Subgradient Methods for Maximum Margin Structured Learning

Nathan D. Ratliff NDR@RI.CMU.EDU
J. Andrew Bagnell DBAGNELL@RI.CMU.EDU
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA. 15213 USA

Martin A. Zinkevich MAZ@CS.UALBERTA.CA
Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E1, Canada

1. Introduction

Maximum margin structured learning (MMSL) has recently gained recognition within the machine learning community as a tractable method for large scale learning. However, most current methods are limited in terms of scalability, convergence, or memory requirements. The original Structured SMO method proposed in (Taskar et al., 2003) is slow to converge, particularly for Markov networks of even medium treewidth. Similarly, dual exponentiated gradient techniques suffer from sublinear convergence as well as often large memory requirements. Recently, (Taskar et al., 2006) have looked into saddle-point methods for optimization and have succeeded in efficiently solving several problems that would have otherwise had intractable memory requirements.

We propose an alternative gradient based approach 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 becomes the well-studied 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.

In what follows, we review the basic formulation of maximum margin structured learning as a convex programming problem before deriving the convex objective and showing how its subgradients can be computed utilizing the specialized inference algorithm inherent to the problem. We finish with theoretical guarantees and some experimental results in two domains: sequence labeling for optical character recognition; and imitation learning for path planning in mobile robot navigation. The former problem is well known to MMSL, but the latter is new to this domain. Indeed, although there is a tractable polynomial sized quadratic programming representation for the problem (Ratliff et al., 2006), solving it directly using one of the previously proposed methods would be intractable practically for reasons similar to those that arise in directly solving the linear programming formulation of Markov Decision Processes.

2. Maximum margin structured learning

We present a brief review of maximum margin structured learning in terms of convex programming. In this setting, we attempt to predict a structured object $y \in \mathcal{Y}(x)$ 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:

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

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:

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

where $\lambda \geq 0$ is a hyperparameter that trades off constraint violations for margin maximization (i.e. fit for simplicity), $q \geq 1$ defines the penalty norm, and $\beta_i \geq 0$ are constants that scale training examples relative to each other. β_i can be used to give training examples equal weight regardless of their differing structure. See Section 6 for a concrete example of this in the case of planning.

3. Subgradient methods and MMSL

We propose rewriting Program 3 as a regularized risk function and taking subgradients of the resulting objective. This leads to a myriad of possible algorithms, the simplest of which is the subgradient method for convex optimization. This method has shown promising experimental results, and as a direct generalization of gradient descent for differentiable functions, is easy to implement and has good theoretical properties in both batch and online settings.

The regularized risk interpretation can be easily derived by noting that the slack variables ζ_i in Equation 4 are tight and thus equal[1] to $\max_{y \in \mathcal{Y}_i}(w^T f_i(y) + \mathcal{L}_i(y)) - w^T f_i(y_i)$ at the minimum. We can therefore move these constraints into the objective function, simplifying the program into a single cost function:

$$\begin{aligned} c(w) =& \frac{1}{n}\sum_{i=1}^{n} \beta_i \left(\max_{y \in \mathcal{Y}_i}(w^T f_i(y) + \mathcal{L}_i(y)) - w^T f_i(y_i)\right)^q \\ &+ \frac{\lambda}{2}\|w\|^2 \end{aligned} \tag{4}$$

This objective is convex, but nondifferentiable; we can optimize it by utilizing the subgradient method (Shor, 1985). A *subgradient* of a convex function $c : \mathcal{W} \to \mathbb{R}$ at w is defined as a vector g_w for which

$$\forall w' \in \mathcal{W},\ g_w^T(w' - w) \leq c(w') - c(w) \tag{5}$$

Note that subgradients need not be unique, though at points of differentiability, they necessarily agree with the gradient. We denote the set of all subgradients of $c(\cdot)$ at point w by $\partial c(w)$.

To compute the subgradient of our objective function, we make use of the following four well known properties: (1) subgradient operators are linear; (2) the gradient is the unique subgradient of a differentiable function; (3) if $f(x, y)$ is differentiable in x, then $\nabla_x f(x, y*)$ is a subgradient of the convex function $\phi(x) = \max_y f(x, y)$ for any $y^* \in \arg\max_y f(x, y)$; (4) an analogous chain rule holds as expected. We are now equipped to compute a subgradient $g_w \in \partial c(w)$ of our objective function (4):

$$\begin{aligned} g_w =& \frac{1}{n}\sum_{i=1}^{n} q\beta_i \left((w^T f_i(y^*) + \mathcal{L}_i(y^*)) - w^T f_i(y_i)\right)^{q-1} \cdot \\ &\Delta^w f_i^* + \lambda w \end{aligned} \tag{6}$$

where $y^* = \arg\max_{y \in \mathcal{Y}_i}(w^T f_i(y) + \mathcal{L}_i(y))$ and $\Delta^w f_i^* = f_i(y^*) - f_i(y_i)$. This latter expression emphasizes that, intuitively, the subgradient compares the feature values between the example class y_i and the current loss-augmented prediction y^*.

