2007

(Online) Subgradient Methods for Structured Prediction

Nathan D. Ratliff
Carnegie Mellon University

J. Andrew Bagnell
Carnegie Mellon University

Martin A. Zinkevich
University of Alberta

(Online) Subgradient Methods for Structured Prediction

Nathan D. Ratliff
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

J. Andrew Bagnell
Robotics Institute / MLD
Carnegie Mellon University
Pittsburgh, PA 15213

Martin A. Zinkevich
Department of Computing Science,
University of Alberta,
Edmonton, AB T6G 2E1, Canada

Abstract

Promising approaches to structured learning problems have recently been developed in the maximum margin framework. Unfortunately, algorithms that are computationally and memory efficient enough to solve large scale problems have lagged behind. We propose using simple subgradient-based techniques for optimizing a regularized risk formulation of these problems in both online and batch settings, and analyze the theoretical convergence, generalization, and robustness properties of the resulting techniques. These algorithms are are simple, memory efficient, fast to converge, and have small regret in the online setting. We also investigate a novel convex regression formulation of structured learning. Finally, we demonstrate the benefits of the subgradient approach on three structured prediction problems.

1 Introduction

Maximum margin structured learning (MMSL) has recently gained prominence within the machine learning community as an efficient approach to predicting multiple, interconstrained labels. However, most current methods are limited in terms of scalability, convergence, or memory requirements. The resulting convex optimization problems are often prohibitively large for generic quadratic programming solvers, and the Structured SMO method proposed in (Taskar et al., 2003) to alleviate this difficulty is slow to converge in both theory and practice. Dual exponentiated gradient techniques suffer from sublinear convergence as well as the often large memory requirements of dual formulations. Recently, (Taskar et al., 2006) have investigated saddle-point methods for optimization and have succeeded in efficiently solving several problems that would have otherwise had intractable memory requirements. These methods are promising, but are limited to the (important) special case where the structured prediction can be cast as a linear program.

Extending work initially proposed in (Ratliff et al., 2006b) for solving problems in imitation learning, we develop an alternative gradient based approach to structured learning using a regularized risk formulation of MMSL derived by placing the constraints into the objective to create a convex function in w. This objective is then optimized by a direct generalization of gradient descent, popular in convex optimization, called the subgradient method (Shor, 1985). The abundance of literature on subgradient methods makes this algorithm a decidedly convenient choice. In this case, it is well known that the subgradient method is guaranteed linear convergence when the stepsize is chosen to be constant. Furthermore, this algorithm is the Greedy Projection algorithm of (Zinkevich, 2003) in the online setting. Using tools developed in (Hazan et al., 2006), we can show that the risk of this online algorithm with respect to the prediction loss grows only sublinearly in time. Perhaps more importantly, the implementation of this algorithm is simple and has intuitive appeal since an integral part of the computation comes from running the inference algorithm being trained in the inner loop.

Our work connects two previously distinct threads of research in structured prediction. We show that the gradient descent approach to learning graph transformer backpropagation networks pioneered in (LeCun et al., 1998) may be straightforwardly extended to solve the novel, margin-scaling structured classification approach developed by (Taskar et al., 2006).[1] This yields perhaps the simplest, most computationally efficient algorithms for solving structured maximum margin problems. The application of subgradient

[1]Recent other work has attempted to make similar connections including suggesting related loss functions (LeCun et al., 2007) that are *not* equivalent to the structured maximum margin criteria. Section 7 suggests these methods have poorer performance both empirically and theoretically.

descent to the structured margin loss functions brings benefits concomitant with convexity: efficient global optimization, small online regret, and new bounds on generalization error for these algorithms. We introduce a new formulation of structured regression that generalizes support vector regression, and is to our knowledge the only convex formulation in the literature. Further, we study the robustness of these algorithms to approximate settings, namely, when inference is only approximate or subgradients cannot be computed exactly. Finally, we consider application of our techniques to two previously studied classification problems as well as a novel regression problem.

2 Maximum margin structured classification

We present a brief review of maximum margin structured classification in terms of convex programming. In this setting, we attempt to predict a structured object $y \in \mathcal{Y}(x)$ (e.g. a parse tree, label sequence, trajectory for a robot) from a given input $x \in \mathcal{X}$. For our purposes we assume that the inference problem can be described in terms of a computationally tractable max over a score function $s_x : \mathcal{Y}(x) \to \mathbb{R}$ such that $y^* = \arg\max_{y \in \mathcal{Y}(x)} s_x(y)$ and take as our hypothesis class functions of the form

$$h(x; w) = \arg\max_{y \in \mathcal{Y}(x)} w^T f(x, y) \tag{1}$$

This class is parameterized by w in a convex parameter space $\mathcal{W}$, and $f(x, y)$ are vector valued feature functions. Given data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ we abbreviate $f(x_i, y)$ as $f_i(y)$ and $\mathcal{Y}(x_i)$ as $\mathcal{Y}_i$.

