EECS 182 Deep Neural Networks
Spring 2023 Anant Sahai

Note 6

1 Momentum and Adaptive Gradient Descent Methods

Vanilla gradient descent has many benefits, but speed is not one of them. The reason behind that is the constraint on the learning rate. This effect can be illustrated by solving the simple scalar equation given in equation 1 which considers the squared loss as the loss function ($L(w)$). We want to minimize $L(w)$ over w until we get as close as possible to the ground truth (y).

$$\begin{aligned} \sigma w &= \hat{y} \\ \min_w L(w) &= (y - \sigma w)^2 \end{aligned} \tag{1}$$

Gradient descent step at k^{th} step is given in 2 with a learning rate (η).

$$\begin{aligned} w_{k+1} &= w_k - \eta \nabla_w L(w^k) \\ w_{k+1} &= w_k + 2\eta\sigma(y - \sigma w_k) \\ w_{k+1} &= (1 - 2\eta\sigma^2)w_k + 2\eta\sigma y \\ \left(w_{k+1} - \frac{y}{\sigma}\right) &= (1 - 2\eta\sigma^2)\left(w_k - \frac{y}{\sigma}\right) \end{aligned} \tag{2}$$

Then the w^{k+1} can be written in terms of the initial guess of w_0 by considering the pattern in equation 2, as below (3)

$$w_{k+1} = (1 - 2\eta\sigma^2)^{k+1}\left(w_0 - \frac{y}{\sigma}\right) + \frac{y}{\sigma} \tag{3}$$

In order to make sure the (3) is recurrence stable, $-1 < (1 - 2\eta\sigma^2) < 1$ this condition should be satisfied. From that, we get a constraint $\eta < \frac{1}{\sigma^2}$. Figure 1 shows the behavior of gradient descent update with the choice of η. As the η is small enough, the solution converges to the optimal solution but takes a lot of iterations. But when η passes some value, it shows an oscillatory behavior, and after increasing it further, it starts to diverge from the solution.

A slight modification called momentum for the gradient descent is applied to solve these problems.

1.1 Momentum

The idea of momentum is finding a safe way to make the η bigger. We know that the dimensions with larger singular values start to oscillate quickly. So the thought is to somehow low pass filter (LPF) those directions that would otherwise oscillate if we take smaller steps. By doing that, instead of moving in the direction

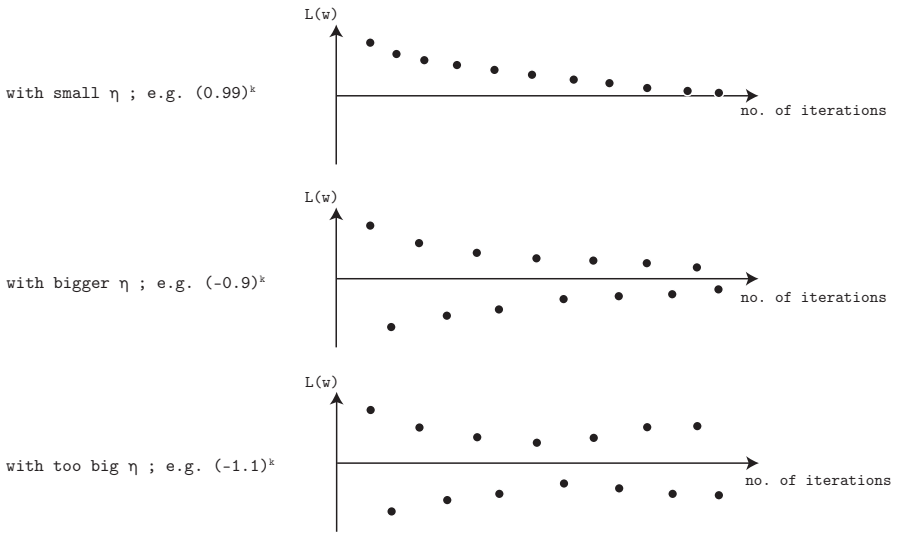

Figure 1: Gradient descent update with the choice of η

of the gradient, the update is moved towards the direction of the average gradient. The functioning of the simplest LPF is where this idea of averaging originally came from, Figure 2.

