Efficient optimization for probably submodular constraints in CRFs

Maxim Berman **Matthew B. Blaschko**
Center for Processing Speech and Images
Dept. of Electrical Engineering
KU Leuven
{maxim.berman, matthew.blaschko}@kuleuven.be

Abstract

Constructive machine learning can frequently be formulated in the setting of structured prediction. Structured prediction is typically modeled using a compatibility function between inputs and outputs and a decoding step, in which an optimization over the compatibility function with respect to the output is performed. When the compatibility function is equivalent to an (unnormalized) probability in a graphical model such as a random field, decoding can be seen as equivalent to MAP inference in that model. Random field models with loops require constraints, such as submodularity of the pairwise potentials, to ensure feasible test time inference. Recently, a framework has been proposed in which a discriminative learning phase only probabilistically guarantees submodularity, enlarging the model space and explicitly trading off between model error and inference error (Zaremba and Blaschko, 2016). A difficulty with this framework is that the optimization requires the enforcement of a set of constraints whose size is proportional to the number of edges in all instances in the training set. In e.g. vision applications such as segmentation, thousands of megapixel images in the training set can lead to billions of constraints during optimization. In this work, we show that a delayed constraint generation framework built on a simple application of the Cauchy-Schwartz inequality and inexact pretraining leads to substantial reduction in computational requirements for *exact* application of the framework. An experimental evaluation shows the computational efficiency of the proposed optimization framework.

1 Introduction

Learning structure in high-dimensional probability spaces lies at the core of machine learning. Efficiently capturing the relevant probabilistic dependencies in data tackles computational issues – fighting the curse of dimensionality, training issues – ensuring generalizability, and semantic issues – finding sensible relationships between objects. In constructive machine learning, the rich structure of the input-output space is itself the main object of interest of the learning process, with large or infinite spaces i.e. of graphs, molecules, or artistic compositions.

Probabilistic graphical models [6] are natural candidates for introducing structure in probability spaces and address a constructive learning task in a principled way. The usability of graphical models is however limited by the intractability of inference in general conditional random fields (CRFs) [3]. Approaches to address this limitation include restricting graph topologies and resorting to approximate inference algorithms. An alternative strategy is to put restrictions on the parameters $\boldsymbol{\theta}$ of the model $f(\mathbf{x};\boldsymbol{\theta}) \rightarrow \mathbf{y}$ to ensure tractable inference of $\mathbf{y}$ for all possible inputs $\mathbf{x}$. One such restriction commonly used in computer vision is that of submodularity of the potentials, as attractive potentials give rise to fast minimization procedures in arbitrary graph topologies, such as graph cuts [4] for binary images. However, these tractability constraints heavily restrict the model.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

In this work, we adopt the framework of probably submodular CRFs introduced by Zaremba & Blaschko [11], which results in a safe and principled strategy to relax the tractability constraints, thereby enlarging the model class, while still *probabilistically* enforcing tractability. Our contribution is a delayed constraint generation scheme that reduces the computational cost incurred by the generation of probably submodular constraints, paving the way to large-scale optimization.

2 Discriminative training of probably submodular models

We use the learning framework of Structured Output Support Vector Machines (SSVM) [10], a large-margin classifier suited to the prediction of complex structured outputs such as image segmentation masks. Given an input $x \in \mathcal{X}$, the output of the model is the solution to the *MAP inference problem*

$$y^* = \arg\max_{y \in \mathcal{Y}} \mathbf{w}^\intercal \boldsymbol{\phi}(x, y) \tag{1}$$

where $\mathbf{w}$ is the weight vector of the model and $\boldsymbol{\phi}(x, y)$ the joint feature map encoding the joint structure of the input-output space. The weights of the model are learned from a training set $\mathcal{S}$ of n training samples $(x_i, y_i)_{i=1\dots n}$ following the regularized large-margin objective

$$\min_{\mathbf{w},\xi} \ \frac{1}{2}\|\mathbf{w}\|^2 + C\xi \tag{2}$$

$$\text{s.t. } \forall (\bar{y}_1, \dots, \bar{y}_n) \in \mathcal{Y}^n : \ \frac{1}{n}\mathbf{w}^\intercal \sum_{i=1}^{n} (\boldsymbol{\phi}(x_i, y_i) - \boldsymbol{\phi}(x_i, \bar{y}_i)) \geq \frac{1}{n}\sum_{i=1}^{n} \Delta(y_i, \bar{y}_i) - \xi \tag{3}$$

in the 1-slack margin-rescaling formulation [5], where the loss $\Delta(y_i, y)$ quantifies the dissimilarity of outputs. Despite the exponential number of constraints $|\mathcal{Y}|^n$, the quadratic program (QP) (2) can be solved, e.g. in a cutting-plane approach [10, 5], with polynomial number of calls to the *augmented inference max-oracle* $\arg\max_{y \in \mathcal{Y}} \mathbf{w}^\intercal \boldsymbol{\phi}(x_i, y) + \Delta(y_i, y)$.

