Efficient Training of Structured SVMs via Soft Constraints

Ofer Meshi
TTI Chicago

Nathan Srebro
TTI Chicago

Tamir Hazan
University of Haifa, Israel

Abstract

Structured output prediction is a powerful framework for jointly predicting interdependent output labels. Learning the parameters of structured predictors is a central task in machine learning applications. However, training the model from data often becomes computationally expensive. Several methods have been proposed to exploit the model structure, or decomposition, in order to obtain efficient training algorithms. In particular, methods based on linear programming relaxation, or dual decomposition, decompose the prediction task into multiple simpler prediction tasks and enforce agreement between overlapping predictions. In this work we observe that relaxing these agreement constraints and replacing them with soft constraints yields a much easier optimization problem. Based on this insight we propose an alternative training objective, analyze its theoretical properties, and derive an algorithm for its optimization. Our method, based on the Frank-Wolfe algorithm, achieves significant speedups over existing state-of-the-art methods without hurting prediction accuracy.

1 Introduction

Structured prediction drives many applications of machine learning, including computer vision, natural language processing, and computational biology. Examples include object detection (Felzenszwalb et al., 2010), parsing (Koo et al., 2010), and protein design (Sontag et al., 2008). In this setting data instances are mapped to labels with rich internal structure, whether objects, parse trees, or molecular structures. Although one may decompose the prediction task into multiple independent predictions, it is often better to jointly predict the structured labels in order to account for the correlations between labels, and thus improve prediction accuracy.

In order to achieve high prediction accuracy, the parameters of structured predictors are learned from training data. In particular, in the *Structured SVM* framework, the learning objective is formulated as regularized structured hinge loss minimization (Collins, 2002; Taskar et al., 2003; Tsochantaridis et al., 2004). Despite the convexity of the structured SVM objective function, finding the optimal parameters of these models is computationally expensive, since it requires comparing training labels with predicted labels. For some specific models (e.g., tree-structured graphs, matchings, and supermodular scores), exact prediction can be done efficiently, however, in general computing the objective or the gradient exactly is intractable. Therefore, one usually resorts to approximate inference algorithms (cf. Taskar et al., 2003; Kulesza and Pereira, 2007; Finley and Joachims, 2008). One family of such approximations that has proved quite successful is based on *linear programming (LP) relaxation*, or *dual decomposition*. In this approach the intractable prediction task is decomposed into simpler prediction tasks, and consistency among overlapping predictions is enforced. Although tractable, these algorithms are often very expensive when used as a subroutine within the learning algorithm.

In this work we propose a novel training algorithm for structured SVMs. We observe that the hard consistency constraints between overlapping predictions are computationally expensive. Instead, we suggest to enforce these constraints in a soft manner, by introducing a penalty term that accounts for constraint violation. We provide theoretical guarantees for the soft training objective, and then derive an efficient algorithm for its optimization based on the Frank-Wolfe algorithm. After analyzing the convergence rate of the proposed algorithm, we proceed to evaluate its performance empirically. Our experiments show sig-

Appearing in Proceedings of the 18^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2015, San Diego, CA, USA. JMLR: W&CP volume 38.

nificant speedups compared to other state-of-the-art structured SVM trainers, especially when training instances pose challenging inference problems. Importantly, this improved runtime comes with no loss in prediction accuracy.

2 Related Work

Learning structured output predictors was proposed as an extension of binary SVMs in Collins (2002); Taskar et al. (2003); Tsochantaridis et al. (2004). These works formulate learning of structured SVMs as a convex program, and in the last decade numerous algorithms have been proposed for solving it. Those include primal methods like cutting plane (Tsochantaridis et al., 2004), structured perceptron (Collins, 2002) and stochastic gradient descent (SGD)[1] (Ratliff et al., 2007; Shalev-Shwartz et al., 2011), as well as dual methods like structured sequential minimal optimization (Taskar et al., 2003), exponentiated gradient (Collins et al., 2008), and the Frank-Wolfe algorithm (Lacoste-Julien et al., 2013). The faster of these methods achieve a convergence rate of $O(1/\epsilon)$, however they all require a call to the so-called *maximization oracle* at each iteration, which is NP-hard in general. Consequently, several works proposed to replace the maximization oracle with linear program relaxations thus making learning structured SVMs tractable for general problems (Taskar et al., 2003; Kulesza and Pereira, 2007; Finley and Joachims, 2008). Nevertheless, the execution of LP solvers as a subroutine is the computational bottleneck of these approaches, and can get quite expensive when training instances are large and complicated.

Recently, several works proposed to decompose the LP within structured SVMs using dual losses (Meshi et al., 2010; Hazan and Urtasun, 2010; Komodakis, 2011). This dual decomposition approach allows to learn structured predictors over general graphs without relying on computationally expensive LP solvers as subroutines. Instead, it requires only cheap local updates that respect the structure of the decomposition. Our method is similar in the sense that it requires only cheap local updates that exploit the structure of the model. However, while previous approaches rely on gradient based optimization and heavily depend on their learning rate (i.e., the gradient step size), ours is a dual method based on Frank-Wolfe optimization (Frank and Wolfe, 1956; Lacoste-Julien et al., 2013). This has the advantage of not requiring tuning of the learning rate, and it allows us to use a sound stopping condition by calculating the duality gap.

Our approach is based on softly enforcing convex program constraints when learning structured predictors. This is known as the *penalty method*, and has a long history in optimization (see Boukari and Fiacco, 1995, for a survey). Recently, Belanger et al. (2014) proposed to replace hard agreement constraints with soft constraints in the context of the prediction task (i.e., MAP inference). Specifically, they introduce a non-smooth penalty term for constraint violation, which results in box-constraints on dual variables. Similar to our work, they show that optimizing the soft objective can be much faster than optimizing the hard-constrained one. Our work differs, since we focus on the learning problem rather than prediction. Also, our penalty terms are smooth, which leads to a different objective function and faster convergence guarantees.

