

Lecture 19

Gradient descent and its generalizations

Consider a cost function

$$E(\vec{\theta}) = \sum_{i=1}^n \ell_i(\vec{x}_i, \vec{\theta})$$

\leftarrow total # datapoints
 \uparrow input datapoint
 \uparrow fitting prms

Gradient descent (GD): $\vec{\theta}_t$

$$(1) \quad \vec{\theta}_{t+1} = \vec{\theta}_t - \eta_t \nabla_{\vec{\theta}} E(\vec{\theta}_t), \quad \text{where}$$

t is the step # & η_t is the learning rate (may be changing with t).

Finally, $(\nabla_{\vec{\theta}})_i = \frac{\partial}{\partial \theta_i}$.

Main drawback: η_t is a hyperparam whose value (or "schedule" if we wish to change it with t) is difficult to obtain a priori.

Compare GD with Newton-Raphson (NR):

$$(2) \quad \vec{\theta}_{t+1} = \vec{\theta}_t - \underbrace{H^{-1}(\vec{\theta}_t)}_{\vec{\eta}_t} \nabla_{\vec{\theta}} E(\vec{\theta}_t), \quad \text{where}$$

H is the Hessian (in practice, H is often regularized: $H^{-1} \Rightarrow (H + \frac{\epsilon}{2} \mathbb{I})^{-1}$, where $\frac{\epsilon}{2}$ is a small offset).

NR requires H (which is extremely expensive), but

its learning rate is set by H^{-1} instead of an ad hoc function (or constant) η_t .

In other words, NR automatically takes large steps in flat directions and smaller steps in directions with large curvature, whereas GD is not as adaptive.

Besides, computing $\vec{\nabla}_{\vec{\theta}} E(\vec{\theta}_t)$ at each step is expensive for large n .

[Various improvements]

① Stochastic gradient descent (SGD) with mini-batches

Idea: incorporate stochasticity into GD while improving performance by computing the gradient on a subset of data called a mini-batch.

Specifically, divide n datapoints randomly into M minibatches s.t.

$$\frac{n}{M} \approx 10^1 - 10^2.$$

↳

datapoints
in a minibatch

at each step, replace $\vec{\nabla}_{\vec{\theta}} E(\vec{\theta}_t)$ by

$$\sum_{i=1}^n \vec{\nabla}_{\vec{\theta}} \ell_i(\vec{x}_i, \vec{\theta}_t)$$

$$\vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_t) = \sum_{i \in B_k} \vec{\nabla}_{\vec{\theta}} \ell_i(\vec{x}_i, \vec{\theta}_t), \text{ where}$$

B_k is a set of all datapoints in minibatch k ($k=1, \dots, M$).

$$\text{Then } \vec{\theta}_{t+1} = \vec{\theta}_t - \eta_t \overbrace{\vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta})}^{\vec{v}_t} \quad (3)$$

One full cycle over all M batches is called an epoch.

② SGD with momentum (SGDM)

$$\begin{cases} \vec{v}_t = \gamma \vec{v}_{t-1} + \eta_t \vec{\nabla}_{\vec{\theta}}^{MB} E(\vec{\theta}_t), \\ \vec{\theta}_{t+1} = \vec{\theta}_t - \vec{v}_t, \text{ where} \end{cases} \quad (4)$$

$0 \leq \gamma \leq 1$ is a momentum param; for $\gamma=0$,
SGDM \rightarrow SGD.

Eqs. (4) can be rewritten as

$$\underbrace{\vec{\theta}_{t+1} - \vec{\theta}_t}_{\Delta \vec{\theta}_{t+1}} = -\gamma \underbrace{\vec{v}_{t-1}}_{\vec{\theta}_{t-1} - \vec{\theta}_t = -\Delta \vec{\theta}_t} - \eta_t \vec{\nabla}_{\vec{\theta}}^{MB} E(\vec{\theta}_t), \text{ or}$$

$$\Delta \vec{\theta}_{t+1} = \gamma \Delta \vec{\theta}_t - \eta_t \vec{\nabla}_{\vec{\theta}}^{MB} E^{MB}(\vec{\theta}_t) \quad (4')$$

clearly, $t=0$:

$$\left\{ \begin{aligned} \Delta \vec{\theta}_1 &= -\eta_0 \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_0), \quad \Leftarrow \Delta \vec{\theta}_0 = 0 \\ \Delta \vec{\theta}_2 &= \gamma \Delta \vec{\theta}_1 - \eta_1 \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_1) = \\ &= -\gamma \eta_0 \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_0) - \eta_1 \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_1), \\ \Delta \vec{\theta}_3 &= -\gamma^2 \eta_0 \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_0) - \gamma \eta_1 \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_1) - \\ &\quad - \eta_2 \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_2), \\ &\dots \end{aligned} \right.$$

Thus $\Delta \vec{\theta}_t$ is a running average over ~~the~~ previous gradients weighted by γ .

Empirically, the "inertia" term $\gamma \Delta \vec{\theta}_t$ which contributes to $\Delta \vec{\theta}_{t+1}$ along with the new gradient value (cf. (4')) helps "gain speed" in directions with persistent gradients, without jumping around too much.

a slight modification of SGDM:

$$\begin{cases} \vec{v}_t = \gamma \vec{v}_{t-1} + \eta_t \nabla_{\vec{\theta}} E^{MB}(\vec{\theta}_t \cdot \gamma \vec{v}_{t-1}), \\ \vec{\theta}_{t+1} = \vec{\theta}_t - \vec{v}_t. \end{cases} \quad (5)$$

$\nabla_{\vec{\theta}} E^{MB}$ is evaluated at

$\vec{\theta}_t \cdot \gamma \vec{v}_{t-1}$ instead of $\vec{\theta}_t$.