The margin is chosen to scale with the loss of choosing class y over the desired y_i. We denote this prediction loss function by $\mathcal{L}(y_i, y) = \mathcal{L}_i(y)$, and assume that $\mathcal{L}_i(y) > 0$ for all $y \in \mathcal{Y}_i \backslash y_i$, and $\mathcal{L}_i(y_i) = 0$. In learning, our goal is to find a score function that scores y_i higher than all other $y \in \mathcal{Y}_i \backslash y_i$ by this margin. Formally, this gives us the following constraint:[2]

$$\forall i,\ y \in \mathcal{Y}_i, \quad w^T f_i(y_i) \geq w^T f_i(y) + \mathcal{L}_i(y). \tag{2}$$

Maximizing the left hand side over all $y \in \mathcal{Y}_i$, and adding slack variables, we can express this mathematically as following convex program:

$$\min_{w, \zeta_i} \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_i \zeta_i \tag{3}$$
$$s.t.\ \ \forall i \quad w^T f_i(y_i) + \zeta_i \geq \max_{y \in \mathcal{Y}_i} \left(w^T f_i(y) + \mathcal{L}_i(y)\right)$$

[2](Tsochantaridis et al., 2005) describes an alternative formulation for MMSC based around scaling the slack variables by the loss rather than scaling the margin. Subgradient methods are applicable to this formulation as well, though we don't formally discuss this case.

where $\lambda \geq 0$ is a hyperparameter that trades off constraint violations for margin maximization (i.e. fit for simplicity).

Finally, we note that the constraints is this convex program are tight (equality holds at the optimum) so we can place them directly into the objective. Doing so we arrive at the following regularized risk function:[3]

$$c(w) = \frac{1}{n}\sum_{i=1}^n r_i(w) + \frac{\lambda}{2}\|w\|^2$$
$$\text{where} \ \ r_i(w) = \max_{y \in \mathcal{Y}_i}(w^T f_i(y) + \mathcal{L}_i(y)) - w^T f_i(y_i)$$

3 Maximum margin structured regression

Rather than focusing on predicting an object correctly, maximum margin structured regression (MMSR) is designed to learn a structured predictor whose score matches supervised values. For example, this form of structured prediction was motivated by the problem of learning informed heuristic functions for high-dimensional planners. Optimal high-dimensional planners often produce very desirable plans, but are too slow to use in practice at face value. By projecting those plans onto a low-dimensional planning space and training a regressor to match the value of those plans, an extremely accurate heuristic function can be learned carefully direct the high-dimensional planner toward the goal. Section 7 describes this problem in more detail.

In this setting, we work within the same structured setting with a set of classes $\mathcal{Y}(x)$ for each input $x \in \mathcal{X}$. But this time we are given a data set $\mathcal{D} = \{(x_i, y_i, v_i)\}_{i=1}^n$ where $v_i \in \mathbb{R}$ and we wish to learn a function $s : \mathcal{X} \to \mathbb{R}$ given by

$$s(x; w) = \max_{y \in \mathcal{Y}(x)} w^T f(x, y). \tag{4}$$

Following the common SVM regression approach, we write down the ϵ-insensitive criteria $\forall i,\ v_i - \epsilon_- \leq \max_{y \in \mathcal{Y}_i} w^T f_i(y) \ \leq \ v_i + \epsilon_+$ as the following constraints:

$$\forall i, \quad v_i - \epsilon_- \leq \max_{y \in \mathcal{Y}_i} w^T f_i(y) \tag{5}$$
$$\forall i, \quad v_i + \epsilon_+ \geq \max_{y \in \mathcal{Y}_i} w^T f_i(y). \tag{6}$$

The non-convexity of constraint 5 can be circumvented by utilizing the structured label information provided

[3]More generally, we can scale the risk by a data dependent constant and raise it to a power $q \geq 1$ as is done in (Ratliff et al., 2006b). The resulting objective is still convex and a chain rule for subgradients allows for the calculation of its subgradient. The primary components of this theory are captured most simply with $q = 1$, however, so we have opted to leave it out.

in the data set. Replacing the max score with the score of the supervised label in that constraint (i.e. replacing $\max_{y \in \mathcal{Y}_i} w^T f_i(y)$ with $w^T f_i(y_i)$) results in a convex constraint. Moreover, satisfying this new constraint implicitly enforces the original constraint since the max score upper bounds the score of the supervised label. Adding slacks $\{\alpha_i, \beta_i\}_{i=1}^n$ as we did for MMSC, and constraining $\alpha_i \geq 0$ and $\beta_i \geq 0$, we arrive at the following convex program:

$$\min_{w,\alpha_i,\beta_i} \quad \frac{1}{n}\sum_{i=1}^{n}(\alpha_i + \beta_i) + \frac{\lambda}{2}\|w\|^2 \tag{7}$$

$$\text{s.t.} \quad \forall i, \quad v_i \geq \max_{y \in \mathcal{Y}_i} w^T f_i(y) - \alpha_i - \epsilon_+ \tag{8}$$

