)})}{\partial \mathbf{w}^{(t)}}$
  $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta^{(t)} \mathbf{g}^{(t)}$
end for

when computing the parameter updates. Therefore all subgradients are computed with respect to the parameters at the *current* iteration, as opposed to using their historical values. This produces more meaningful descent directions, as evidenced by the results in Section 5.

We now analyze the convergence properties of the algorithm presented in Algorithm 1. Although finding true subgradients as defined in Eq. 8 cannot be guaranteed for loopy CRFs, interesting results can still be obtained even if one can only find an approximate ϵ-subgradient $\mathbf{g}$, as defined in [22]:

$$\forall \mathbf{w}' : \mathbf{g}^T(\mathbf{w} - \mathbf{w}') \geq f(\mathbf{w}) - f(\mathbf{w}') - \epsilon \qquad (11)$$

The convergence properties of ϵ-subgradient descent methods were studied in [20, 22, 19]. The "regret" (i.e., loss) of the parameter vector $\mathbf{w}$ can be bounded as follows (the re-derivation of the proof using our notation is given in the supplementary material):

$$\mathbb{E}\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|_2^2 \leq \frac{G^2}{\lambda^2 t} + \frac{\epsilon}{\lambda}, \qquad (12)$$

where G is a constant satisfying the condition $||\mathbf{g}||^2 \leq G^2$ and $\lambda = \frac{1}{C}$.

Given that the choice of the step size satisfies Eq. 10, we can see that the first term on the right side of Eq. 12 goes to 0 so stochastic ϵ-subgradient descent converges to a certain distance ϵ to the optimum. The key to improving convergence is thus to obtain more accurate ϵ-subgradients, and we show below how this is achieved through the use of working sets.

Let $\mathbf{g}_1, \dots, \mathbf{g}_m \in \mathbb{R}^d$ be the approximate subgradients with respect to example (X^n, Y^n) of $\mathcal{L}$ for labelings in the working set $\mathcal{A}^n$ that still violates the margin constraint at a given iteration. Assume that each $\mathbf{g}_i \in \mathbb{R}^d$ comes from some distribution with mean $\boldsymbol{\mu}_i \in \partial\mathcal{L}(\mathbf{w})$ and bounded variance.

Let $\boldsymbol{\delta}_i = \mathbf{g}_i - \boldsymbol{\mu}_i$ be the difference between approximate ϵ-subgradient $\mathbf{g}_i$ and true ϵ-subgradient $\boldsymbol{\mu}_i$, and assume that all $\boldsymbol{\delta}_i$ are independent of one another. Note that, by definition, each $\boldsymbol{\delta}_i$ has zero expectation and hence their average $\bar{\boldsymbol{\delta}} = \frac{1}{m}\sum \boldsymbol{\delta}_i = \frac{1}{m}\sum \mathbf{g}_i - \frac{1}{m}\sum \boldsymbol{\mu}_i$.

Therefore, using Hoeffding's inequality [6] and the union bound, we can show that the average error $\bar{\boldsymbol{\delta}}$ concentrates around its expectation, i.e., 0 in this case, as the number of violated constraints in the working set m increases:

$$\Pr\left(\left|\left|\bar{\boldsymbol{\delta}}\right|\right| \geq r\right) \leq 2d \exp\left(\frac{-mr^2}{2G^2}\right), \qquad (13)$$

The convexity of the subdifferential $\partial\mathcal{L}(\mathbf{w})$ implies that $\bar{\boldsymbol{\mu}} = \frac{1}{m}\sum_i \boldsymbol{\mu}_i \in \partial\mathcal{L}(\mathbf{w})$. Therefore the probability of $\mathbf{g}^{(t)} \triangleq \frac{1}{m}\sum \mathbf{g}_i$ being more than a distance r away from any true subgradient is bounded by Eq. 13 as well.

