CSC 588: Machine learning theory **Spring 2021**

Lecture 22: OGD for Strongly Convex Functions and Kernel Methods

Lecturer: Chicheng Zhang *Scribe: Zisu Wang*

1 Fast Algorithms for Regularized Loss Minimization

In today's lecture, we will discuss how to exploit the strong-convexity structure of functions to conduct online convex optimization (OCO), as well as kernel methods. For motivation, let us consider the following familiar example on regularized loss minimization:

$$\min_w \underbrace{\frac{1}{m}\sum_{i=1}^{m} \ell(w,(x_i,y_i)) + \frac{\lambda}{2}||w||_2^2}_{F_S(w)}, \qquad ||x_i||_2 \le B, \tag{1}$$

where $\ell(w,(x_i,y_i))$ is the loss function for sample (x_i,y_i) given w, such as logistic loss or hinge loss. Then, our goal is to approximately optimize the objective (1), i.e., find w such that

$$F_S(w) \le F_S(\hat{w}) + \epsilon, \qquad \hat{w} = \operatorname*{argmin}_w F_S(w). \tag{2}$$

To do this, we can consider the following ideas to achieve fast convergence.

- **Idea 1: Gradient Descent**

  Algorithm 1 lists the implementation of Gradient Descent. For each iteration t, we use the current iterate w_t and the negative (sub)gradient of the objective F_S to update the next iterate w_{t+1}, i.e., $w_{t+1} \leftarrow w_t - \eta\nabla F_S(w_t)$. To analyze the algorithm's convergence, we can apply the analysis of OCO seen before. Specifically, let $f_t = F_S$ and by the regret guarantee of online gradient descent (OGD), we can show that by appropriately tuning the step size η, we can guarantee that for the average iterate $\bar{w}_T = \frac{1}{T}\sum_{t=1}^{T} w_t$ satisfies $F_S(\bar{w}) - F_S(\hat{w}) \le O\left(\frac{1}{\sqrt{T}}\right)$. Then, by choosing the number of iterations $T = O(\frac{1}{\epsilon^2})$, we can find the suboptimal w that satisfies the objective (2).

  To derive the computational complexity of the approach, notice that for each iteration t, evaluating the full (sub)gradient $\nabla F_S(w_t)$ requires to calculate the (sub)gradient of individual losses $\nabla(\ell(w_t,(x_i,y_i)))$ for each sample (x_i,y_i), thus taking $O(m)$ time in total. Then, the total running time would be $O(\frac{m}{\epsilon^2})$.

Algorithm 1 Gradient Descent

Input: data $(x_i,y_i)_{i=1}^m$, initialize $w_1 \in \Omega$.
for $t = 1, 2, \cdots, T$ **do**
  $F_S(w_t) = \frac{1}{m}\sum_{i=1}^{m} \ell(w,(x_i,y_i)) + \frac{\lambda}{2}||w||_2^2$
  $w_{t+1} \leftarrow w_t - \eta\nabla(F_S(w_t))$
end for
Output: $\bar{w}_T = \frac{1}{T}\sum_{t=1}^{T} w_t$

- **Idea 2: Stochastic Gradient Descent**

  Define $\hat{D} = \text{uniform}((x_i,y_i)_{i=1}^m)$. Then we can apply the Stochastic Gradient Descent algorithm shown in Algorithm 2 by using OGD on the regularized losses induced by random examples drawn from $\hat{D}$ and doing online-to-batch conversion.

To analyze the algorithm, we can show that $\mathbb{E}[L_D(\bar{w}_T)] - L_D(w^*) \leq \mathbb{E}[R_T(w^*)] \leq O\left(\frac{1}{\sqrt{T}}\right)$, where $w^* = \text{argmin}_w L_D(w)$ based on the results from previous lecture. Then, as $F_S(\bar{w}_T) = L_D(\bar{w}_T)$ and $F_S(\hat{w}) = L_D(w^*)$, we have $\mathbb{E}[F_S(\bar{w}_T)] - F_S(\hat{w}) \leq O\left(\frac{1}{\sqrt{T}}\right)$. and we can thus find the suboptimal w that satisfies the objective (2) by letting $T = O\left(\frac{1}{\epsilon^2}\right)$,

However, the key insight of Stochastic Gradient Descent is that instead of evaluating the (sub)gradient of individual losses for all samples, we just need to calculate the (sub)gradient for only one training example in each iteration. Therefore, the running time for the algorithm is $O(\frac{1}{\epsilon^2})$, which significantly improves over the running time $O(\frac{m}{\epsilon^2})$ for Gradient Descent.