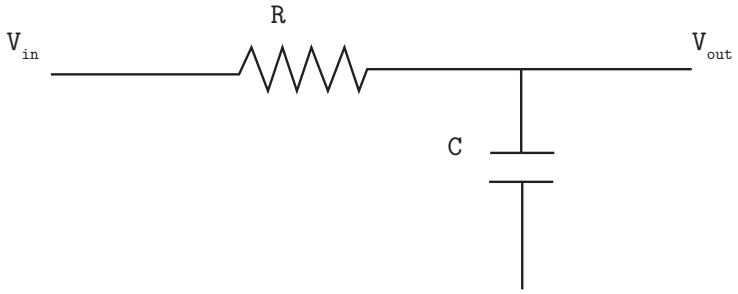

Figure 2: Simplest Low Pass Filter (LPF)

This low-pass filter (LPF) will regulate voltage. The discrete time first order differential-equation to achieve this would be:

Discrete-Time LPF Differential Equation

$$a_{t+1} = (1-\beta)a_t + \beta\mu_t$$
$$\text{Where } \mu_t \text{ is a time sequence and } \beta \in (0,1) \tag{4}$$

This average is beneficial, as it only requires one internal state to keep to track of, and β will determine how long averaging will go for. This technique is commonly referred to as exponential smoothing. Now we want to apply this technique to create a more stable gradient descent. Recall that the original gradient descent follows the formula:

Original Gradient Descent

$$\theta_{t+1} = \theta_t + \eta(-\nabla f_{\theta_t}) \tag{5}$$

In order to achieve gradient descent with momentum, we want to replace our gradient term in regular gradient descent with our low-pass filter equation applied to our gradient. Gradient descent with momentum can be considered as a “smooth gradient".

Gradient Descent with Momentum

$$\begin{aligned} \theta_{t+1} &= \theta_t + \eta(\alpha_{t+1}) \\ \alpha_{t+1} &= (1-\beta)\alpha_t + \beta(-\nabla f_{\theta_t}) \end{aligned} \tag{6}$$

The main difference with momentum is that gradients at previous time steps have an impact on the current gradient's value.

To understand how exactly did we arrive at 4: We can solve the first order differential equation that governs the LPF on 2 by equation 7 with a dummy variable τ. The exponential weighted average part is taken as $h(t-\tau)$, which is the impulse response that defines the filter. Since $h(t-\tau)$ is an exponentially weighted average of the V_{in}, the dying exponential has lesser weight as the τ gets further into the past.

$$\begin{aligned} V_{out}(t) &= \int_{-\infty}^{t} V_{in}(\tau)\frac{e^{-\frac{1}{RC}(t-\tau)}}{RC}d\tau \\ V_{out}(t) &= \int_{-\infty}^{t} V_{in}(\tau)h(t-\tau)d\tau \end{aligned} \tag{7}$$

This is also known as convolution integral as what it does is sliding along the input and taking averages by re-centering the input to t. And it integrates to 1 as it expresses a normalized average. In the discrete-time point of view, we can write 7 as a sum, equation 8. The geometrically dying exponential ($h(t-\tau)$ can be written with normalization parameter β as shown in 8. $\beta(1-\beta)^{t-\tau}$ makes sure that sums to 1.

$$V_{out}(t) = \sum_{-\infty}^{t} V_{in}(\tau)\beta(1-\beta)^{t-\tau} \tag{8}$$

This discrete time solution is also the solution to a first-order difference equation. Instead of input V_{in} we can plug in the gradients of $L(w)$ at current step, equation 9. β controls the averaging, as closer it gets to zero, the more averaging we get through the recurrence relationship of past gradients given by a_k.

$$a_{k+1} = (1-\beta)a_k + \beta\nabla_w L(w^k) \tag{9}$$

This momentum term a_{k+1} is used to update the $(k+1)^{st}$ gradient update rather than directly using the gradient as in vanilla gradient descent, equation 10.

$$w_{k+1} = w_k - \eta a_{k+1} \tag{10}$$

In this case, we have two hyper-parameters that can be adjusted, β & η. With the appropriate choice of β, we can increase the η than the limit imposed in vanilla gradient descent. The slower directions converge faster by allowing that increment, increasing the overall speedup.