Algorithm 2

INPUTS :
  $\mathcal{D}$: Training set of N examples.
  $\mathcal{Q}$: MCMC walker.
  β : Learning rate parameter.
  $\mathbf{w}^{(1)}$: Arbitrary initial values, e.g., 0.
OUTPUT : $\mathbf{w}^{(T+1)}$
Intialized $\mathcal{A}^n \leftarrow \emptyset$ for each $n = 1 \dots N$
for $t = 1 \dots T$ **do**
  Pick some example (X^n, Y^n) from $\mathcal{D}$
  Sample Y^* according to $\mathcal{Q}(w^{(t)}, Y^n)$
  $\mathcal{A}^n \leftarrow \mathcal{A} \cup \{Y^*\}$
  $\mathcal{A}^{n'} \leftarrow \{Y \in \mathcal{A}^n \mid l(Y, Y^n, \mathbf{w}^{(t)}) > 0\}$
  $\eta^{(t)} \leftarrow \frac{\beta}{t}$
  $\mathbf{z} \leftarrow \mathbf{w}^{(t)}$
  for $Y \in \mathcal{A}^{n'}$ **do**
    $\mathbf{g}^{(t)} \leftarrow \frac{1}{|\mathcal{A}^{n'}|} \frac{\partial f(Y^n, Y, \mathbf{w}^{(t)})}{\partial \mathbf{w}^{(t)}}$
    $\mathbf{z} \leftarrow \mathbf{z} - \eta^{(t)} \mathbf{g}^{(t)}$ * atomic update *
  end for
  $\mathbf{w}^{(t+1)} \leftarrow \mathbf{z}$
end for

Algorithm 1 solves the loss-augmented inference to generate new constraints, which can be expensive to compute. The analysis presented in Section 4 suggests that it is possible to use a sampling method instead of the loss-augmented inference to obtain new constraints, and under similar assumptions the average subgradient $\bar{g}$ still converges to a valid subgradient. Based on this observation, we propose an adaptation of Algorithm 1 that uses sampling instead of solving the loss-augmented inference. This adapation described in Algorithm 2 generates new constraints using an MCMC walker denoted $\mathcal{Q}$ similar to the one described

in [29]. We also replace the standard update of Algorithm 1 by a sequence of atomic updates that has been shown to improve the speed of convergence [29]. Concerning the practicality, we would like to point out that the working set does not lead to a significant increase in memory as we only need to store the feature maps rather than the whole labellings.

5. Experimental Results

We apply our algorithm to image segmentation. We first briefly describe how a CRF model is used for this task and then present our experimental results on two distinct datasets to demonstrate the effectiveness of our method.

5.1. CRF for Image Segmentation

As a standard preprocessing step, we perform a preliminary over-segmentation of our input image into superpixels [1] using SLIC [1]. 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.

The score function associated with the CRF can then be written as

$$S_{\mathbf{w}}(Y) = \sum_{i \in \mathcal{V}} D_i(y_i) + \sum_{(i,j) \in \mathcal{E}} V_{ij}(y_i, y_j), \qquad (14)$$

where D_i is the unary data term and V_{ij} is the pairwise spatial term. Both D_i and V_{ij} are linear in the CRF parameters $\mathbf{w}$ and also depend on the observed data X in addition to the labeling Y. For inference, we use graph cuts when the corresponding energy function (i.e., negated score) is submodular [7] and belief propagation otherwise.

A natural choice for the task loss Δ (Eq. 3) is the per-superpixel 0-1 loss $\Delta(Y^n, Y) = \sum_{i \in \mathcal{V}} \mathcal{I}(y_i \neq y_i^n)$, 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.

5.2. Methods

In the following, we will compare our learning methods (referred as **Working sets + inference** and **Working sets + sampling**) with the following baselines. We also experimented with averaging all past subgradients [17, 30], which did not produce meaningful results for our task.

- **Linear SVM** – A linear SVM classifying each sample independently (i.e., without CRF).

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

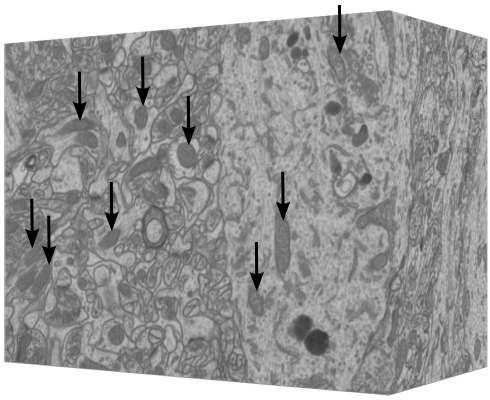


Figure 1. A nearly isotropic stack of neural tissue acquired using EM microscopy annotated for training and testing. This stack contains 1065 images of 2048×1536 pixels, and was used for the task of segmenting mitochondria, indicated by arrows.