3 Problem Formulation

Consider a supervised learning setting with data instances x and labels y. In structured output learning, the possible structures are incorporated into the high-dimensional labels $y = (y_1, \dots, y_n)$. In this supervised learning setting, training data $\{(x^{(m)}, y^{(m)})\}_{m=1}^M$ is used to learn the parameters $w \in \mathbb{R}^d$ of the linear prediction rule $y(x; w) = \text{argmax}_y\, w^\top \phi(x, y)$, where $\phi(x, y) \in \mathbb{R}^d$ is a function that maps data-label pairs to a feature vector. The goodness of fit is measured by the empirical risk, i.e., the average prediction loss $\Delta(y(x^{(m)}; w), y^{(m)})$ over the training data. Due to the non-convexity of the empirical risk, structured SVMs upper bound the task loss by a convex surrogate called the structured hinge loss (Taskar et al., 2003; Tsochantaridis et al., 2004). This yields the objective:

$$\min_w \; \frac{\lambda}{2}\|w\|^2 + \frac{1}{M}\sum_m \max_y \left[w^\top \phi(x^{(m)}, y) + \Delta(y, y^{(m)}) \right] - w^\top \phi(x^{(m)}, y^{(m)}) \qquad (1)$$

Since the space of possible outputs $y = (y_1, ..., y_n)$ is exponential in n, the maximization over outputs will generally be intractable. In many applications, it is common to assume that the score function $w^\top \phi(x, y)$ *decomposes* into simpler score functions with respect to subsets of indexes $\alpha \subset \{1, ..., n\}$, namely $w^\top \phi(x, y) = \sum_\alpha w_\alpha^\top \phi_\alpha(x, y_\alpha)$. Such decomposed scores consider only (possibly overlapping) subsets of output variables $y_\alpha = \{y_i\}_{i \in \alpha}$. Assuming that the task loss Δ decomposes in a similar manner, one can write the maximization problems for prediction and training in the form $\max_y \sum_\alpha \theta_\alpha(y_\alpha)$. For example, in many applications the model assigns scores to single and pairs of output variables that correspond to nodes and edges of an undirected graph $\mathcal{G}$:

[1] More precisely, in this context it is a stochastic *subgradient* algorithm since the objective is non-differentiable, but, as is common, we use SGD.

$\sum_{ij\in E(\mathcal{G})}\theta_{ij}(y_i,y_j)+\sum_{i\in V(\mathcal{G})}\theta_i(y_i)$. For some decompositions, such as tree-structured graphs and supermodular potential functions, the maximization over outputs can be solved exactly and efficiently. However, this problem is generally NP-hard, and some kind of approximation will be necessary.

One approach to relax the hard learning problem of Eq. (1) is to replace the computationally intractable structured hinge loss for each training example by its (relaxed) dual (Taskar et al., 2005; Meshi et al., 2010). Dividing the subsets α into singletons, denoted by $i = 1, ..., n$, and high-order subsets c (also called "factors"), the resulting training problem is:

$$G: \quad \min_{w,\delta} g(w,\delta) \;\; := \tag{2}$$

$$\frac{\lambda}{2}\|w\|^2 + \frac{1}{M}\sum_m \left[\sum_i \max_{y_i}\left(\theta_i^{(m)}(y_i;w) + \sum_{c:i\in c}\delta_{ci}^{(m)}(y_i)\right) + \sum_c \max_{y_c}\left(\theta_c^{(m)}(y_c;w) - \sum_{i:i\in c}\delta_{ci}^{(m)}(y_i)\right)\right],$$

where $\theta_\alpha^{(m)}(y_\alpha;w) = w_\alpha^\top\left(\phi_\alpha(x^{(m)},y_\alpha) - \phi_\alpha(x^{(m)},y_\alpha^{(m)})\right) + \Delta(y_\alpha,y_\alpha^{(m)})$ for all $\alpha \in \{c,i\}$, and w_α is the set of parameters pertaining to a particular variable or factor. In this formulation the primal variables[2] $\delta_{ci}^{(m)}(y_i)$ encourage agreement between the maximizing argument of a factor and the maximizing arguments of its corresponding variables. For each sample m there exists such variable for a factor c, variable $i \in c$, and assignment y_i (cf. Sontag et al., 2011).

In many convex optimization problems the dual function is easier to optimize than the primal. The dual problem associated with problem G takes the form:

$$F: \quad \max_{\mu\in\mathcal{M}_\mathcal{L}^\times} f(\mu) \;\; := \;\; \mu^\top\ell - \frac{\lambda}{2}\|\Psi\mu\|^2 \; , \tag{3}$$

where μ is the set of dual variables, $\Psi_{m,\alpha,y_\alpha} = \frac{1}{\lambda M}\left(\phi_\alpha(x^{(m)},y_\alpha^{(m)}) - \phi_\alpha(x^{(m)},y_\alpha)\right)$ is a column vector in $\mathbb{R}^d$, and $\ell_{m,\alpha,y_\alpha} = \frac{1}{M}\Delta(y_\alpha,y_\alpha^{(m)})$ is a scalar. In this formulation the dual variables $\mu_\alpha^{(m)}(y_\alpha)$ can be viewed as the marginal probability of the subset assignment y_α (see, e.g., Taskar et al., 2003; Wainwright and Jordan, 2008). Furthermore, the constraint set $\mathcal{M}_\mathcal{L}^\times$, known as the *local marginal polytope*, is a product domain which enforces agreement between local marginals within each training example:[3]

$$\mathcal{M}_\mathcal{L}^{(m)} = \left\{ \mu^{(m)} \geq 0 : \begin{array}{ll} \mu_c^{(m)}(y_i) = \mu_i^{(m)}(y_i) & \forall c, i\in c, y_i \\ \sum_{y_\alpha}\mu_\alpha^{(m)}(y_\alpha) = 1 & \forall\alpha\in\{c,i\} \end{array} \right\} , \tag{4}$$

where $\mu_c^{(m)}(y_i) = \sum_{y_{c\setminus i}}\mu_c^{(m)}(y_c)$ is the marginal of the variable assignment y_i taken from the factor marginal $\mu_c^{(m)}$. Finally, by strong duality we have the mapping $w(\mu^*) = \Psi\mu^*$ from dual to primal optimal solutions.