Algorithm 2 Stochastic Gradient Descent

Input: data $(x_i, y_i)_{i=1}^m$
for $t = 1, 2, \cdots, T$ **do**
  Sample $i_t \sim$ uniform$(\{1, 2, \cdots, m\})$
  $f_t(w) = \ell(w, (x_{i_t}, y_{i_t})) + \frac{\lambda}{2}||w||_2^2$
  $w_{t+1} \leftarrow w_t - \eta g_t$ where $g_t \in \partial f_t(w_t)$
end for
Output: $\bar{w}_T = \frac{1}{T}\sum_{t=1}^T w_t$

2 OGD with Time-Varying Step Size

As we have seen, both Gradient Descent and Stochastic Gradient Descent can achieve good convergence for the regularized loss minimization. However, it turns out that by applying OGD with time-varying step size, whose details are listed in Algorithm 3, we can further exploit the strong-convexity structure of the objective function (1) and get a better regret guarantee.

Algorithm 3 Online Gradient Descent with Time-Varying Step Size

Input: data $(x_i, y_i)_{i=1}^m$
for $t = 1, 2, \cdots, T$ **do**
  Sample $i_t \sim$ uniform$(\{1, 2, \cdots, m\})$
  $f_t(w) = \ell(w, (x_{i_t}, y_{i_t})) + \frac{\lambda}{2}||w||_2^2$
  $\eta_t = \frac{1}{\lambda t}$
  $w_{t+1} \leftarrow w_t - \eta_t g_t$ where $g_t \in \partial f_t(w_t)$
end for
Output: $\bar{w}_T = \frac{1}{T}\sum_{t=1}^T w_t$

Although Algorithm 3 looks similar to the Stochastic Gradient Descent algorithm, the step size in Algorithm 3 is $\eta_t = \frac{1}{\lambda t}$ and thus time-varying. From the following theorem, we can show that such time-varying step size, combined with the strong-convexity structure of the loss functions $\{f_t\}_{t=1}^T$, allows us to achieve a better regret guarantee when the (sub)gradient g_t is bounded.

Theorem 1. *Assume the decision set Ω is convex and $\{f_t\}_{t=1}^T$ are λ-SC w.r.t. $||\cdot||_2$. We run the OGD with time-varying step size as follows: $w_{t+1} \leftarrow w_t - \eta_t g_t$ where $g_t \in \partial f_t(w_t)$. Let $\eta_t = \frac{1}{\lambda t}$ and assume $\forall t, ||g_t||_2 \leq L$. Then, $\mathcal{R}_T(\Omega) \leq \frac{(\ln T+1)L^2}{2\lambda}$.*

Note.

1. The regret $\mathcal{R}_T(\Omega) \leq \frac{(\ln T+1)L^2}{2\lambda}$ is much better than the basic $O(\sqrt{T})$ regret by exploiting strong convexity.

2. Theorem 1 implies that running $\frac{1}{\lambda t}$-step-size OGD yields $F_S(\bar{w}_T) - F_S(\hat{w}) \leq \tilde{O}\left(\frac{1}{T}\right)$. Therefore, we only need to choose $T = O\left(\frac{1}{\epsilon}\right) \ll O\left(\frac{1}{\epsilon^2}\right)$.

Proof.

By quadratic lower bound property of λ-SC functions (see Figure 2), we have

$$f_t(w_t) - f_t(u) \leq \langle g_t, w_t - u \rangle - \frac{\lambda}{2}||w_t - u||_2^2, \tag{3}$$

which improves over the linearization step by utilizing λ-SC. Moreover, recall that in OGD analysis,

$$\langle g_t, w_t - u \rangle \leq \frac{||u - w_t||_2^2 - ||u - w_{t+1}||_2^2}{2\eta_t} + \frac{\eta_t}{2}||g_t||_2^2. \tag{4}$$

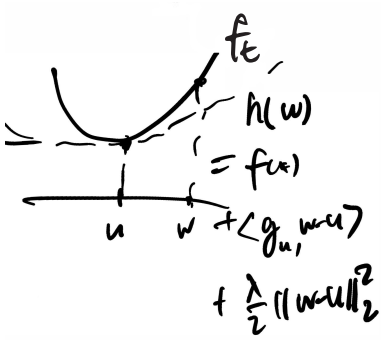

Figure 1: We can lower-bound the λ-SC function $f_t(w)$ by $h(w) = f(u) + \langle g_u, w - u \rangle + \frac{\lambda}{2}||w - u||_2^2$.