$$\forall i, \quad v_i \leq w^T f_i(y_i) + \beta_i + \epsilon_- \tag{9}$$

$$\forall i, \quad \alpha_i \geq 0, \quad \beta_i \geq 0 \tag{10}$$

If we rearrange inequalities (8) and (9) so that α_i and β_i are on the left, and take the max of the resulting right hand sides with 0, the inequalities 10 become implicit and the final inequality constraints are tight. This reduces the constraints to

$$\forall i, \quad \alpha_i \geq \max\{0, \ \max_{y \in \mathcal{Y}_i} w^T f_i(y) - v_i - \epsilon_+\} \tag{11}$$

$$\forall i, \quad \beta_i \geq \max\{0, \ v_i - w^T f_i(y_i) - \epsilon_-\} \tag{12}$$

Placing these constraints into the objective gives us the following convex regularized risk function:

$$R(w) = \frac{1}{n}\sum_{i=1}^{n}(\psi_i(w) + \phi_i(w)) + \frac{\lambda}{2}\|w\|^2 \tag{13}$$

where $\psi_i(w) = \max\{0, \ \max_{y \in \mathcal{Y}_i} w^T f_i(y) - v_i - \epsilon_+\}$ and $\phi_i(w) = \max\{0, \ v_i - w^T f_i(y_i) - \epsilon_-\}$.

4 Optimization via the subgradient method

In Section 2 and 3 we derived convex regularized risk functions for solving structured prediction problems. Learning algorithms can be derived using to standard tools from convex optimization. Probably the most widely used optimization algorithm from this area is the subgradient method. As a direct generalization of gradient descent to nondifferentiable convex objective functions, this method iteratively computes and steps in the negative direction of a gradient-like vector known as a subgradient. Additionally, a simple extension of this iterative algorithm gives way to straightforward online variant within the framework of online convex programming (Zinkevich, 2003). In this section, we define what we mean by a subgradient, present some tools for computing them in the case of MMSC and MMSR, and present the resulting algorithm.

Algorithm 1 MMSC subgradient calculation

```
1: procedure SubgradMMSC( $(x_i, y_i, v_i)$, $\mathcal{L}_i(y)$, $f_i : \mathcal{X} \to \mathbb{R}^d$, $w \in \mathcal{W}$ )
2:     $y^* = \arg\max_{y \in \mathcal{Y}} w^T f_i(y) + \mathcal{L}_i(y)$
3:     $g \leftarrow g + f_i(y^*) - f_i(y_i)$
4:     return $g$
5: end procedure
```

Algorithm 2 MMSR subgradient calculation

```
1: procedure SubgradMMSR( $(x_i, y_i, v_i)$, $f_i : \mathcal{X} \to \mathbb{R}^d$, $w \in \mathcal{W}$ )
2:     $y^* = \arg\max_{y \in \mathcal{Y}} w^T f_i(y)$
3:     if $w^T f_i(y^*) > v_i$ then
4:         $g \leftarrow g - f(y^*)$
5:     end if
6:     if $w^T f_i(y_i) < v_i$ then
7:         $g \leftarrow g + f_i(y_i)$
8:     end if
9:     return $g$
10: end procedure
```

We define a subgradient using a tangent-like lower-bound property of the function at the point in question. This can be written formally as follows:

Definition 4.1: Subgradient. *Let $h : \mathcal{W} \to \mathbb{R}$ be a convex function over $\mathcal{W} \subset \mathbb{R}^d$. A subgradient at $w \in \mathcal{W}$ is a vector $g \in \mathbb{R}^d$ for which $\forall w' \in \mathcal{W}$, $h(w') \geq h(w) + g^T(w' - w)$.*

The set of subgradients at w is called a subdifferential and is denoted by $\partial h(w)$. A subdifferentiable function is a function for which we can efficiently compute a subgradient at every point in $\mathcal{W}$. The gradient is the unique subgradient at any point of differentiability, while there exists a continuum of subgradient vectors at points of nondifferentiability.

Algorithms 1 and 2 demonstrate how to calculate the exact subgradients for a single term of the MMSC and MMSR regularized risk functions, respectively. Following the negative of these subgradients has intuitive appeal; the algorithm decreases the score if it is too high and increases the score if it is too low. The theoretical analysis and experimental results that follow show that even these simple, intuitively appealing, algorithms perform well for structured learning. In some cases, their performance is currently unsurpassed.