- **SSVM** – The cutting plane algorithm described in [28].
- **SampleRank** – The method described in [29].
- **SGD + inference** – solve the loss-augmented inference using graph-cuts or belief-propagation. This algorithm is the SGD (subgradient descent) formulation of [19].
- **SGD + sampling** – Instead of performing inference, use MCMC to sample constraints from a distribution targeting the loss-augmented score. This is equivalent to the method named "SampleRank SVM" described in [29].

In all cases, we used a decreasing step size rule of $\frac{1}{\sqrt{t}}$ and used cross-validation on the training set to determine the regularization constant. The results reported for the sampling method and SampleRank were averaged over 5 runs.

5.3. MSRC Dataset

The MSRC dataset is a popular multi-class object segmentation dataset containing 591 images with objects from 21 categories. Training and testing are done using the standard split of the dataset [23], and we used the annotated images provided by [14]. We extract feature vectors by first over-segmenting images using SLIC superpixels [1]. We then extract SIFT descriptors and color histograms from image patches surrounding each superpixel centroid. We then create a bag-of-words descriptor by first generating a dictionary containing 1,000 words for SIFT features, and 400 words for color histograms are constructed using k-means on extracted features. We also include location information as in [9] and the unary potentials of [8]. The resulting feature vector x_i is used to train the various methods. Similarly to [23], the pairwise term we used was made gradient-adaptive by including parameters for each discretized im-

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. Bold entries are used for each part separately to indicate best performance. Note that all the methods in the top part of the table were optimized for the average score, for which our method achieves the best results. We also include the results of two state-of-the-art algorithms as reported in [31].

	building	grass	tree	cow	sheep	sky	airplane	water	face	car	bicycle	flower	sign	bird	book	chair	road	cat	dog	body	boat	**Average**	Global
Linear SVM	63	88	77	90	85	91	89	59	86	88	94	85	80	53	93	85	79	65	50	82	0	**74.8**	79.6
SSVM [28]	69	91	82	80	82	92	78	68	85	79	85	90	77	58	95	82	87	74	44	80	44	**77.1**	82.3
SampleRank	66	91	80	84	80	92	76	70	82	65	89	91	85	54	93	80	88	69	48	89	40	**76.7**	82.2
SGD + sampling	67	93	80	82	75	93	64	67	84	74	81	93	82	43	93	75	88	68	52	73	48	**75.4**	82.1
SGD + inference [19]	69	91	84	87	82	92	80	64	86	82	85	83	76	61	95	82	88	73	43	74	35	**76.7**	82.2
Working set + sampling	71	90	87	91	84	92	83	74	82	84	85	87	66	46	94	84	88	79	49	74	37	**77.2**	83.3
Working set + inference	67	89	85	93	79	93	84	75	79	87	89	92	71	46	96	79	86	76	64	77	50	**78.9**	83.7
Ladický *et al*. [10]	73	93	82	81	91	98	81	83	88	74	85	97	79	38	96	61	90	69	48	67	18	**75.8**	85.0
Yao *et al*. [31]	67	92	80	82	89	97	86	83	86	79	94	96	85	35	98	70	86	78	55	62	23	**77.4**	84.4