Fortunately, training the model with the relaxed objective (Eq. (3) or Eq. (2)) often yields very accurate predictors (Kulesza and Pereira, 2007; Finley and Joachims, 2008). Motivated by this success, in this work our goal is to solve this relaxed training problem efficiently. Focusing on the dual problem F, our main insight is that part of the computational difficulty in optimizing this objective stems from the fact that the marginals associated with a training example $\mu^{(m)}$ are coupled together through hard agreement constraints in $\mathcal{M}_\mathcal{L}$. Therefore, in what follows we alleviate this complication by introducing soft agreement constraints, which facilitates more efficient training.

4 Learning with Soft Constraints

In this section we present an alternative training objective that softly enforces the marginalization constraints in Eq. (4). By penalizing violation of those constraints we are able to learn structured predictors effectively while avoiding the costly hard constraints. Moreover, below we show that although we enforce the constraints only softly, we can still control the accuracy of learning.

The penalty method is a generic approach for constrained optimization in which some constraints are replaced by a penalty term for violation of those constraints (Boukari and Fiacco, 1995). Formally, consider the general optimization problem $\max_{\mu\in\mathcal{S}} f(\mu)$ s.t. $A\mu = 0$, and replace the constraint with a penalty term: $\max_{\mu\in\mathcal{S}} f(\mu) - \frac{1}{2\rho}\|A\mu\|^2$. Here ρ is a parameter that controls the strength of the penalty. Clearly, as $\rho \to 0$ the penalty increases and the solution of the unconstrained problem converges to a solution for the original constrained problem.

Applying this idea to the marginalization constraints in F (see Eq. (4)) means replacing the constraint $\mu_c^{(m)}(y_i) = \mu_i^{(m)}(y_i)$ with a penalty term of the form $\frac{1}{2\rho}\left(\mu_c^{(m)}(y_i) - \mu_i^{(m)}(y_i)\right)^2$ for all $m, c, i \in c, y_i$. As before, we can write the resulting problem concisely as:

$$F_\rho: \quad \max_{\mu\in\mathcal{S}^\times} f_\rho(\mu) \;\; := \;\; \mu^\top\ell - \frac{\lambda}{2}\|\Psi\mu\|^2 - \frac{\rho}{2}\|A\mu\|^2 \; , \tag{5}$$

where[4] $(A\mu)_{m,c,i,y_i} = \frac{1}{\rho M}\left(\mu_c^{(m)}(y_i) - \mu_i^{(m)}(y_i)\right)$, and $\mathcal{S}^\times$ is a product domain with per-factor simplex constraints (see Eq. (4)). Intuitively, the additional

[2] In the context of MAP inference δ are usually treated as dual variables, but here we use them in the primal.

[3] Although we focus here on the first-order LP relaxation, our results can be easily extended to tighter relaxations (Sontag et al., 2008; Werner, 2008).

[4] We scale A by $1/M$ to get a similar form as Ψ. This is used in our analysis later on.

penalty term serves to "smooth" the boundaries of the local marginal polytope constraints in F, while keeping the feasible domain $\mu \in \mathcal{M}_{\mathcal{L}}^{\times}$ unchanged. Notice that in this formulation each factor is constrained *independently* so there is no coupling between factors. In Section 5 we use this to derive an efficient training algorithm, but first we study the theoretical properties of the proposed objective.

4.1 Analysis

To better understand learning with soft constraints (Eq. (5)), we begin by examining the dual of F_ρ, which takes the form (see Appendix A):

$$G_\rho : \quad \min_{w,\delta} g_\rho(w,\delta) \; := \; g(w,\delta) + \frac{\rho}{2}\|\delta\|^2 \; . \qquad (6)$$

Strong duality provides the primal-dual mapping $\delta(\mu) = A\mu$, and $w(\mu) = \Psi\mu$. This problem is the same as the primal G, except for the additional L_2 term for δ. Notice that this regularization makes the primal g_ρ strongly convex in both w and δ, whereas g is strongly convex in w, but only piecewise linear in δ. This is important from an optimization point of view, since strong convexity may lead to better convergence rates (cf. Shamir and Zhang, 2013). We next justify the use of the softly constrained objective by bounding its difference from the constrained one.

In the next theorem we use the following notation. Let[5] $\|w\|_2 \leq B$ for all w, let $\|\phi(x,y)\|_2 \leq R$ for all (x,y), and let $\Delta(y,y') \leq L$ for all (y,y'). Therefore, $\|\theta\|_\infty \leq 2BR + L$. In addition, let $|Y_i|$ be the number of states of output variable i, and let $Y_{\max} = \max_i |Y_i|$ denote the maximum over all variables. Finally, let q be the maximal number of factors (including singletons) in any instance.

Theorem 4.1. *Let g_ρ^* be the optimal value of G_ρ, and let g^* be the optimal value of G. Then $g_\rho^* - \frac{\rho}{2}h \leq g^* \leq g_\rho^*$, where $h = M(8Y_{\max}q(BR+L))^2$.*

The proof is given in Appendix B. This theorem shows that despite using soft constraints, we still have guarantees w.r.t. the original constrained objective. We point out that such result does not hold in general, and our proof makes use of special properties of the objective function (see Appendix B). At first glance, the bound in Theorem 4.1 may seem quite loose due to the linear dependence on the number of samples M. However, recall that the difference between g_ρ and g is the L_2 regularization term for δ. Unlike the weight vector w, the number of agreement variables (length of δ) grows with M, and therefore its norm also grows linearly with M. In Section 6 we demonstrate that this is not a serious limitation of our approach, since in practice the difference between the soft- and hard-constrained solutions is not so large, even for relatively high values of ρ.