Note that computing the subgradient requires solving the problem $y^* = \arg\max_{y \in \mathcal{Y}_i}(w^T f_i(y) + \mathcal{L}_i(y))$ for each example. If we can efficiently solve $\arg\max_{y \in \mathcal{Y}_i} w^T f_i(y)$ using a particular specialized algorithm, we can often use the same algorithm to efficiently compute this loss-augmented optimization for a particular class of loss function and hence efficiently compute this subgradient. Algorithm 1 details the application of the subgradient method to maximum margin structured learning.

Given $g_t \in \partial c(w_t)$ and α_t, the basic iterative update is

$$w_{t+1} = \mathcal{P}_{\mathcal{W}}\left[w_t - \alpha_t g_t\right] \tag{7}$$

where $\mathcal{P}_{\mathcal{W}}$ projects w onto a convex set $\mathcal{W}$ formed by any problem specific convex constraints we may impose on w.[2]

3.1. Optimization in the batch setting

In the batch setting, this algorithm is one of a well studied class of algorithms forming the subgradient method (Shor, 1985).[3] Crucial to this method is the choice of stepsize sequence $\{\alpha_t\}$, and convergence guarantees vary accordingly. Our results are developed

[1] The right hand term maximizes over a set that includes y_i, and $\mathcal{L}_i(y_i) = 0$ by definition.

[2] It is actually sufficient that $\mathcal{P}_{\mathcal{W}}$ be an approximate projection operator for which $\mathcal{P}_{\mathcal{W}}[w] \in \mathcal{W}$ and $\forall w' \in \mathcal{W},\ \|\mathcal{P}_{\mathcal{W}}[w] - w'\| \leq \|w - w'\|$.

[3] The term "subgradient method" is used in lieu of "subgradient descent" because the method is not technically a descent method. Since the stepsize sequence is chosen in advance, the objective value per iteration can, and often does, increase.

Algorithm 1 Subgradient Method for Maximum Margin Structured Learning

procedure sMMSL($\{x_i, y_i, f_i(\cdot), \mathcal{L}_i(\cdot)\}_{i=1}^n$, Regularization parameter $\lambda > 0$, Stepsize sequence $\{\alpha_t\}$ (learning rate), Iterations T)
  $t \leftarrow 1$
  $w \leftarrow 0$
  while $t \leq T$ **do**
    Solve $y_i^* = \arg\max_{y \in \mathcal{Y}_i}(w^T f_i(y) + \mathcal{L}_i(y))$ for each i.
    Compute $g \in \partial c(w)$ as in Equation 6.
    Update $w \leftarrow w - \alpha_t g$
    (Optional): Project w on to any additional constraints.
    $t \leftarrow t + 1$
  end while
  return w
end procedure

from (Nedic & Bertsekas, 2000) who analyze *incremental* subgradient algorithms, of which the subgradient method is a special case.

Our results require a strong convexity assumption to hold for the objective function. Given $\mathcal{W} \subseteq \mathbb{R}^d$, a function $f : \mathcal{W} \to \mathbb{R}$ is η-strongly convex if there exists $g : \mathcal{W} \to \mathbb{R}^d$ such that for all $w, w' \in \mathcal{W}$:

$$f(w') \geq f(w) + g_w^T(w' - w) + \eta\|w' - w\|^2. \quad (8)$$

In our case, Equation 4 is $\frac{\lambda}{2}$-strongly convex.

Theorem 3.1: Linear convergence of constant stepsize sequence. *Let the stepsize sequence $\{\alpha_t\}$ of Algorithm (1) 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 $\|w_{\min} - w^*\| \leq \sqrt{\frac{\alpha C^2}{\lambda}} \leq \frac{C}{\lambda}$.*

Proof: (Sketch) By the strong convexity of $c_q(w)$ and Proposition 2.4 of (Nedic & Bertsekas, 2000) we have

$$\|w_{t+1} - w^*\|^2 \leq (1 - \alpha\lambda)^{t+1}\|w_0 - w^*\|^2 + \frac{\alpha C^2}{\lambda}$$
$$\xrightarrow[t \to \infty]{} \frac{\alpha C^2}{\lambda} \leq \frac{C^2}{\lambda^2}$$

□

This theorem shows that we attain linear convergence to a small region of the minimum using a constant stepsize. Alternatively, we can choose a diminishing stepsize rule of the form $\alpha_t = \frac{\gamma}{t}$ for $t \geq 1$, where γ is some positive constant that can be thought of as the learning rate. Under this rule, Algorithm 1 is guaranteed to converge to the minimum, but only at a sublinear rate under the above strong convexity assumption (see (Nedic & Bertsekas, 2000), Proposition 2.8).