The current gradient used in 9 can have two interpretations depending on which variant we want to use. The

two variants are "Vanilla momentum" and "Nesterov Momentum", equation 11.

$$\begin{aligned} &\text{"Vanilla" Momentum } \nabla L(w_t) \\ &\text{"Nesterov" Momentum } \nabla L(w_t - \eta(1-\beta)a_t) \end{aligned} \tag{11}$$

The vanilla momentum uses the gradient where the current weights are. In the Nesterov momentum, we take advantage of the fact where we are going by peeking into the future, as given from the first part of the momentum term (9). This is because we already know where we are going to end up, and by taking that new information to the calculation of the gradient at this step, we can take a bit of an advantage on learning.

1.2 Adaptive (e.g. Adam) Methods

The simple perspective of the origin of the adaptive methods comes from the fact that even with momentum, the largest singular value and the smallest singular value are the ones that govern the process. Values in between are not relevant for setting up the parameters. So the simple idea is that instead of having a single step size, use different step sizes (η_i) along different dimensions of the parameter vector. In vanilla gradient descent, we go down in the steepest gradient direction. But in the adaptive methods, we no longer go in the steepest direction but still in a downward direction.

The formulation of Adam's method where different step sizes in a different direction are shown in equation 12. In the large gradient directions, it takes smaller steps and vice versa. That way, it takes evenly sized steps along different directions. To track the size of the gradient, a vector $\mathbf{V}$ is introduced, which depends on the element-wise product of the gradient vector. Then the gradient update for the i^{th} parameter is computed with the square root of elements of $\mathbf{V}$ to make the units right. A constant (ϵ) is added to the denominator to make sure it is stable.

$$\begin{aligned} \mathbf{a}_{k+1} &= (1-\beta)\mathbf{a}_k + \beta \nabla_w L(w^k) \\ \mathbf{V}_{k+1} &= (1-\beta')\mathbf{V}_k + \beta' \nabla_w \begin{pmatrix} \cdots \\ \left(\frac{\partial L(w)}{\partial w_i}\right)^2 \\ \cdots \end{pmatrix} \\ w_{k+1}[i] &= w_k[i] - \eta \frac{a_{k+1}[i]}{\sqrt{V_{k+i}[i] + \epsilon}} \end{aligned} \tag{12}$$

Typically $\beta' << \beta$, to make sure the average size of the gradient is over a longer period than the average gradient. Otherwise, it would create oscillatory movement when it is closer to the convergence. This can be observed by the numerator and the denominator in the weight update step. Also, this creates another challenge when the β' is very small (which is typically the case). In the start, this creates problems as the $\mathbf{V}$ is going to be small for a while until it builds up. This is partially protected by the (ϵ). The solution for this problem is normalizing, as shown in equation 13. This normalization dies away as the iterations keep going but solve the problem at the start.

$$\begin{aligned}
\hat{\mathbf{a}}_{k+1} &= \frac{\mathbf{a}_{k+1}}{1-\beta^k} \\
\hat{\mathbf{V}}_{k+1} &= \frac{\mathbf{V}_{k+1}}{1-(\beta')^k} \\
w_{k+1}[i] &= w_k[i] - \eta \frac{\hat{\mathbf{a}}_{k+1}[i]}{\sqrt{\hat{\mathbf{V}}_{k+1}[i]} + \epsilon}
\end{aligned} \tag{13}$$