The previous theorem shows that the soft optimum is not far from the hard constrained one. We can actually obtain a similar result for a near-optimal solution.

Theorem 4.2. *Let μ_ρ^ϵ be a dual solution to F_ρ for which the duality gap is bounded: $D_\rho(\mu_\rho^\epsilon) = g_\rho(w(\mu_\rho^\epsilon), \delta(\mu_\rho^\epsilon)) - f_\rho(\mu_\rho^\epsilon) \leq \epsilon$. Then $(w(\mu_\rho^\epsilon), \delta(\mu_\rho^\epsilon))$ is $\left(\epsilon + \frac{\rho}{2}h\right)$-optimal for G.*

The proof is given in Appendix C. Theorem 4.2 implies that in order to get an $\hat{\epsilon}$-optimal solution to G, it is enough to set $\rho = \hat{\epsilon}/h$ and require a bound of $\hat{\epsilon}/2$ on the duality gap. We next proceed to derive an optimization algorithm for the soft dual objective f_ρ.

5 Algorithm

In this section we propose to use the Frank-Wolfe algorithm for optimizing our alternative dual problem F_ρ (Eq. (5)). As this is a constrained convex program, there are other methods that can be applied to reach the same global optimum, such as exponentiated gradient. We opt for Frank-Wolfe due to its simplicity and parameter-free implementation. In particular, it does not require tuning of meta parameters (e.g., step-size).

The original Frank-Wolfe algorithm (Frank and Wolfe, 1956) is a conditional gradient method, where in each step a linear upper-bound of the objective function is computed, and then an optimal solution of the linear function under the constraints is found. The variables are then updated by taking a step towards this optimal solution. Recently, a block-coordinate Frank-Wolfe algorithm (BCFW) has been used for structured SVMs, where in each iteration only a few of the variables are updated (Lacoste-Julien et al., 2013) (in this setting it is equivalent to *stochastic dual coordinate ascent* (Shalev-Shwartz and Zhang, 2013)). Moreover, for structured SVMs the optimal step size can be computed analytically, resulting in a parameter-free algorithm.

The BCFW algorithm is applicable to product domains, where variables in each block are constrained independently of the other blocks. In structured SVMs the constraints couple together all dual variables in the same training example. As a result, in order to compute an update, the algorithm requires calling a maximization oracle per example. Even for approximate oracles, this may get quite costly when training instances consist of many interdependent variables. In

[5] For a similar assumption see Theorem 1 in Weiss and Taskar (2010). In fact, we can drop the assumption $\|w\|_2 \leq B$ and use a bound on $\|w^*\|_2$ similar to the one in Shalev-Shwartz et al. (2011).

Algorithm 1 Block-coordinate Frank-Wolfe for soft structured SVM

```
1: Initialize: $w = 0$, $\delta = 0$, $\mu_\alpha^{(m)}(y_\alpha) = \mathbb{1}\{y_\alpha = y_\alpha^{(m)}\}$ for all $m, \alpha, y_\alpha$
2: while not converged do
3:   Randomly sample a block $(m, \alpha)$
4:   if Variable chosen $(\alpha \in \{i\})$ then
5:     Let $\hat{\theta}_i(y_i) = \theta_i^{(m)}(y_i; w) + \sum_{c:i\in c} \delta_{ci}^{(m)}(y_i)$
6:     Let $y_i^* = \operatorname{argmax}_{y_i} \hat{\theta}_i(y_i)$, and let $s_i$ be the corresponding indicator vector
7:     Let $\gamma = \frac{\hat{\theta}_i^\top \left(s_i - \mu_i^{(m)}\right)}{\lambda \|\Psi_{m,i}(s_i - \mu_i^{(m)})\|^2 + \rho N_i \|s_i - \mu_i^{(m)}\|^2}$ and clip to $[0, 1]$
8:   end if
9:   if Factor chosen $(\alpha \in \{c\})$ then
10:    Let $\hat{\theta}_c(y_c) = \theta_c^{(m)}(y_c; w) - \sum_{i:i\in c} \delta_{ci}^{(m)}(y_i)$
11:    Let $y_c^* = \operatorname{argmax}_{y_c} \hat{\theta}_c(y_c)$, and let $s_c$ be the corresponding indicator vector
12:    Let $\gamma = \frac{\hat{\theta}_c^\top \left(s_c - \mu_c^{(m)}\right)}{\lambda \|\Psi_{m,c}(s_c - \mu_c^{(m)})\|^2 - \rho \sum_{i:i\in c} \|A_{ci}(s_c - \mu_c^{(m)})\|^2}$ and clip to $[0, 1]$
13:  end if
14:  Update $\mu_\alpha^{(m)} \leftarrow (1-\gamma)\mu_\alpha^{(m)} + \gamma s_\alpha$
15:  Update $w = \Psi\mu$ and $\delta = A\mu$
16: end while
```

contrast, in our objective f_ρ each factor is constrained independently, so the oracle calls are much cheaper.