3.2. Optimization in an online setting

In contrast with many optimization techniques, the subgradient method naturally extends from the batch setting (as presented) to an online setting. In the online setting one imagines seeing several problems on closely related domains: in particular, one may observe a domain, be required to predict within it, and only then observe the "correct" solution (or observe the corrections to the predicted solution).

At each time step t, we are given $\mathcal{Y}_t$ and $f_t(\cdot)$ with which to select a weight vector w_t and make a prediction. Once this prediction is made, we can then observe the true class y_t and update the weight vector based on the error. Thus, we can define $c_t(w) = \frac{\lambda}{2}\|w\|^2 + \max_{y \in \mathcal{Y}_t}(w^T f_t(y) + \mathcal{L}_t(y)) - w^T f_t(y_t) = \frac{\lambda}{2}\|w\|^2 + r_t(w)$ to be the $\frac{\lambda}{2}$-strongly convex cost function (see Equation 4) at time t, which we can evaluate given y_t, $\mathcal{Y}_t$, and $f_t(\cdot)$. This is now an **online convex programming problem** (Zinkevich, 2003), to which we will apply Greedy Projection with learning rate $1/(t\lambda)$.

(Hazan et al., 2006) have shown that with this learning rate, our online optimization problem has logarithmic regret with respect to the objective funtion. 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. Space doesn't permit a proof, but the following may be derived using tools from (Hazan et al., 2006):

Theorem 3.2: Sublinear regret for subgradient MMSL. *Assume that the features in each state are bounded in norm by 1, then:*

$$\sum_{t=1}^{T} \mathcal{L}_t(y_t^*) \leq \sum_{t=1}^{T} r_t(w^*) + \lambda T\|w^*\|^2 + \frac{1}{\lambda}(1 + \ln T) \quad (9)$$

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

$$\sum_{t=1}^{T} \mathcal{L}_t(y_t^*) \leq \sum_{t=1}^{T} r_t(w^*) + \|w^*\|\sqrt{T(1 + \ln T)} \quad (10)$$

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

Observe that Theorem 3.2 is a result about the additive loss functions $\mathcal{L}_t$ specifically. If we attempted instead to consider, for instance, a setting where we did not require more margin from higher loss paths

(e.g. 0/1 loss on paths) our bound would be much weaker and scale with the size of the domain.

4. Generalization guarantees

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):[4]

$$E[\mathcal{L}_{T+1}(\bar{w})] \leq \frac{1}{T}\sum_{t=1}^{T} r_t(w_t) + \sqrt{\frac{2}{T}\log\left(\frac{1}{\delta}\right)} \qquad (11)$$

This bound is rather 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.

5. Slack-scaling

In principle, we can use these tools to compute subgradients of the slack-scaling formulation detailed in (Tsochantaridis et al., 2005). Disregarding the regularization, under this formulation Equation 4 becomes

$$\tilde{c}(w) = \frac{1}{n}\sum_{i=1}^{n} \beta_i \left\{ \max_{y \in \mathcal{Y}_i} \mathcal{L}_i(y) \left(w^T \left(f_i(y) - f_i(y_i) \right) + 1 \right) \right\}^q$$

Multiplying by the loss inside the maximization makes this expression more restrictive in terms of optimization since the specialized inference algorithm cannot usually be applied directly. However, when the loss function takes on only a relatively small number of values $\mathcal{L}_i(y) \in \{l_j\}_{j=1}^k$ for all $y \in \mathcal{Y}_i$, we can often solve the inner inference problem for each j individually with respect to the set $\mathcal{Y}_i^{(j)} = \{y \in \mathcal{Y}_i \mid \mathcal{L}_i(y) = l_j\}$. Using this we can then compute the subgradient and utilize the subgradient method as before to optimize the objective.

6. Experimental results

We present results for two problems: sequence labeling for optical character recognition (Taskar et al., 2003); and imitation learning for path planning in mobile robot navigation (Ratliff et al., 2006).