While analogous to gradient descent for differentiable functions, there is one key difference. Applying line search algorithms to nondifferentiable objective function optimization may cause Algorithm 3 to converge to a suboptimal fixed point. Consequently, the stepsize sequence is assumed to be chosen in advance. Com-

Algorithm 3 Subgradient algorithm

```
1: procedure SUBGRAD( $h = \sum_{i=1}^m h_i$, $w_0 \in \mathcal{W}$ )
2:     $w_\alpha \leftarrow 0$,   $\alpha_s \leftarrow 0$
3:     for $t = 1, \ldots, T-1$ do
4:         Compute   $g_t \leftarrow \nabla h(w_t)$
5:         Update   $w_{t+1} \leftarrow \mathcal{P}_{\mathcal{W}}[w_t - \alpha_t g_t]$
6:     end for
7:     return $\arg\min_t h(w_t)$
8: end procedure
```

mon sequences include $\{\frac{c}{t}\}_{t=1}^\infty$ and $\{\frac{c}{\sqrt{t}}\}_{t=1}^\infty$ for $c > 0$. In Algorithm 3, $\{\alpha_t\}_{t=1}^\infty$ denotes a predefined stepsize sequence. A given step might land outside the feasible set $\mathcal{W}$, so after each step, the resulting point is projected back onto $\mathcal{W}$ using a projection operator $\mathcal{P}_{\mathcal{W}} : \mathbb{R}^d \rightarrow \mathcal{W}$. [4]

5 Theoretical results

Framing these structured learning problems as convex regularized risk functions and optimizing them via variants of the subgradient method allows for straightforward analysis of the optimization and learning convergence in the batch, online, and approximate settings. Here we consider the case in which we can compute the subgradients exactly. Approximate settings are analyzed in Section 6. A number of proofs have been omitted due to their length; they can be found in the extended version of this paper (Ratliff et al., 2006a)

5.1 Convergence rate of batch optimization

To bound the convergence rate of the subgradient method in the batch setting we adapt some results introduced by (Nedic & Bertsekas, 2000) who analyze *incremental* subgradient algorithms. The theorem shows that we attain linear convergence to a small region of the minimum using a constant stepsize. This result require a strong convexity assumption to hold for the objective function. Given $\mathcal{W} \subseteq \mathbb{R}^d$, a function $f : \mathcal{W} \rightarrow \mathbb{R}$ is μ-strongly convex if there exists $g : \mathcal{W} \rightarrow \mathbb{R}^d$ such that for all $w, w' \in \mathcal{W}$:

$$f(w') \geq f(w) + g_w^T(w' - w) + \mu\|w' - w\|^2. \qquad (14)$$

In our case, both the MMSC and MMSR objective functions are are $\frac{\lambda}{2}$-strongly convex because of their regularization term.

Theorem 5.1: Linear convergence of constant stepsize sequence. *Let the stepsize sequence $\{\alpha_t\}$ of Algorithm (3) be chosen as $\alpha_t = \alpha \leq \frac{1}{\lambda}$. Furthermore, assume that for a particular region of radius R around the minimum, $\forall w, g \in \partial c(w)$, $\|g\| \leq C$. Then the algorithm converges at a linear rate to a region of the minimum $w^* = \arg\min_{w \in \mathcal{W}} c(w)$ bounded by $\|w_{\min} - w^*\| \leq \sqrt{\frac{\alpha C^2}{\lambda}} \leq \frac{C}{\lambda}$.*

[4] This projection operator need not be exact. See (Ratliff et al., 2006b) for details.

5.2 Regret bound in the online setting

In this section, we analyze the online setting of MMSC. Following online convex programming framework of (Zinkevich, 2003), our sequence of convex objective functions can be written as follows: $c_t(w) = \frac{\lambda}{2}\|w\|^2 + \max_{y \in \mathcal{Y}_t}(w^T f_t(y) + \mathcal{L}_t(y)) - w^T f_t(y_t) = \frac{\lambda}{2}\|w\|^2 + r_t(w)$. which we evaluate given y_t, $\mathcal{Y}_t$, and $f_t(\cdot)$. Note that these are $\frac{\lambda}{2}$-strongly convex.

(Hazan et al., 2006) have shown that with this learning rate, our online optimization problem has logarithmic regret with respect to the objective function. However, the loss we truly care about on round t is the prediction loss, $\mathcal{L}_t(y_t^*)$, where y_t^* is the prediction made during this round. The following may be derived using tools from (Hazan et al., 2006):

Theorem 5.2: Sublinear regret for subgradient MMSC. *Assume that the features in each state are bounded in norm by 1, then:*

$$\sum_{t=1}^T \mathcal{L}_t(y_t^*) \leq \sum_{t=1}^T r_t(w^*) + \lambda T\|w^*\|^2 + \frac{1}{\lambda}(1 + \ln T) \quad (15)$$

Choosing $\lambda = \frac{\sqrt{1+\ln T}}{\|w^*\|\sqrt{T}}$, then:

$$\sum_{t=1}^T \mathcal{L}_t(y_t^*) \leq \sum_{t=1}^T r_t(w^*) + \|w^*\|\sqrt{T(1 + \ln T)} \quad (16)$$

Thus, if we know our horizon T and the achievable margin, our loss grows only sublinearly with time.

5.3 Generalization guarantees on out-of-sample examples for MMSC

Our online algorithm also inherits interesting generalization guarantees when applied in the batch setting. Given independent, identically distributed data, the expected loss of our algorithm can be bounded, with probability greater than or equal to $1-\delta$, by the errors it makes at each step of the incremental subgradient method using the techniques of (Cesa-Bianchi et al., 2004):[5]

$$E[\mathcal{L}_{T+1}(\bar{w})] \leq \frac{1}{T}\sum_{t=1}^T r_t(w_t) + \sqrt{\frac{2}{T}\log\left(\frac{1}{\delta}\right)} \quad (17)$$

[5] To achieve this result we must actually use the average weight vector $\bar{w}$ computed during learning, not merely the last one.