Applying BCFW to our dual objective f_ρ yields Algorithm 1. Here, $N_i = |\{c : i \in c\}|$ is the number of factors containing variable i, $\Psi_{m,\alpha}$ is the part in Ψ corresponding to sample m and factor α, and A_{ci} marginalizes μ_c to values of variable $i \in c$. To derive this algorithm, we need to first compute a linear bound on the objective, which reduces to computing the gradient of each block. It turns out that in our case the block gradients are $\nabla_{\mu_\alpha^{(m)}(y_\alpha)} f_\rho = \hat{\theta}_\alpha^{(m)}(y_\alpha)$ (lines 5 and 10). Interestingly, this has the exact same form as the factor scores in the primal g (Eq. (2)), and is known as a *reparameterization* of the model scores (Wainwright and Jordan, 2008; Sontag et al., 2011). The next step in the algorithm is to optimize the linear function over the constraints, which means computing $s_\alpha = \operatorname{argmax}_{s'_\alpha \in S_\alpha} \left\langle s'_\alpha, \nabla_{\mu_\alpha^{(m)}} f_\rho \right\rangle$. Since S_α consists of local simplex constraints, the optimal s_α is just an indicator vector for the maximal element (lines 6 and 11). Next, the optimal step size is obtained by solving the univariate quadratic problem $\min_{\gamma\in[0,1]} f_\rho(\mu + \gamma(s_\alpha - \mu_\alpha))$, which results in the values in lines 7 and 12. Finally, a step is taken towards the optimal point s_α (line 14).

Algorithm 1 has several compelling properties. First, the update of primal variables (w, δ) in line 15 is computationally cheap since a change in $\mu_\alpha^{(m)}$ affects only w_α and δ variables pertaining to the chosen α (i.e., its neighbors in the factor graph). For more details see Appendix D. Second, the algorithm employs simple per-factor maximization oracles, unlike more expensive oracles employed by other algorithms (e.g., max-marginals oracle in Meshi et al. (2010)). Third, the algorithm can naturally handle global factors (i.e., $y_c = y$) by storing only primal variables (w, δ) and some auxiliary compact variables, and executing all updates in primal space (see Appendix D). In this case, all we need to do is solve the factor maximization in line 11, which is possible for several global factors (cf. Koo et al., 2010). On the other hand, if dual variables μ can be maintained, then this algorithm can be naturally used with kernels. Fourth, since the *optimal* step-size is computed analytically, there are no hyperparameters to tune. Finally, as a stopping criterion we use the duality gap, which requires computing the primal objective g_ρ every several passes over the data.

5.1 Convergence Rate

We next analyze the convergence rate of Algorithm 1. We build on the following theorem.

Theorem 5.1 (Lacoste-Julien et al. (2013), Theorem 2). *For each $t > 0$ it holds that $f_\rho^* - \mathbb{E}\left[f_\rho^{(t)}\right] \le \frac{2Mq}{t+2Mq}\left(C_{f_\rho}^{\otimes} + (f_\rho^* - f_\rho^{(0)})\right)$, where $C_{f_\rho}^{\otimes}$ is the curvature constant of f_ρ, and the expectation is over the random choice of blocks. Furthermore, the duality gap is bounded by:*[6] $\mathbb{E}\left[D_\rho^{(\hat{t})}\right] \le \frac{6Mq}{t+1}\left(C_{f_\rho}^{\otimes} + (f_\rho^* - f_\rho^{(0)})\right)$*, for some $0 \le \hat{t} \le t$.*

Given this theorem, it remains to compute the curvature constant $C_{f_\rho}^{\otimes}$. In Appendix E we show that $C_{f_\rho}^{\otimes} = O\left(\frac{q}{M}\left(\frac{1}{\lambda} + \frac{1}{\rho}\right)\right)$, which leads to the following result.

Corollary 5.2. *Algorithm 1 obtains (in expectation) an ϵ-optimal solution to problem G_ρ with duality gap $\mathbb{E}[D_\rho] \le \epsilon$ after $O\left(\frac{q^2}{\epsilon}\left(\frac{1}{\lambda} + \frac{1}{\rho}\right)\right)$ iterations.*[7]

In comparison, for the constrained objective f (Eq. (3)) the rate obtained in Lacoste-Julien et al. (2013) is $O\left(\frac{1}{\lambda\epsilon}\right)$, which seems faster. However, recall that each iteration requires calling a maximization

[6] Using a simple argument it is easy to show that the Lagrange duality gap D_ρ is equal to the Fenchel duality gap in our case, which justifies this statement of the theorem.

[7] This actually requires assuming a bound on the initial suboptimality $f_\rho^* - f_\rho^{(0)}$, which can be partly relaxed with a finer analysis of the line-search procedure (see Lacoste-Julien et al., 2013).

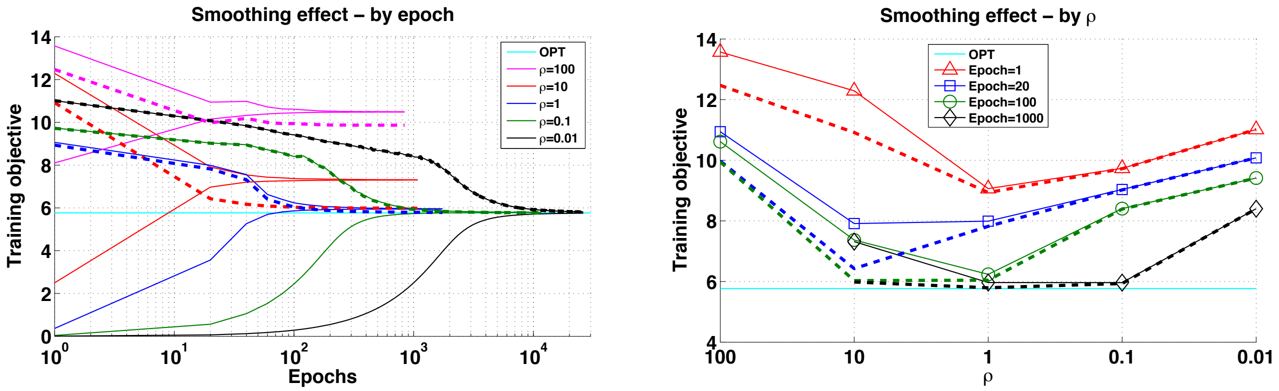

Figure 1: (Left) Training objective as a function of training epochs for different values of ρ. The primal g_ρ (upper) and dual f_ρ (lower) soft objectives are shown with solid lines, and the constrained primal $g(w_\rho, \delta_\rho)$ with a dashed line. The horizontal line marks the optimal value g^*. (Right) Training objective as a function of the constant ρ for fixed numbers of epochs. The soft primal g_ρ is denoted with a solid line, and the constrained primal $g(w_\rho, \delta_\rho)$ with a dashed line. The horizontal line marks the optimal value g^*.

oracle for a complete training example ($\mu^{(m)}$ block), while Algorithm 1 requires optimizing only over factor blocks $\mu_\alpha^{(m)}$, which can be much cheaper. Combining this rate with Theorem 4.2 yields the following result.