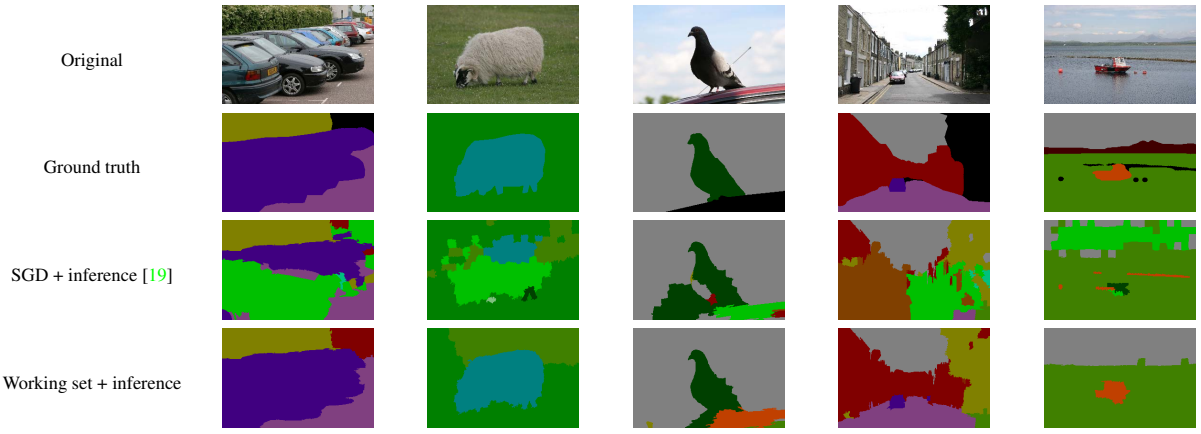


Figure 2. Example segmentations from the MSRC dataset. Best viewed in color.

age gradient level. In a similar fashion, it also considers geometric relationships such as "sky appears above grass".

Table 1 summarizes the segmentation performance of the various approaches and example segmentations appear in Fig. 2. The quantitative results show that the working set of constraints improves the average score regardless of whether inference or sampling was used during learning. The results obtained by the sampling approach are close to those from using inference, but with a significantly lower running time as shown in Table 3. In addition to the baseline methods described above, we compare our approach to state-of-the-art approaches [10, 31]. We achieve the best results in terms of the average score for which we optimize our algorithm.

5.4. Electron Microscopy Dataset

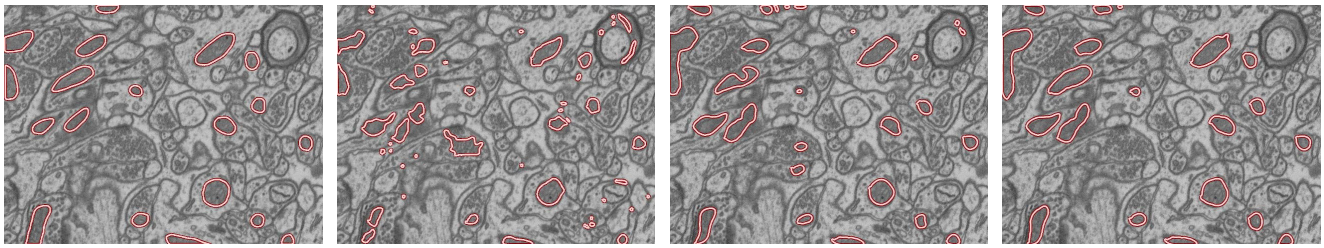
Here, we perform mitochondria segmentation in 3D using the large image stack from Fig. 1. This electron miscropy dataset is publicly available at http://cvlab.epfl.ch/data/em. Performance is measured by the *Jaccard index* commonly used for image segmentation [4]. The Jaccard index is the ratio of the areas of the intersection between what has been segmented and the ground truth, and of their union. It is written as:

$$\text{Jaccard index} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive} + \text{False Negative}} .$$

The segmentation process begins by over-segmenting the volume using SLIC supervoxels [1]. For each supervoxel, we extract a feature vector that captures local shape and texture information using Ray descriptors [13] and intensity histograms. These feature vectors are used to train each baseline method, as well as our model. In a second set of experiments, we also transform the features using the kernel method of [12]. The original feature vectors are 120-dimensional and are thus mapped to a higher dimensional space. The details are described in [12]. Due to the high cost of labeling such large volumes, our experiments are restricted to two subvolumes containing $1024\times768\times165$ voxels. The first subvolume was used to train the various meth-

Table 2. Segmentation performance measured with the Jaccard index for the mitochondria EM dataset. We report results for two different set of features (see text for full description). Note that the original features were already kernelized with a RBF-SVM in [13].