\vec{v}_t w/out the gradient term in (5)

$\vec{\theta}_{t+1}$ w/out the gradient term in (5)

3. Methods that use the 2nd moment of the gradient (RMS prop, ADAM)

SGD & SGDM still need η_t - a ~~drawback~~ major drawback. Ideally, we would need an exact or approximate H (Hessian), but this is too expensive.

RMS prop, ADAM and other algorithms try to adapt step sizes to the landscape without computing H .

RMS prop:

$$\left\{ \begin{array}{l} \bar{g}_t = \nabla_{\theta} E^{MB}(\bar{\theta}_t), \\ \bar{S}_{t,j} = \beta \bar{S}_{t-1,j} + (1-\beta) g_{t,j}^2, \\ \bar{\theta}_{t+1,j} = \bar{\theta}_{t,j} - \eta_t \frac{\bar{g}_{t,j}}{\sqrt{\bar{S}_{t,j} + \epsilon}} \end{array} \right. \quad (6)$$

$\sim 10^{-8}$ regularizer

Here, $0 \leq \beta \leq 1$ sets the scale of the running average for all elements of the \bar{S}_t vector.

Clearly, w/out $\frac{1}{\sqrt{\bar{S}_{t,j} + \epsilon}}$ in the update rule we recover basic SGD.

If $\bar{S}_{t,j}$ is large (meaning that $g_{t',j}^2$ was large for several previous steps), $\frac{\eta_t}{\sqrt{\bar{S}_{t,j} + \epsilon}}$ will be reduced and

the algorithm will use smaller step sizes.

If however $\bar{S}_{t,j}$ is small the step sizes will be large.

ADAM :

$$\begin{aligned}
 \vec{g}_t &= \vec{\nabla}_{\vec{\theta}} E^{MB}(\vec{\theta}_t), \\
 \vec{m}_t &= \beta_1 \vec{m}_{t-1} + (1-\beta_1) \vec{g}_t, \\
 S_{t,j} &= \beta_2 S_{t-1,j} + (1-\beta_2) g_{t,j}^2, \\
 \hat{\vec{m}}_t &= \frac{\vec{m}_t}{1-\beta_1^t}, \\
 \hat{S}_t &= \frac{\vec{S}_t}{1-\beta_2^t}, \\
 \vec{\theta}_{t+1} &= \vec{\theta}_t - \eta_t \frac{\hat{\vec{m}}_t}{\sqrt{\hat{S}_t + \epsilon}}
 \end{aligned}$$

(7)

↙ regularizer

Here, \vec{m}_t & \vec{S}_t are the usual running averages with the scales β_1 & β_2 , respectively. $\hat{\vec{m}}_t$ & \hat{S}_t turn running averages into "true" (i.e., unbiased) averages.

For simplicity, consider a single prm θ . Then $\underbrace{\sigma_t^2}_{\text{variance}} = \hat{S}_t - \underbrace{\hat{m}_t^2}_{\text{mean squared}}$, and

$$\Delta \theta_{t+1} = -\eta_t \frac{\hat{m}_t}{\sqrt{\sigma_t^2 + \hat{m}_t^2 + \epsilon}}$$

\mathcal{I}_0 $\underbrace{\sigma_t^2 \ll \hat{m}_t^2}_{\text{persistent gradient}} \ \& \ \hat{m}_t \gg \frac{\epsilon}{2}$, we obtain
 $\underbrace{\Delta\theta_{t+1} \rightarrow -\eta_t}_{\text{prm changes according to the maximum allowed step size, } \eta_t \text{ (i.e., the max allowed step is regularized)}}$

\mathcal{I}_0 $\sigma_t^2 \gg \hat{m}_t^2$, $\Delta\theta_{t+1} \rightarrow -\eta_t \underbrace{\frac{\hat{m}_t}{\sigma_t}}_{\text{signal-to-noise ratio}}$
 $\sigma_t^2 \gg \frac{\epsilon}{2}$

[Here, σ_t serves as the natural adaptive scale]

o
 [Unbiased averages:]

Consider a single prm for simplicity, s.t.

$$m_0 = 0,$$

$$m_1 = \beta_1 m_0 + (1 - \beta_1) g_1 = (1 - \beta_1) g_1,$$

$$m_2 = \beta_1 m_1 + (1 - \beta_1) g_2 = \beta_1 (1 - \beta_1) g_1 + (1 - \beta_1) g_2 = (1 - \beta_1) [g_2 + \beta_1 g_1],$$

$$m_3 = (1 - \beta_1) [g_3 + \beta_1 g_2 + \beta_1^2 g_1], \text{ etc.}$$

In general,

$$m_t = (1-\beta_1) [g_t + \beta_1 g_{t-1} + \beta_1^2 g_{t-2} + \dots + \beta_1^{t-1} g_1]$$

If $t = \infty$, the total weight of all the terms is

$$(1-\beta_1) [1 + \beta_1 + \beta_1^2 + \dots] = (1-\beta_1) \frac{1}{1-\beta_1} = 1,$$

as expected.

With finite t , the weight is

$$(1-\beta_1) [1 + \beta_1 + \dots + \beta_1^{t-1}] = (1-\beta_1) \frac{1-\beta_1^t}{1-\beta_1} =$$
$$= 1 - \beta_1^t < 1.$$

To remove this bias, we rescale the series for m_t by $\frac{1}{1-\beta_1^t}$:

$$m_t \rightarrow \frac{m_t}{1-\beta_1^t} = \hat{m}_t.$$

Same argument holds for S_t and for multiple fitting prms.