This bound is similar in form to previous generalization bounds given using covering number techniques (Taskar et al., 2003). Importantly, though, this approach removes the dependence entirely on the number of bits b being predicted in structured learning; most existing techniques introduce a $\log b$ factor for the number of predicted bits. Similar results hold for our regression algorithm. A similarly tight bound does not hold for the loss functions considered in (LeCun et al., 2007).

6 Robustness to approximate settings

This section derives two robustness results. In the first subsection, we consider the case in which inference is only approximate, and in the second subsection we analyze the case in which we can only compute approximate subgradients of the structured margin objective. Unfortunately, we find that the approximate subgradient resulting from approximate inference is not that which is needed in the latter theoretical analysis, but nevertheless these results illustrate a general robustness in our algorithm.

6.1 Using approximate inference

Following (Shmoys & Swamy, 2004), we define a γ-subgradient similar to the way an exact subgradient is defined in Definition 4.1, but we replace the inequality with $\forall w' \in \mathcal{W},\ h(w') \geq h(w) + g^T(w'-w) - \gamma h(w)$. In other words, we allow the lower bound to be violated slightly by an amount that scales with the approximation constant γ and objective value $h(w)$ at the point in question.

Additionally, we define an approximate inference operator η-max as follows:

Definition 6.1 : η-max. *We call an algorithm an η-approximate max operator, denoted* $\max^\eta$*, if for any collection* $\{s_y \mid y \in \mathcal{Y}\}$*, we are guaranteed* $\max^\eta_{y\in\mathcal{Y}} s_y \geq \eta \max_{y\in\mathcal{Y}} s_y$. *$\eta$ is known as the competitive ratio of the approximate max.*

It is well known that if each s_y is a convex function over $\mathcal{W}$, then $h(w) = \max_{y\in\mathcal{Y}} s_y(w)$ is a convex function and $\nabla s_{y^*}(w)$ is a subgradient of that function for any $y^* = \arg\max_{y\in\mathcal{Y}} s_y(w)$. We prove here a generalized theorem of this sort in terms of an approximate max operator.

Theorem 6.2: η-max gives $(1-\eta)$-subgradient. *Define* $h = \max_{y\in\mathcal{Y}} s_y(w)$ *and let* $g = \nabla s_{y^*_\eta}(w)$ *where* $y^*_\eta = \arg\max^\eta_{y\in\mathcal{Y}} w_y(w)$. *Then g is a $(1-\eta)$-subgradient per Definition 4.1.*

Proof. Since g is a subgradient of the score function $s_{y^*_\eta}(w)$, we have $g^T(w'-w) \leq s_{y^*_\eta}(w') - s_{y^*_\eta}(w) \leq h(w') - \eta h(w)$, where the final inequality comes from the optimality of h and the definition of η-max. Rearranging, we get
$h(w') - h(w) \geq g^T(w'-w) - (1-\eta)h(w)$. □

6.2 Optimizing with approximate subgradients

In this section, we can bound the regret of following approximate subgradients rather than exact subgradients within the online setting defined in Section 5.2. Borrowing notation from that Section and following arguments similar to those that lead to Theorem 5.2, we can derive the following

$$\sum_{t=1}^{T}\mathcal{L}_t(y_t^*) \leq \sum_{t=1}^{T} c_t(w_t)$$
$$\leq \sum_{t=1}^{T} r_t(w^*) + \|w^*\|\sqrt{T(1+\ln T)} + \gamma\sum_{t=1}^{T} c_t(w_t)$$

Another, potentially more insightful, way to write this is in terms of the average regret. In this case, if we denote $S(T) = \|w^*\|\sqrt{T(1+\ln T)}$ (note that this is a sublinear function), we find

$$\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_t(y_t^*) \leq \frac{1}{1-\gamma}\left(\frac{1}{T}\sum_{t=1}^{T} r_t(w^*) + \frac{S(T)}{T}\right) \quad (18)$$
$$\underset{T\to\infty}{\longrightarrow} \frac{1}{1-\gamma}R, \quad (19)$$

where $R = \lim_{T=\infty}\frac{1}{T}\sum_{t=1}^{T} r_t(w^*)$ is the asymptotic optimal average risk. Equation 19 says that in the limit, we have paid on average only a factor $\frac{1}{1-\gamma}$ more regret each time step than if we had been able to compute and follow exact subgradients.