Combining (3) and (4), we have

$$\sum_{t=1}^{T} f_t(w_t) - f_t(u) \leq \sum_{t=1}^{T} \frac{||u - w_t||_2^2 - ||u - w_{t+1}||_2^2}{2\eta_t} - \sum_{t=1}^{T} \frac{\lambda}{2}||u - w_t||_2^2 + \sum_{t=1}^{T} \eta_t ||g_t||_2^2. \tag{5}$$

For the above inequality, we break down the summation of $||u - w_t||_2^2$ over t and check their coefficients:

- Coefficient of $||u - w_1||_2^2 : \frac{1}{2\eta_1} - \frac{\lambda}{2} = 0$;
- Coefficient of $||u - w_2||_2^2 : -\frac{1}{2\eta_1} + \frac{1}{2\eta_2} - \frac{\lambda}{2} = 0$;
- $\cdots$
- Coefficient of $||u - w_T||_2^2 : -\frac{1}{2\eta_{T-1}} + \frac{1}{2\eta_T} - \frac{\lambda}{2} = 0$.

Therefore, all the terms except $-\frac{||u - w_{T+1}||_2^2}{2\eta_T} \leq 0$ in $\sum_{t=1}^{T} \frac{||u - w_t||_2^2 - ||u - w_{t+1}||_2^2}{2\eta_t}$ will be cancelled out. We thus have

$$\begin{aligned}
\mathcal{R}_T(u) &\leq \sum_{t=1}^{T} \frac{\eta_t}{2}||g_t||_2^2 \leq \frac{L^2}{2\lambda}\left(\sum_{t=1}^{T} \frac{1}{t}\right) = \frac{L^2\left(1 + \sum_{t=2}^{T} \frac{1}{t}\right)}{2\lambda} \\
&\leq \frac{L^2\left(1 + \int_1^T \frac{1}{t} dt\right)}{2\lambda} = \frac{L^2\left(1 + \ln T\right)}{2\lambda},
\end{aligned}$$

where the first inequality comes from cancelling out $||u - w_t||_2^2$ terms, the second inequality comes from the assumption that $||g_t||_2 \leq L$, and the last inequality can be shown from the following fact that if f is decreasing, then $\sum_{i=k}^{l} f(i) \leq \int_{k-1}^{l} f(x)dx$, which is obvious from the following Figure 2. □

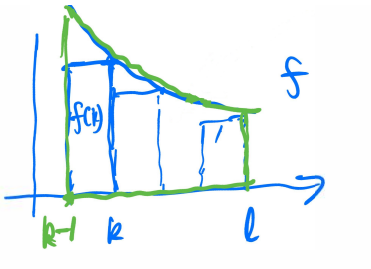

Figure 2: Pictorial illustration of the inequality $\sum_{i=k}^{k} f(i) \leq \int_{k-1}^{l} f(x)dx$ if f is decreasing. The green part area is $\int_{k-1}^{l} f(x)dx$ by integral, which is larger than the area of blue boxes $\sum_{i=k}^{k} f(i)$.

In Theorem 1, the (sub)gradient g_t is bounded. However, we can instantiate the result to unconstrained regularized loss minimization by relating the (sub)gradient of $f_t(w)$ with the (sub)gradients of $\ell(w, (x_{i_t}, y_{i_t}))$ and $\frac{\lambda}{2}||w||_2^2$ to upper-bound g_t. More specifically, it is easy to show that if f, g are convex, then for any w and $h = f + g$, if $a \in \partial f(w)$, $b \in \partial g(w)$, then $a + b \in \partial h(w)$. Therefore, the calculation of $g_t = \partial f_t$ can be decomposed into two steps:

- $v_t \in \partial \ell_t(w_t) = \partial \ell(w_t, (x_{i_t}, y_{i_t}))$;
- $g_t = v_t + \lambda w_t$.