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

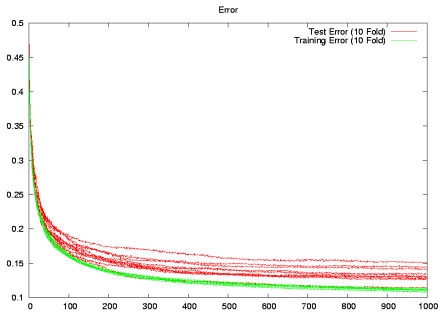

Figure 1. Training error (green) and test error (red) for each iteration of 10 fold cross validation using 5500 training and 600 validation examples.

6.1. Optical character recognition

We implemented the incremental subgradient method for the sequence labeling problem originally explored by (Taskar et al., 2003) who used the Structured SMO algorithm.[5] 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. Figure 1 shows the error per iteration for the larger experiment.

6.2. Path planning

We present briefly a result from our implementation of the batch learning algorithm for the MMSL approach to imitation learning. Details and additional results can be found in (Ratliff et al., 2006). In this setting, the objective is to learn to predict the correct path between two points in a world given features over the underlying MDP. Examples show the sequence of decisions a teacher might make through a number of regions. In order to give approximately equal weight to each example in the objective function given by equation 4, we chose $\beta_i = |y_i|^{-q}$, where $|y_i|$ denotes the length the ith example path. We used a variant of A* as our specialized algorithm to solve the inference problem represented by Equation 1.

Differing example trajectories, demonstrated in a par-

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

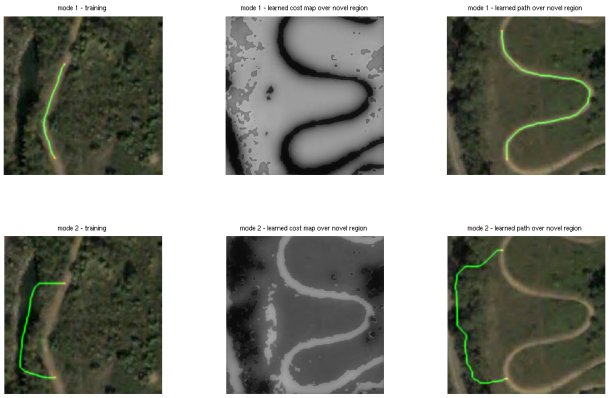

Figure 2. Demonstration of learning to plan based on satellite color imagery. For a particular training/holdout region pair, the top row trains the learner to follow the road while the bottom row trains the learner to "hide" in the trees. From left to right, the columns depict the single training example presented, the learned cost map over the holdout region, and the corresponding learned behavior over that region. Cost scales with intensity.

ticular region of a map, lead to significantly different behavior in a separate holdout region after learning. Figure 2 shows qualitatively the results of this experiment. The behavior presented in the top row suggests a desire to stay on the road, while the bottom row portrays a more clandestine behavior. By column from left to right, the images depict the training example presented to the algorithm, the learned cost map on a holdout region after training, and the resulting behavior produced by A* over this region.

7. Related and future work

The subgradient optimization presented here makes it practical to learn in problems for which straightforward QP techniques are intractable. This is exemplified in case of path planning. In this senario, subgradient methods for maximum margin structured learning converge quickly while the explicit quadratic program would be too large to even represent for a generic QP solver. Recently, a number of other techniques have been proposed to solve problems of these kinds, including cutting plane (Tsochantaridis et al., 2005) and extragradient techniques (Taskar et al., 2006). The latter is applicable where inference may be written as a linear program and is also able to achieve linear convergence rates. The subgradient method has the advantage of being applicable to any problems where loss augmented inference may be quickly solved including by combinatorial methods. Further, our algorithm extends naturally to the online case, where sublinear regret bounds are available. It will be interesting to compare these methods on problems where they are both applicable. In recent work, (Duame et al., 2006) has considered reinforcement learning based approaches to structured classification. Subgradient methods for (unstructured) margin linear classification were considered in (Zhang, 2004). (LeCun et al., 1998) considers the use of gradient methods for learning using decoding methods such as Viterbi; our approach (if applied to sequence labeling) extends such methods to use notions of structured maximum margin.

Acknowledgements

The first two authors gratefully acknowledge the partial support of this research by the DARPA Learning for Locomotion contract.

References

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).

Duame, H., Langford, J., & Marcu, D. (2006). Search-based structured prediction. In Preparation.

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).

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

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

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

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.

Zhang, T. (2004). Solving large scale linear prediction problems using stochastic gradient descent algorithms. *Proceedings of ICML*.

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