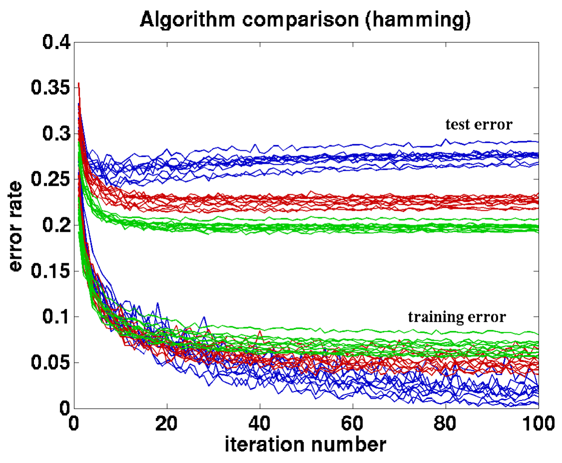
7 Experimental results

We present experimental results on two previously studied structured classification problems: optical character recognition (Taskar et al., 2003), and ladar classification (Anguelov et al., 2005). Additionally, we present results on a new problem for maximum margin structured regression: value function approximation. We compare the latter to a method demonstrated (Ratliff et al., 2006c).

7.1 Optical character recognition

We implemented the incremental subgradient method[6] for the sequence labeling problem originally explored by (Taskar et al., 2003) who used the Structured SMO

[6]Similar to the online method, this method updates the weights with each term's subgradient contribution rather than combining them into a single step.



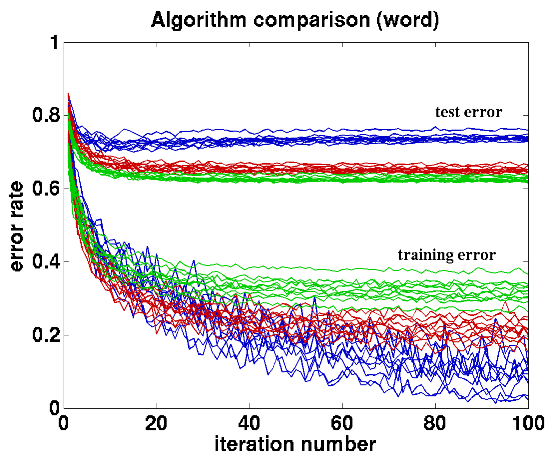


Figure 1: These plots show a comparison between the structured margin (green), perceptron (blue), and unstructured margin (red) algorithms using 10 fold cross-validation iterations of 600 training examples and 5500 test examples. The figure on the left displays error in terms of hamming loss, and the figure on the right displays word classification error. Upper lines of a given color represent test error and lower lines represent training error. See text for details.

algorithm.[7] Running our algorithm with 600 training examples and 5500 test examples using 10 fold cross validation, as was done in (Taskar et al., 2003), we attained an average prediction error of 0.20 using a linear kernel. This result is statistically equivalent to the previously published result; however, the entire 10 fold cross validation run completed within 17 seconds. Furthermore, when running the experiment using the entire data set partitioned into 10 folds of 5500 training and 600 test examples each, we achieved a significantly lower average error of 0.13, again using the linear kernel.

We additionally compared our algorithm to two previously proposed algorithms: the perceptron algorithm, and the unstructured margin (LeCun et al., 2007).[8] We ran each algorithm using 10 fold cross validation with the partitioning of 600 training examples and 5500 test examples. Figure 1 plots both the training error (lower lines) and the test error (upper lines) for each in terms of both hamming loss (left) and word classification (right). The structured margin algorithm (our algorithm), displayed in green, generalizes noticeably better than the other two algorithms. The perceptron algorithm (blue) overfits very quickly on this problem, and the unstructured margin algorithm (red) falls somewhat between the other two in terms of performance. In all cases, we used a stepsize rule of $\alpha_t = \frac{1}{2\sqrt{t}}$ and set the regularization constant to $\lambda = \frac{1}{200N}$ where N is the number of training examples.

[7]This data can be found at `http://www.cs.berkeley.edu/~taskar/ocr/`

[8]The perceptron risk is given by $r_i(w) = \max_{y\in\mathcal{Y}_i} w^T f_i(y) - w^T f_i(y_i)$; The unstructured margin risk is given by $r_i(w) = \max\{0, 1 + \max_{y\in\mathcal{Y}_i\setminus y_i} w^T f_i(y) - w^T f_i(y_i)\}$.

7.2 Ladar scan classification

We next consider application of subgradient techniques to a problem of classifying ladar point clouds captured by a mobile robot. Full details of the training data can be found in (Anguelov et al., 2005). Briefly, a maximum margin structured classification problem is set up to classify each point in a point cloud of laser range data into one of four classes: ground, shrubbery, trees, and building. One-vs-all classification of ground based on a height threshold was reportedly simple, effectively reducing the problem to a three class classification problem (per ladar point).

To capture spatial correlation between classification labels of the ladar points, an associative conditional Markov random field between nearby points was constructed throughout the point cloud. Labels for the point clouds were determined by the joint maximum probability labeling of the nodes in the Markov network. (Anguelov et al., 2005) built the maximum margin structured classification problem as a quadratic program (QP) and solved it using CPLEX, a well known commercial solver. Node potentials were log-linear in 90 features each derived from the original ladar data (e.g. spin images features, distance from ground) and edge potentials were constant for each class. See (Anguelov et al., 2005) for more additional information on the features.