Corollary 5.3. *Running Algorithm 1 for $O\left(\frac{q^2}{\epsilon}\left(\frac{1}{\lambda}+\frac{1}{\rho}\right)\right)$ iterations yields a primal solution w that is $\left(\epsilon + \frac{\rho}{2}h\right)$-optimal for problem G. Put differently, in order to get an $\hat{\epsilon}$-optimal solution for problem G, it is enough to set $\rho = \hat{\epsilon}/h = O(\hat{\epsilon}/Mq^2)$ and run Algorithm 1 for $O\left(\frac{q^2}{\lambda\hat{\epsilon}} + \frac{Mq^4}{\hat{\epsilon}^2}\right)$ iterations.*

We observe that due to the linear dependence on M in Theorem 4.1, we need to set ρ small in order to get an accurate solution to problem G. In turn, this results in a distressing term of $Mq^4/\hat{\epsilon}^2$ in our bound. However, we will see in Section 6 that in actual applications we do not have to set ρ very small to achieve good accuracy.

6 Experiments

We next evaluate our algorithm empirically and compare its performance against state-of-the-art baselines. For all algorithms we use our own `C++` implementation. We tune the regularization constant λ via cross-validation, and report results for the optimal value, when training with the entire trainset. We conduct experiments on two different domains: multi-label classification and image segmentation.

Multi-label classification In multi-label classification the task is to assign a subset of possible labels that best fits a given input. In this problem each label is represented by a binary variable $y_i \in \{0,1\}$, and the model consists of singleton and pairwise scores: $w^\top \phi(x,y) = \sum_i (w_i^\top x) y_i + \sum_{i,j} w_{ij} y_i y_j$ (for all $\binom{n}{2}$ possible pairs). We first focus on the *Yeast* dataset, where there are $n = 14$ labels, $|x| = 103$ features (hence, $d = 1533$), $M = 1500$ training samples, and 917 test samples.[8]

We begin by studying the effect of the penalty constant ρ. Figure 1 (left) shows the training objective as a function of the number of effective passes through the data (epochs) for different values of ρ. First, we observe, as expected, that as ρ becomes smaller the optimum of the soft objective g_ρ^* gets closer to the optimal constrained objective g^*, and the algorithm takes longer to converge. Second, the soft objective $g_\rho(w_\rho, \delta_\rho)$ approaches the constrained objective $g(w_\rho, \delta_\rho)$, and in fact, for $\rho \leq 0.1$ the two become practically indistinguishable. This suggests that the bounds in Section 4.1 may sometimes be loose.

Figure 1 (right) gives a different perspective on the effect of ρ. Here we fix the number of training epochs (i.e., allowed computation) and plot the training objective for different values of ρ. We see that setting ρ too large yields loose objective values, while setting ρ too small results in slow convergence, which also makes the objective value suboptimal. Setting $\rho = 1$ seems to work well across different computational budgets (a similar behavior was observed for the other datasets).

We next compare our algorithm to other state-of-the-art trainers for structured SVM. In particular, we implement the block-coordinate FW algorithm (Lacoste-Julien et al., 2013) for the constrained problem F, where each block consists of a single training example $\mu^{(m)}$, and we use the GLPK solver to compute the

[8] The multi-label datasets are available at `http://mulan.sourceforge.net`.

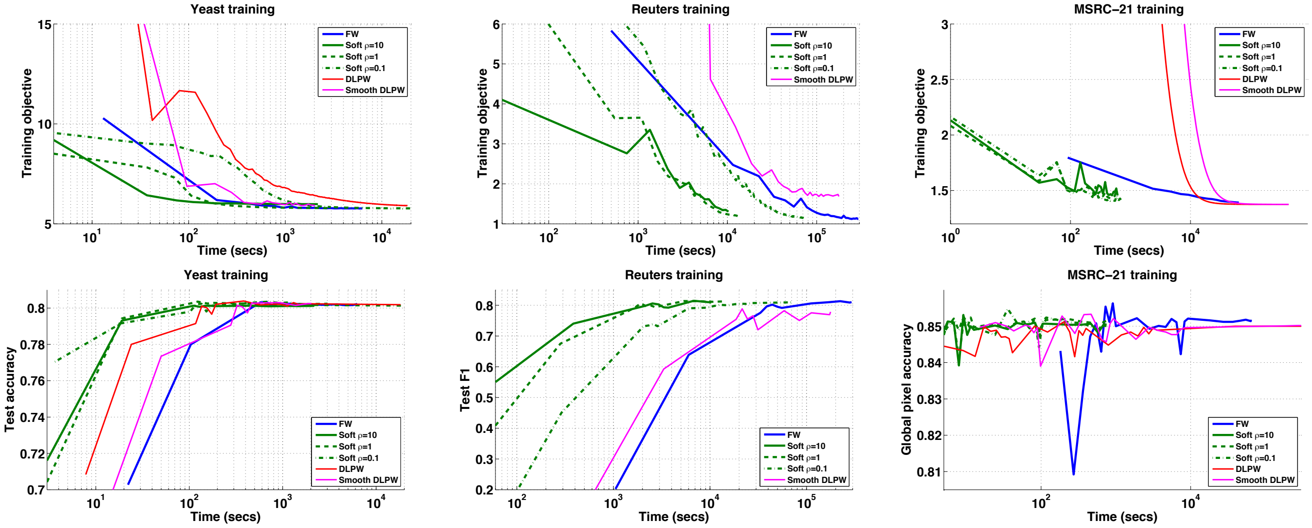

Figure 2: Comparison of training objective (top row) and performance measure (bottom row) as a function of runtime for the Yeast (left column), Reuters (middle column) and MSRC-21 (right column) datasets. In the top row we show only the hard-constrained objective value $g(w(\mu), \delta(\mu))$.