	SVM	Lucchi [13]	SSVM [28]	SampleRank [29]	SGD + sampling	SGD + inference [19]	Working set + sampling	Working set + inference
Original features	73.0%	80.0%	80.5%	81.2%	77.5%	79.9%	83.0%	**84.5%**
Kernelized features	75.4%	-	83.5%	82.9%	80.1%	81.5%	84.4%	**86.7%**

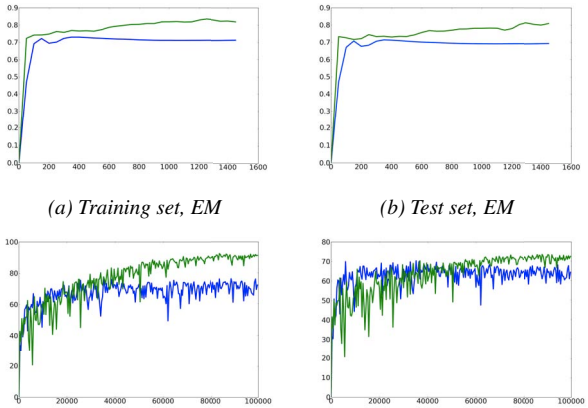


Ground truth | Linear SVM | SGD + inference [19] | Working set + inference

Figure 3. Segmentation results on the EM dataset.

Table 3. Running time for the EM and MSRC datasets for $T = 1000$ iterations. The computational overhead reported in the brackets is the increase in time resulting from the working set. On both datasets, our method achieves better results at the price of a very slight overhead.

	EM	**MSRC**
SampleRank [29]	2524s	80s
SGD + Sampling	2481s	72s
Working set + Sampling	2619s (+5.5%)	76s (+5.2%)
SGD + inference [19]	5315s	546s
Working set + inference	5842s (+9.9%)	583s (+6.8%)



(a) Training set, EM *(b) Test set, EM*

(c) Training set, MSRC *(d) Test set, MSRC*

Figure 4. Evolution of the training and test scores (Jaccard index for EM and average score for MSRC) as a function of the number of iterations t. We report results for the sampling method with and without working set in green and blue respectively.

ods while the second one was used for testing. Each subvolume contains $\sim$13K supervoxels. The resulting graphs have $\sim$91K edges. Example segmentations are shown in Fig. 3 and quantitative results are provided in Table 2. The increased reliability due to the use of working sets leads to higher scores for both the inference and sampling methods. The inference version of our algorithm outperforms the previous state-of-the-art [13].

5.5. Time analysis

We conducted a time analysis of the standard subgradient method of [19] against the 2 versions of the algorithm introduced in this paper. As shown in Table 3, the sampling method is much faster than solving the loss-augmented inference to find the most violated constraint. We can see that the computational overhead due to the working set is of the order of 5% for the sampling method and less than 10% when solving the loss-augmenting inference to find the most-violated constraint. The evolution of the training scores as a function of the number of iterations is shown on Fig. 4 for the EM and MSRC datasets. The curves clearly show that the working set of constraints leads to a much higher score on both the training and test sets.

6. Conclusion

We have presented a working set based approximate subgradient descent method for learning graphical models for structured prediction. Our method is particularly appealing for learning large CRFs with loops, which are common in computer vision tasks, since under these circumstances the use working sets of constraints produces better subgradient estimates and higher-quality solutions. We applied our method the the task of image segmentation, where the results show that our method compares favorably against previous methods in terms of segmentation accuracy. Moreover the method allows us to use sampling to replace the more expensive inference step without much performance loss, leading to significantly lower learning time.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Suesstrunk. SLIC Superpixels Compared to State-Of-The-Art Superpixel Methods. *PAMI*, 2012. 5, 6

[2] J. Besag. Spatial Interaction and the Statistical Analysis of Lattice Systems. *J. Royal Stat. Soc., B*, 36(2), 1974. 1, 2

[3] Y. Boykov and M. Jolly. Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images. In *ICCV*, pages 105–12, 2001. 1

[4] M. Everingham, C. W. L. Van Gool and, J. Winn, and A. Zisserman. The Pascal Visual Object Classes Challenge 2010 (VOC2010) Results. 6