Then, updating w_t would become $w_{t+1} \leftarrow w_t - \frac{1}{\lambda t}(\lambda w_t + v_t) = \left(1 - \frac{1}{t}\right) w_t - \frac{1}{\lambda t} v_t$. Therefore, we have

$$\begin{aligned} \underbrace{t \cdot w_{t+1}}_{A_{t+1}} &= \underbrace{(t-1)w_t}_{A_t} - \frac{1}{\lambda} v_t \\ \implies A_{t+1} &= \sum_{s=1}^{t} -\frac{1}{\lambda} v_s \\ \implies w_{t+1} &= -\frac{1}{\lambda t} \sum_{s=1}^{t} v_s \end{aligned} \tag{6}$$

Now, coming back to the guarantee $\frac{\mathcal{R}_T(\Omega)}{T} = O\left(\frac{\ln T}{T}\right)$ shown in Theorem 1, let us calculate the constant factor inside $O\left(\frac{\ln T}{T}\right)$ for the unconstrained regularized loss minimization. Suppose additionally, that the unregularized loss ℓ_t's are B-Lipschitz w.r.t. $||\cdot||_2$, then we have $||v_t||_2 \leq B \implies ||w_t||_2 \leq \frac{B}{\lambda} \implies g_t = \lambda w_t + v_t$ satisfies $||g_t||_2 \leq 2B$ for every t. Therefore, applying Theorem 1, we have $\forall u \in \Omega$, $\mathcal{R}_T(u) \leq \frac{2B^2(\ln T + 1)}{\lambda}$. Then, from online-to-batch conversion, we have $\mathbb{E}F_S(\bar{w}_T) - F_S(\hat{w}) \leq \frac{2B^2(\ln T+1)}{\lambda T}$ and setting $T = \tilde{O}\left(\frac{B^2}{\lambda \epsilon}\right)$ ensures $\mathbb{E}F_S(\bar{w}_T) - F_S(\hat{w}) \leq \epsilon$.

3 Kernel Methods

In this section, we provide a brief introduction to kernel methods. Suppose all examples are transformed with a nonlinear map: $\phi : \mathbb{R}^d \to \mathbb{R}^N$ where N is large. The goal is to find w that approximately minimizes

$$\frac{1}{m} \sum_{i=1}^{m} f(y_i \langle w, \phi(x_i) \rangle) + \frac{\lambda}{2} ||w||_2^2. \tag{7}$$

However, as N is large, we would not like to maintain the iterates $w \in \mathbb{R}^N$ explicitly. To tackle this issue, a sliver lining is that $\langle \phi(x), \phi(z) \rangle = k(x, z)$, where $k(x, z)$ is called the kernel function induced by ϕ, can be

evaluated efficiently. For example, for the polynomial kernel $k(x, z) = (1 + \langle x, z \rangle)^l$, the ϕ inducing $k(x, z)$ will have $N = O(d^l)$ dimensions. However, we can evaluate $k(x, z)$ very efficiently as we only need to calculate $\langle x, z \rangle$ that takes $O(d)$ time and then take the power of l that takes $O(1)$ time. The driving question is then:

Can we develop efficient training and test algorithms with running time independent from N?

The key observation is that since the loss functions have the structures in Equation (7), if we take (sub)gradient of f w.r.t. w, the (sub)gradient will be always a scaling of some $\phi(x_i)$. Then, from Equation (6), we have that updating the iterates w_t will require calculating the linear combination of $\phi(x_{i_t})$. Therefore, instead of bookkeeping g_t and w_t, which are of high dimensions, we can keep track of coefficients $\alpha_t \in \mathbb{R}^m$ of w_t and maintain invariant that $w_t = \sum_{i=1}^m \alpha_t(i)\phi(x_i)$. Then, as $m \ll N$, the computation can be much more efficient.

After getting the final iterate $w = \sum_{i=1}^m \alpha(i)\phi(x_i)$, the question now becomes how we shall make prediction. It turns out that in the prediction stage, using the assumption that $\langle \phi(x), \phi(z) \rangle = k(x, z)$ we have $\langle w, \phi(x) \rangle = \sum_{i=1}^m \alpha_i \langle \phi(x_i), \phi(x) \rangle = \sum_{i=1}^m \alpha_i k(x_i, x)$, which can be done efficiently.

In the next lecture, we will discuss in detail how we can modify Algorithm 3 so that instead of bookkeeping w_t's, we keep track of α_t's to train the model and make prediction.