In the case of pairwise CRF models, $\boldsymbol{\phi}$ decomposes over the vertices $\mathcal{V}$ and edges $\mathcal{E}$ of the input graph [9], and the weight vector parametrizes the log-linear energy of the network

$$E_{\mathcal{V},\mathcal{E}}(x, y) = -\mathbf{w}^\intercal \boldsymbol{\phi}(x, y) = -\mathbf{w}_u^\intercal \sum_{k \in V} \boldsymbol{\phi}_{\mathcal{V}}(x^k, y^k) - \mathbf{w}_p^\intercal \sum_{(k,l) \in E} \boldsymbol{\phi}_{\mathcal{E}}(x^k, y^k, x^l, y^l). \tag{4}$$

In a segmentation task, x^k, x^l are adjacent (super-)pixels, and y^k, y^l their corresponding labels. We write the joint feature maps as Kronecker products [7]

$$\boldsymbol{\phi}_{\mathcal{V}}(x^k, y^k) = \mathbf{1}(y^k) \otimes \boldsymbol{\phi}_u(x^k); \qquad \boldsymbol{\phi}_{\mathcal{E}}(x^k, y^k, x^l, y^l) = \mathbf{1}(y^k) \otimes \mathbf{1}(y^l) \otimes \boldsymbol{\phi}_p(x^k, x^l) \tag{5}$$

where $\mathbf{1} : \mathcal{L} \mapsto \{0, 1\}^{|\mathcal{L}|}$ is a one-hot encoding of the labels $y \in \mathcal{L}$ and $\boldsymbol{\phi}_u$ and $\boldsymbol{\phi}_p$ are the unary and pairwise feature vectors extracted on input x. Accordingly, $\mathbf{w}_u$ decomposes into unary weights $\mathbf{w}_\alpha$ for every label $\alpha \in \mathcal{L}$, and $\mathbf{w}_p$ into $\mathbf{w}_{\alpha,\beta}$ for every $\alpha, \beta \in \mathcal{L}$. The SSVM loss Δ is chosen to decompose over the vertices $\Delta(y_i, y) = \sum_{k \in \mathcal{V}} \delta(y_i, y^k)$, such that the augmented inference reduces to a MAP inference (1) on the graph with modified unaries.

Solving MAP inference in pairwise CRFs is NP-hard in general [1]; however, particular restrictions on the pairwise potentials give rise to efficient algorithms. In particular, imposing submodularity of the pairwise energies of the graph, i.e.

$$\langle \mathbf{w}_{\alpha\alpha}, \boldsymbol{\phi}_p(x^k, x^l) \rangle + \langle \mathbf{w}_{\beta\beta}, \boldsymbol{\phi}_p(x^k, x^l) \rangle \geq \langle \mathbf{w}_{\alpha\beta}, \boldsymbol{\phi}_p(x^k, x^l) \rangle + \langle \mathbf{w}_{\beta\alpha}, \boldsymbol{\phi}_p(x^k, x^l) \rangle \tag{6}$$

for every edge $(k, l) \in \mathcal{E}$ and every pair of labels $\alpha, \beta \in \mathcal{L}$, leads to tractable polynomial-time inference – exact with binary labels (max-flow algorithm), approximate with strong approximation bounds with more than two labels (e.g. with the $\alpha - \beta$ swap algorithm [2]). This holds regardless of the unary energies of the graph, hence the conditions for augmented inference are the same.