updates. In the same spirit of our approach, we also tried to decrease the tolerance of constraint satisfaction or solution accuracy within the LP solver, however this did not result in any noticeable improvement in runtime. We also run the Dual-Loss-Primal-Weights (DLPW) algorithm (Meshi et al., 2010), which optimizes g by combining block-minimization updates on δ with SGD updates on w. Notice that this method is similar to ours in the sense that the updates are "local" and do not process complete training samples. In addition, we compare to a smooth version of DLPW in which the max terms in g are replaced with soft-max terms (Hazan and Urtasun, 2010). Here we try several smoothing constants and show the best performing one. For both DLPW variants we try several step sizes for SGD (i.e., η/t) and show the one with the best convergence behavior. We compare computational cost of the algorithms in terms of runtime rather than oracle calls since different algorithms require oracles with very different costs.

Figure 2 (left, top) shows that our method is competitive in terms of convergence speed. Notice that in this problem the training instances consist of only 14 variables, so the LP solver employed by FW is pretty effective and overall convergence is fast. Figure 2 (left, bottom) compares test accuracy of the predictors. For all algorithms we use an LP solver to compute test prediction, and we use a simple rounding scheme in case of fractional solutions. Here we observe that the performance of our method is rather insensitive to the choice of ρ, and even a large value $\rho = 10$ is sufficient to obtain good prediction accuracy. We notice that our algorithm is able to quickly learn an accurate model.

To compare the algorithms in a more challenging setting, we run experiments on the *Reuters* multi-label dataset (RCV1). We adopt the experimental setting of Samdani and Roth (2012), where there are 6000 instances, with $|x| = 47,236$ input features, and the output is reduced to the $n = 30$ most frequent labels (here, $d \approx 1.4$M). We randomly split the data into $M = 5000$ training and 1000 test samples. Performance is measured by an $F1$ score rather than plain accuracy, since the number of active labels per instance is relatively small (see Samdani and Roth, 2012).

In Figure 2 (middle) we observe that our method is up to *two orders of magnitude* faster than the baselines.[9] Since outputs are modeled by a fully connected graph over 30 labels, the LP solver consumes more time, which in turn slows down the FW algorithm. Here we see that choosing a strong penalty ($\rho = 0.1$) might increase the runtime (although it is still significantly faster than the baselines), however, mild values ($\rho \in \{1, 10\}$) yield fast training and accurate predictors.

Image segmentation We next conduct experiments on the task of semantic segmentation from computer vision. Here the goal is to assign a class to each pixel of an input image. We focus on the MSRC-21 dataset, and use the model and features from Yao et al. (2012).[10] In this problem there are 21 possible classes, and the model has two types of output variables corresponding to fine-level and

[9] We omit DLPW here since it performed poorly.

[10] Available from `http://ttic.uchicago.edu/~yaojian/HolisticSceneUnderstanding.html`.

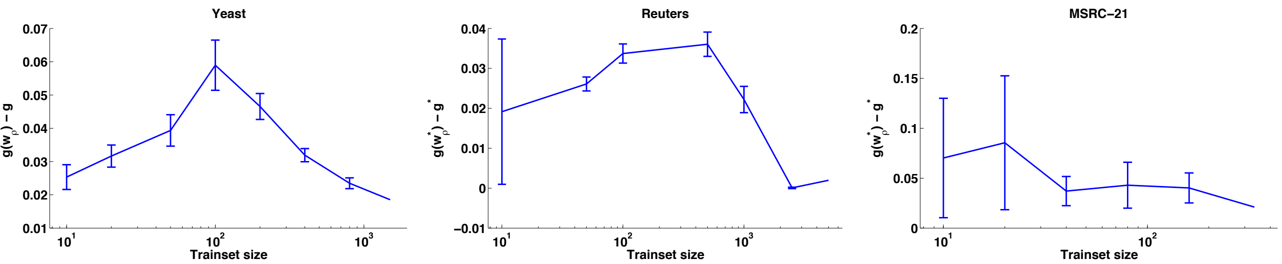

Figure 3: Difference between primal objective values for soft- and hard-constrained solutions $(g(w(\mu^*_\rho), \delta(\mu^*_\rho)) - g^*)$, as a function of trainset size. The mean and 95% confidence interval over 10 random subsamples are shown.

coarse-level superpixels in the image. Each fine-level superpixel i is involved in a pairwise score with a single coarse-level superpixel $j(i)$, encouraging agreement. The weights are shared for all singleton and all pairwise scores, and the overall score is: $w^\top \phi(x, y) = \sum_{i \in \text{Fine}} \left(w^s_{y_i} x_{i,y_i} - w^p \mathbb{1}\{y_i \neq y_{j(i)}\} \right) + \sum_{j \in \text{Coarse}} w^s_{y_j} x_{j,y_j}$. In this dataset each image has on average 65 fine-level and 31 coarse-level superpixels. There are a total of 591 images divided into $M = 335$ train and 256 test samples.