Limited by CPLEX's fairly intensive memory requirements, the training set consisted of only approximately 30 thousand of the original 20 million points in the data set. We note that the subgradient methods we here have only linear memory requirements in the number of training points.

Moreover, the quadratic programming problem used for training was derived as a relaxation to the

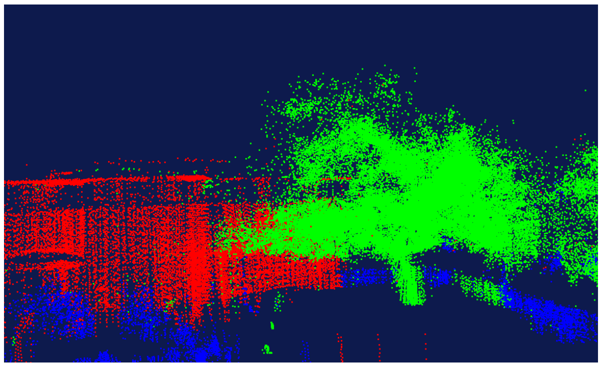
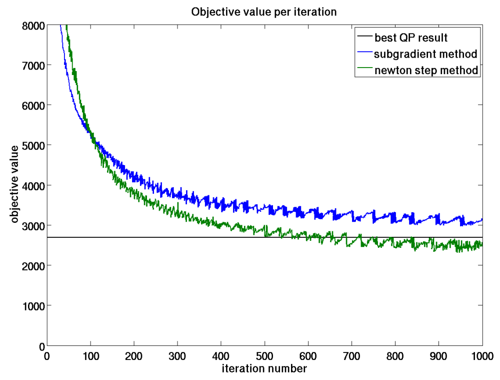



Figure 2: Left: Pictorial representation of ladar classification results on a test region. Classes are denoted as red: building, green: tree, and blue: shrubbery. Right: Ladar scan classification results. Subgradient method (blue) converges off the edge of the graph, but within the same amount of time as it took to obtain the best QP result. The Newton Step method converges significantly faster. See Section 7.2 for details.

intractable integer programming problem, but the alpha-beta swap/expansion algorithm (Szeliski et al., 2006) was employed for approximate inference at test time. While both of these algorithms admit a constant factor approximation, they qualitatively differ in practice. The subgradient method has the additional appeal of relying solely on the alpha-beta swap/expansion algorithm (Szeliski et al., 2006), iteratively optimizing it to perform well.

We ran the subgradient method and a modified approximate Newton step method[9] to optimize this problem, the results of which are shown in Figure 2. We preprocessed the node features using a whitening operation to remove linear dependencies and poor conditioning of the features. Whitening intuitively amounts to scaling the principle directions of variance of the feature vectors inversely proportional to the standard deviation along those directions.

The black horizontal line across Figure 2 denotes the minimum objective value attained by CPLEX on this problem, and the blue and green plots, respectively, show the objective values per iteration of the subgradient method and the Newton step method. The Newton step objective progression drops below the smallest CPLEX value within 550 iteration, which is equivalent to approximately 15 minutes of CPU time. This computation time is primarily dominated by executing of the alpha-beta expansion algorithm (Szeliski et al., 2006). While the first-order subgradient method lags behind the Newton step counterpart, it is important to note that it also does well, surpassing the CPLEX result by iteration 1950. This amounts to approximately 65 minutes of computation time, the same amount of time as was reported in (Anguelov et al., 2005) for CPLEX training. Importantly, however, both of these subgradient-based algorithms scale to data set sizes significantly greater than those reported here, which neared the upper bound of what CPLEX could originally handle. Indeed, they are limited solely by the computational performance of the inference algorithm.

7.3 Value function approximation

Maximum margin structured regression is well suited for learning a value function approximator. Value function approximators attempt to estimate the cost-to-go of running a policy from any state. It can be very difficult to design features (especially to generalize across different goals and problems) that can capture the cost-to-go. Even in problems with higher dimensional state-spaces it is often the case that a simple, lower dimensional state-space can explain much of the value for a given state. For instance, in mobile robot navigation, much of the cost-to-go is captured by considering a simple two-dimensional holonomic planning problem.

We consider the heuristic value function learning problem studied in Section 4.3 of (Ratliff et al., 2006c). At a high level, a value function approximator is learned to approximate the value of a policy acting within a space of footsteps on a quadrupedal robot. An optimal policy is available through the path returned by footstep planner, and a set of examples can be collected accordingly. Each example consists of the navigational trajectory induced by the optimal footstep plan between two points in the world and the corresponding cost of that plan. The objective is to learn a navigational cost map using this data for which the cost-to-go

[9] This more complex variant works better in practice for certain problems and is an extension of Newton type methods to nondifferentiable problems where the Hessian might not exist. See (Hazan et al., 2006) for details and analysis of this method. Briefly, under the Newton step method, the update rule becomes $w_{t+1} \leftarrow w_t - \alpha_t(H_t + \epsilon I)^{-1}g_t$, where g_t is the subgradient at time t and H_t is updated as $H_{t+1} \leftarrow \frac{t}{t+1}H_t + \frac{1}{t+1}g_t g_t^T$.