The submodularity conditions (6) can be enforced as constraints on the weight vector $\mathbf{w}$. Assuming positive pairwise features $\boldsymbol{\phi}_p(x^k, x^l) \succcurlyeq \mathbf{0}$, Zaremba et al. [11] introduce the set of *definitively submodular constraints*

$$\mathcal{D} = \{\mathbf{w} \mid \mathbf{w}_{\alpha\alpha} \succcurlyeq \mathbf{0} \wedge \mathbf{w}_{\beta\beta} \succcurlyeq \mathbf{0} \wedge \mathbf{w}_{\alpha\beta} \preccurlyeq \mathbf{0} \wedge \mathbf{w}_{\beta\alpha} \preccurlyeq \mathbf{0} \quad \forall\, \alpha \neq \beta \in \mathcal{L}\}$$

enforcing conditions (6) for all inputs $x \in \mathcal{X}$, and the set of *probably submodular constraints*

$$\mathcal{P} = \left\{ \mathbf{w} \;\middle|\; \begin{array}{l} \langle \mathbf{w}_{\alpha\alpha}, \boldsymbol{\phi}_p(x_i^k, x_i^l) \rangle + \langle \mathbf{w}_{\beta\beta}, \boldsymbol{\phi}_p(x_i^k, x_i^l) \rangle \\ \qquad \geq \langle \mathbf{w}_{\alpha\beta}, \boldsymbol{\phi}_p(x_i^k, x_i^l) \rangle + \langle \mathbf{w}_{\beta\alpha}, \boldsymbol{\phi}_p(x_i^k, x_i^l) \rangle \end{array} \quad \forall\, \alpha \neq \beta \in \mathcal{L},\ (k, l) \in \mathcal{E},\ x_i \in \mathcal{S} \right\}$$

enforcing these conditions for the examples $x_i \in \mathcal{S}$ only. Under probably submodular constraints, some edges corresponding to a new test instance may have non-submodular potentials. However, since the set $\mathcal{S}$ captures the probability distribution of the feature vectors, this occurs with low probability. On the other hand, since $\mathcal{D} \subset \mathcal{P}$, moving from definitely submodular to probably submodular constraints increases the model capacity, yielding better performance on the segmentation task [11].

3 Efficient constraint generation

The tractability constraints in sets $\mathcal{D}, \mathcal{P}$ can be written as hard linear constraints $\mathbf{c}^\intercal \mathbf{w} \geq \mathbf{0}$. As such, we can incorporate them in the QP optimization (2). However, $\mathcal{P}$ comprises $(|\mathcal{L}|\cdot(|\mathcal{L}|-1)/2)\cdot|\mathcal{E}|\cdot|\mathcal{S}|$ constraints; even for moderately sized binary segmentation tasks with limited connectivity on small datasets, this large amount cannot be handled by QP solvers. Zaremba et al. [11] address this problem in a cut approach; the most violated constraints are iteratively added to the $\mathbf{w}$-update (QP solver) subroutine of the SSVM until all constraints are satisfied, leading experimentally to a small, manageable, number of constraints added to the QP at any learning iteration.

Noting d the dimension of the pairwise features, let $\mathbf{P}$ be the $|\mathcal{E}| \cdot |\mathcal{S}| \times d$ matrix of all pairwise feature vectors in the training data and $\mathbf{B}$ the $|\mathcal{L}|(|\mathcal{L}|-1)/2 \times |\mathcal{L}|^2$ matrix of rows

$$(\mathbf{1}(\alpha) \otimes \mathbf{1}(\alpha))^\intercal + (\mathbf{1}(\beta) \otimes \mathbf{1}(\beta))^\intercal - (\mathbf{1}(\alpha) \otimes \mathbf{1}(\beta))^\intercal - (\mathbf{1}(\beta) \otimes \mathbf{1}(\alpha))^\intercal$$

for all labels $\alpha \neq \beta$. The constraints in $\mathcal{P}$ take the form $(\mathbf{B} \otimes \mathbf{P})\mathbf{w}_p \geq 0$. As observed in [11], the complexity of computing the constraint margins $(\mathbf{B} \otimes \mathbf{P})\mathbf{w}_p$ can be reduced by observing that $(\mathbf{B} \otimes \mathbf{P})\mathbf{w}_p = \operatorname{vec} \mathbf{V}$, with $\mathbf{V} = \mathbf{P}(\tilde{\mathbf{w}}_p \mathbf{B}^\intercal)$ and $\tilde{\mathbf{w}}_p$ a matrix constructed such that $\operatorname{vec} \tilde{\mathbf{w}}_p = \mathbf{w}_p$. Computing the margins in $\mathbf{V}$ saves a factor of $|\mathcal{L}|^2$ operations.