In Figure 2 (right, top) we compare the training time of our algorithm to that of the baselines. We observe once more that our algorithm is *two orders of magnitude* faster than the others. As before, the per-sample LPs are challenging, so each iteration of FW becomes quite expensive. Furthermore, we find out that DLPW is sensitive to the choice of step-size, and even the best choice is rather slow. In contrast, our algorithm has cheap local updates and uses the optimal step at each iteration, thereby achieving fast convergence. In Figure 2 (right, bottom) we compare prediction quality by computing the average per-pixel accuracy. Due to parameter sharing, the total number of parameters is small ($d = 22$), and all models reach high accuracy levels after less than one epoch. However, since our method has very low cost per iteration, it is able to settle on an accurate model faster than the others. Here we see that our method is insensitive to the choice of ρ, and all values perform similarly.

Finally, we test the effect of the train size M on the soft constrained objective function. To this end, we randomly sample a subset of the training data of increasing size (M), we set $\rho = 1$ and compute the difference $g(w(\mu^*_\rho), \delta(\mu^*_\rho)) - g^*$, where μ^*_ρ is obtained by optimizing f_ρ. This is averaged over 10 random subsamples. In Figure 3 we see that for all datasets the difference in objective values grows much slower than linearly (notice the log-scale). In fact, as the trainset becomes larger the difference in objectives stops growing, and even shrinks. This again shows that the bound in Theorem 4.2 may be loose in practice.

7 Conclusion and Future Work

In this paper we present a novel training algorithm for structured prediction. The key idea of our work is that strictly enforcing local marginalization constraints is not so crucial in the context of structured learning. Based on this insight, we apply the penalty method to the structured SVM objective function. We provide theoretical guarantees for both the resulting objective function and optimization algorithm. We also demonstrate empirically on diverse datasets, that using soft constraints can significantly reduce training effort while maintaining the same predictive quality.

Our method can be extended in several ways. First, it is reasonable to start the penalty strength ρ from a fairly loose (high) value and gradually decrease it along the iterations in order to enforce the marginalization constraints more strictly. Second, one can use an augmented Lagrangian formulation which includes Lagrange multipliers for the constraints in addition to the quadratic penalty. The multipliers are updated in an outer loop, where in each iteration we need to solve the problem in Eq. (5) with an additional linear term (requires a minor modification to Algorithm 1). The optimal solution is then guaranteed to satisfy the marginalization constraints, for any value ρ. Third, to improve the convergence rate, it is possible to smooth the max terms in the primal objective Eq. (2), for example by applying the smooth-max-of-hinge technique in Shalev-Shwartz and Zhang (2014). This adds a simple regularizer to the dual problem Eq. (5) (which again requires a small change in Algorithm 1).

Finally, our work is related to previous work on statistical estimation and optimization errors in machine learning. In particular, it has been shown that in some scenarios, it is not necessary to carry out the optimization with great accuracy (Bottou and Bousquet, 2007; Shalev-Shwartz and Srebro, 2008). In the same spirit, our method trades-off optimization error with computational cost by using an approximate more convenient objective function. Studying this trade-off in our case is an interesting future direction.

References

D. Belanger, A. Passos, S. Riedel, and A. McCallum. Message passing for soft constraint dual decomposition. In *UAI*, 2014.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems*, pages 161–168. 2007.

D. Boukari and A. V. Fiacco. Survey of penalty, exact-penalty and multiplier methods from 1968 to 1993. *Optimization*, 32(4):301–334, 1995.

M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, 2002.

M. Collins, A. Globerson, T. Koo, X. Carreras, and P. L. Bartlett. Exponentiated gradient algorithms for conditional random fields and max-margin markov networks. *JMLR*, 9:1775–1822, 2008.

P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE PAMI*, 32(9):1627–1645, 2010.

T. Finley and T. Joachims. Training structural SVMs when exact inference is intractable. In *Proceedings of the 25th International Conference on Machine learning*, pages 304–311, 2008.

M. Frank and P. Wolfe. *An algorithm for quadratic programming*, volume 3, pages 95–110. 1956.

T. Hazan and R. Urtasun. A primal-dual message-passing algorithm for approximated large scale structured prediction. In *NIPS*, pages 838–846. 2010.

N. Komodakis. Efficient training for pairwise or higher order crfs via dual decomposition. In *CVPR*, CVPR '11, pages 1841–1848, 2011.

T. Koo, A. M. Rush, M. Collins, T. Jaakkola, and D. Sontag. Dual decomposition for parsing with non-projective head automata. In *EMNLP*, 2010.

A. Kulesza and F. Pereira. Structured learning with approximate inference. In *Advances in Neural Information Processing Systems 20*, pages 785–792. 2007.

S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *ICML*, pages 53–61, 2013.

O. Meshi, D. Sontag, T. Jaakkola, and A. Globerson. Learning efficiently with approximate inference via dual losses. In *ICML*, pages 783–790. ACM, 2010.

N. Ratliff, J. A. D. Bagnell, and M. Zinkevich. (Online) subgradient methods for structured prediction. In *AISTATS*, 2007.

R. Samdani and D. Roth. Efficient decomposed learning for structured prediction. In *ICML*, 2012.

S. Shalev-Shwartz and N. Srebro. SVM optimization: inverse dependence on training set size. In *Proceedings of the 25th International Conference on Machine learning*, 2008.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *JMLR*, 14:567–599, 2013.

S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In *ICML*, 2014.

S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

O. Shamir and T. Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *ICML*, 2013.

D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, pages 503–510, 2008.

D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In *Optimization for Machine Learning*, pages 219–254. MIT Press, 2011.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*. MIT Press, 2003.

B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: a large margin approach. In *ICML*, 2005.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, pages 104–112, 2004.

M. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc., Hanover, MA, USA, 2008.

D. Weiss and B. Taskar. Structured Prediction Cascades. In *AISTATS*, 2010.

T. Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (MAP-MRF). In *CVPR*, 2008.

J. Yao, S. Fidler, and R. Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *CVPR*, 2012.