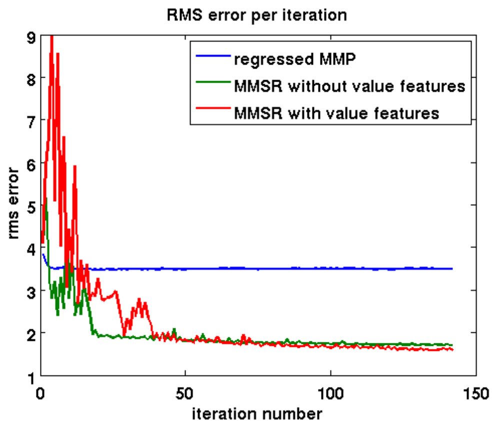


Figure 3: MMSR Value function approximation results. See text for details.

of the optimal paths approximates these values.

The method studied in (Ratliff et al., 2006c), is built atop a maximum margin structured classification algorithm applied to learning MDPs, known as Maximum Margin Planning (MMP). As such, the primary learning algorithm cares more about getting the shape of the path correct than getting the value correct. The resulting path costs are consistent only relative to the costs of other paths (e.g. a scaled cost map produces the same minimum cost paths as the original cost map). For that reason, once the cost map is learned, the best scaling of that map is learned via simple linear regression. We refer to this method as regressed MMP.

Maximum margin structured regression provides a more direct way to learn such a cost function. By construction, MMSR utilizes the provided data to learn precisely the cost map we desire: that for which the minimum cost paths match the supervised values. Figure 3 demonstrates the performance improvement seen over the previous suggested regressed MMP method (blue). Within a relatively small number of iterations, MMSR (green) surpasses the best rms error attained by the more indirect method. Additionally, we experimented with augmenting the objective function using a linear combination of "value features" (i.e. features that are not plugged in as costs to the lower-dimensional planner, but are instead fit directly to help estimate the cost-to-go.) such as distance-to-goal to improve the approximation. The result, shown in red, shows a slight improvement over the basic method.

Acknowledgements

The first two authors gratefully acknowledge the partial support of this research by the DARPA Learning for Locomotion contract. We gratefully acknowledge Drago Anguelov for providing us with the ladar data set and inference implementation.

References

Anguelov, D., Taskar, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., & Ng, A. (2005). Discriminative learning of markov random fields for segmentation of 3d scan data. *Conference on Computer Vision and Pattern Recognition*. San Diego, CA.

Cesa-Bianchi, N., Conconi, A., & Gentile, C. (2004). On the generalization ability of on-line learning algorithms. *IEEE Trans. on Information Theory* (pp. 2050–2057). Preliminary version in Proc. of the 14th conference on Neural Information processing Systems (NIPS 2001).

Hazan, E., Kalai, A., Kale, S., & Agarwal, A. (2006). Logarithmic regret algorithms for online convex optimization. To appear in COLT 2006.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE* (pp. 2278–2324).

LeCun, Y., Chopra, S., Hadsell, R., Huang, F.-J., & Ranzato, M.-A. (2007). A tutorial on energy-based learning. In *Predicting structured outputs*. The MIT Press.

Nedic, A., & Bertsekas, D. (2000). Convergence rate of incremental subgradient algorithms. *Stochastic Optimization: Algorithms and Applications*.

Ratliff, N., Bagnell, J. A., & Zinkevich, M. (2006a). *(approximate) subgradient methods for structured prediction* (Technical Report). Carnegie Mellon University.

Ratliff, N., Bagnell, J. A., & Zinkevich, M. (2006b). Maximum margin planning. *Twenty Second International Conference on Machine Learning (ICML06)*.

Ratliff, N., Bradley, D., Bagnell, J. A., & Chestnutt, J. (2006c). Boosting structured prediction for imitation learning. *NIPS*. Vancouver, B.C.

Shmoys, D., & Swamy, C. (2004). An approximation scheme for stochastic linear programming and its application to stochastic integer programs. *FOCS*.

Shor, N. Z. (1985). *Minimization methods for non-differentiable functions*. Springer-Verlag.

Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., & Rother, C. (2006). A comparative study of energy minimization methods for markov random fields. *European Conference on Computer Vision* (pp. II: 16–29).

Taskar, B., Guestrin, C., & Koller, D. (2003). Max margin markov networks. *Advances in Neural Information Processing Systems (NIPS-14)*.

Taskar, B., Lacoste-Julien, S., & Jordan, M. (2006). Structured prediction via the extragradient method. In *Advances in neural information processing systems 18*.

Tsochantaridis, I., Joachims, T., Hofmann, T., & Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 1453–1484.

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. *Proceedings of the Twentieth International Conference on Machine Learning*.