Even with this acceleration, the computation of the $|\mathcal{E}|\cdot|\mathcal{S}| \times |\mathcal{L}|(|\mathcal{L}|-1)/2$ matrix $\mathbf{V}$ still scales as $\mathcal{O}(|\mathcal{E}|\cdot|\mathcal{S}|\cdot|\mathcal{L}|^2\cdot d)$. On a small-sized problem with 10^3 edges, 200 images, 2 labels and pairwise features of dimension 500, this results in 400 million floating point operations for updating the hard constraints margins after each update of the weight vector $\mathbf{w}$. Computing the most violated hard constraint therefore dominates the learning time, as illustrated by experiments in the next section.

Margin bounds We introduce a delayed constraint generation approach to tackle this issue. The key observation is that in later learning iterations, the optimal weight vector $\mathbf{w}$ does not change drastically. Constraints corresponding to a high positive margin in $\mathbf{V}$ are therefore likely to stay enforced after updating $\mathbf{w}$. Formally, for each probably submodular constraint $c : \mathbf{c}^\intercal \mathbf{w} \geq 0$, we introduce a lower bound on the margin $l_c \leq \mathbf{c}^\intercal \mathbf{w}$. After a weight update $\mathbf{w} \to \mathbf{w}'$, we have

$$\mathbf{c}^\intercal \mathbf{w}' = \mathbf{c}^\intercal \mathbf{w} + \langle \mathbf{w}' - \mathbf{w}, \mathbf{c} \rangle \geq l_c - \|\mathbf{w}' - \mathbf{w}\| \cdot \|\mathbf{c}\|; \tag{7}$$

by application of the Cauchy-Schwartz inequality. Therefore the update $l_c \to l'_c = l_c - \|\mathbf{w}' - \mathbf{w}\| \cdot \|\mathbf{c}\|$ yields a correct new lower bound. We can safely save computations by avoiding the updating of constraint margins that are lower-bounded by a positive value. As before, we write the operations in matrix form. We store the norm of all constraints, and the margin of lower bounds (initialized to $-\infty$), in two matrices $\mathbf{N}$ and $\mathbf{L}$ of same size as $\mathbf{V}$. By storing the results of margin computations in $\mathbf{L}$, raising the bound to the actual value, we avoid referring to $\mathbf{V}$ altogether. Algorithm 1 presents the resulting algorithm integrated in the $\mathbf{w}$-update subroutine, called at each iteration t of the SSVM.

Algorithm 1: Accelerated $\mathbf{w}$-update solver subroutine of probably submodular SSVM

input : SSVM constraints $\mathcal{C}^{(t)}$ at iteration t; constraint matrices $\mathbf{P}, \mathbf{B}, \mathbf{N}$; current $\mathbf{w}$, current bounds $\mathbf{L}$
output : optimal $\mathbf{w}^*$ satisfying $\mathcal{C}^{(t)}$ and submodular constraints; new bounds $\mathbf{L}$

Loop
- $(\mathbf{w}^*, \xi^*) \leftarrow \arg\min_{(\mathbf{w},\xi)} \{\|\mathbf{w}\|^2/2 + C\xi\}$ s.t. $(\mathbf{w}, \xi) \in \mathcal{C}^{(t)}$ *// QP solver*
- $\mathbf{L} \leftarrow \mathbf{L} - \|\mathbf{w}^* - \mathbf{w}\| \cdot \mathbf{N}$ *// update bounds*
- $\mathbf{W} \leftarrow \tilde{\mathbf{w}}_p \mathbf{B}^\intercal$
- **for** *(i,j) such that* $L_{i,j} \leq 0$ **do**
  - $L_{i,j} \leftarrow \sum_k P_{i,k} W_{k,j}$ *// compute margins*
- $(i, j) \leftarrow \arg\min \mathbf{L}$
- **if** $L_{i,j} \geq 0$ **then return** $\mathbf{w}^*, \mathbf{L}$ *// all bounds positive*
- **else** $\mathcal{C}^t = \mathcal{C}^t \cup \{\text{constraint } (\mathbf{b}_j \otimes \mathbf{p}_i)\mathbf{w}_p \geq 0\}$ *// most violated constraint*

Pretraining In initial iterations of the learning procedure, $\mathbf{w}$ changes significantly and most of the constraints have to be recomputed. To mitigate this, we use a two-stage learning. First, the weights and dual variables of the SSVM are trained until convergence with no submodular constraints – resulting in an inexact truncated graph-cut inference. Second, we enfore submodularity with the above approach, with exact inference. The SSVM converges to the same global optimum, but the pretraining warm-starts the exact learning closer to convergence.