[5] T. Finley and T. Joachims. Training Structural SVMs When Exact Inference is Intractable. In *ICML*, 2008. 2

[6] W. Hoeffding. Probability Inequalities for Sums of Bounded Random Variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963. 4

[7] V. Kolmogorov and R. Zabih. What Energy Functions Can Be Minimized via Graph Cuts? *PAMI*, 26(2):147–159, 2004. 5

[8] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*, 2011. 5

[9] L. Ladicky, C. Russell, P. Kohli, and P. H. S. Torr. Associative Hierarchical CRFs for Object Class Image Segmentation. In *ICCV*, 2009. 5

[10] L. Ladickỳ, P. Sturgess, K. Alahari, C. Russell, and P. Torr. What, Where and How Many? Combining Object Detectors and Crfs. In *ECCV*, pages 424–437, 2010. 6

[11] J. Lafferty, A. Mccallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, 2001. 1, 2

[12] A. Lucchi, Y. Li, K. Smith, and P. Fua. Structured Image Segmentation Using Kernelized Features. In *ECCV*, October 2012. 6

[13] A. Lucchi, K. Smith, R. Achanta, G. Knott, and P. Fua. Supervoxel-Based Segmentation of Mitochondria in EM Image Stacks with Learned Shape Features. *TMI*, 31(2):474–486, 2011. 6, 7

[14] T. Malisiewicz and A. Efros. Improving Spatial Support for Object via Multiple Segmentations. In *BMVC*, 2007. 5

[15] D. Mcallester, T. Hazan, and J. Keshet. Direct Loss Minimization for Structured Prediction. In *NIPS*, 2010. 1, 2, 3

[16] K. Murphy, Y. Weiss, and M. Jordan. Loopy Belief Propagation for Approximate Inference: An Empirical Study. In *UAI*, 1999. 1

[17] Y. Nesterov. Primal-Dual Subgradient Methods for Convex Problems. *Math. Program.*, 120(1):221–259, April 2009. 1, 3, 5

[18] J. Platt. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*. MIT Press, 1998. 2

[19] N. Ratliff, J. A. Bagnell, and M. Zinkevich. (online) Subgradient Methods for Structured Prediction. In *AISTATS*, 2007. 1, 2, 3, 4, 5, 6, 7

[20] S. M. Robinson. Linear convergence of epsilon-subgradient descent methods for a class of convex functions. *Mathematical Programming*, 86:41–50, 1999. 4

[21] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: Primal Estimated Sub-Gradient Solver for SVM. *Math. Program.*, 127(1):3–30, March 2011. 2

[22] N. Shor, K. Kiwiel, and A. Ruszcayǹski. *Minimization Methods for Non-Differentiable Functions*. Springer-Verlag New York, Inc., 1985. 4

[23] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context. *IJCV*, 81(1), January 2009. 5

[24] C. Sutton and A. Mccallum. Introduction to conditional random fields for relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2006. 2

[25] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning Structured Prediction Models: A Large Margin Approach. In *ICML*, 2005. 2

[26] B. Taskar, C. Guestrin, and D. Koller. Max-Margin Markov Networks. In *NIPS*, 2003. 1, 2, 3

[27] B. Taskar, S. Lacoste-julien, and M. Jordan. Structured Prediction, Dual Extragradient and Bregman Projections. *JMLR*, 7:1627–1653, 2006. 2

[28] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support Vector Machine Learning for Interdependent and Structured Output Spaces. In *ICML*, 2004. 1, 2, 3, 5, 6, 7

[29] M. Wick, K. Rohanimanesh, K. Bellare, A. Culotta, and A. Mccallum. Samplerank: Training Factor Graphs with Atomic Gradients. In *ICML*, 2011. 2, 5, 7

[30] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *JLMR*, 11:2543–2596, Dec. 2010. 1, 3, 5

[31] J. Yao, S. Fidler, and R. Urtasun. Describing the Scene as a Whole: Joint Object Detection, Scene Classification and Semantic Segmentation. In *CVPR*, pages 702–709, 2012. 6