4 Evaluation

We evaluate the computational gains of our method on a 10-fold cross validation of the TU Darmstadt cows dataset, reproducing the experimental setup of [11]. The learning algorithms were implemented on top of the Python structured prediction module `pystruct` [8].

Table 1: Training efficiency gains with/without delayed constraint generation and 2-stage weights pretraining

method	# margins computed	SSVM iterations	total constraint gen. time	total training time
def. submodular		296		607 s
1-pass, full [11]	$102.5 \cdot 10^6$	581	400 s	593 s
1-pass, delayed	$67.9 \cdot 10^6$	581	329 s	652 s
2-pass delayed	$6.5 \cdot 10^6$	658	41 s	555 s

Table 1 shows the improvements in computational efficiency of our delayed constraint generation scheme and the 2-stage training, for one fold of the data. Figure 1 shows the number of computed constraint margins. Combining inexact pretraining and delayed constraint generation limits the number of computed margins. In the first SSVM iteration, many hard constraints have to be added to make $\mathbf{w}$ satisfy the constraints $\mathcal{P}$; in subsequent iterations, the number of added constraints per iteration becomes small (1 or 2). This explains the inflection point in the convergence plots.

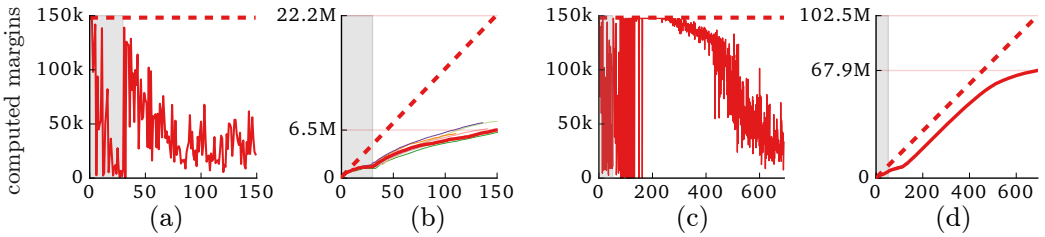

Figure 1: Number of constraint margins computed with (solid) or without (dashed) delayed constraint generation, (a): per call to the QP solver, (b): running total. (a) shows only one fold of the dataset, (b) also shows other folds, which have similar behavior. Shaded area: first SSVM iteration. (c), (d): margins computation without inexact pretraining: in this case, the delayed constraints approach saves much less computations.

Our approach significantly reduces the probably submodular constraints generation time, which scales poorly with the training set size. Although the total training time is only marginally reduced in this small setting, we expect the gain to be significant for larger-scale problems. For completeness, we present in Table 2 the performance of the probably submodular framework in our setting.

Table 2: Evaluation of the probably submodular framework, verifying that the computational speedup proposed in this work results in exact optimization of the objective.

constraint	non-submodular edges (%)	absolute precision (%)	average precision (%)
$\mathbf{w} \in \mathcal{P}$	0.004 ± 0.008	92.77 ± 0.60	89.17 ± 0.89

Conclusion We have showed that the computation cost incurred by probably tractable constraints generation during probably submodular training can be reduced by delaying constraint generation. Future developments include extensive evaluation of the probably submodular framework on competitive computer vision datasets, in particular in multi-label segmentation settings, and generalizations of the approach to richer classes of potential functions, such as neural networks.

Code available on `https://github.com/bermanmaxim/probsub/`.

Acknowledgements This work is partially funded by Internal Funds KU Leuven and FP7-MC-CIG 334380. We acknowledge support from the Research Foundation - Flanders (FWO) through project number G0A2716N.

References

[1] F. Barahona. On the computational complexity of ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241, 1982.

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001.

[3] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[4] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 271–279, 1989.

[5] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, Oct. 2009.

[6] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[7] J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley & Sons, 1995.

[8] A. C. Müller and S. Behnke. pystruct - learning structured prediction in python. *Journal of Machine Learning Research*, 15:2055–2060, 2014.

[9] M. Szummer, P. Kohli, and D. Hoiem. Learning crfs using graph cuts. In D. A. Forsyth, P. H. S. Torr, and A. Zisserman, editors, *European Conference on Computer Vision*, volume 5303 of *Lecture Notes in Computer Science*, pages 582–595. Springer, 2008.

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

[11] W. Zaremba and M. B. Blaschko. Discriminative training of CRF models with probably submodular constraints. In *IEEE Winter Conference on Applications of Computer Vision*. 2016.