Author：Liu Jian

Time：2021-03-17

1. SGD

记：

$$g(\theta) \triangleq \nabla_\theta \left(\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f(x^{(i)};\theta), y^{(i)}) \right)$$

随机梯度下降 ($t = 1, 2, \cdots$，下同)：

$$\theta^{t+1} = \theta^t - \varepsilon g(\theta^t)$$

1.1 动量法

1.1.1 Momentum

Require: Learning rate ε, momentum parameter α

Require: Initial parameter θ^1, initial velocity v^0

$$v^t = \alpha v^{t-1} - \varepsilon g(\theta^t)$$
$$\theta^{t+1} = \theta^t + v^t$$

Momentum 的另一种表达形式：

$$v^t = \alpha v^{t-1} - \varepsilon g(\theta^t) = \varepsilon \left(\frac{\alpha}{\varepsilon} v^{t-1} - g(\theta^t) \right) \triangleq \varepsilon u^t$$
$$\theta^{t+1} = \theta^t + \varepsilon u^t$$

接着寻找 u^t 和 u^{t-1} 之间的递推关系：

$$u^t = \frac{\alpha}{\varepsilon} v^{t-1} - g(\theta^t)$$
$$v^t = \varepsilon u^t \Rightarrow v^{t-1} = \varepsilon u^{t-1}$$
$$\Downarrow$$
$$u^t = \frac{\alpha}{\varepsilon} \varepsilon u^{t-1} - g(\theta^t) = \alpha u^{t-1} - g(\theta^t)$$

将u替换成v，整理得如下的等价迭代格式：

$$v^t = \alpha v^{t-1} - g(\theta^t)$$
$$\theta^{t+1} = \theta^t + \varepsilon v^t$$

这就是 CS231n 给出的格式。

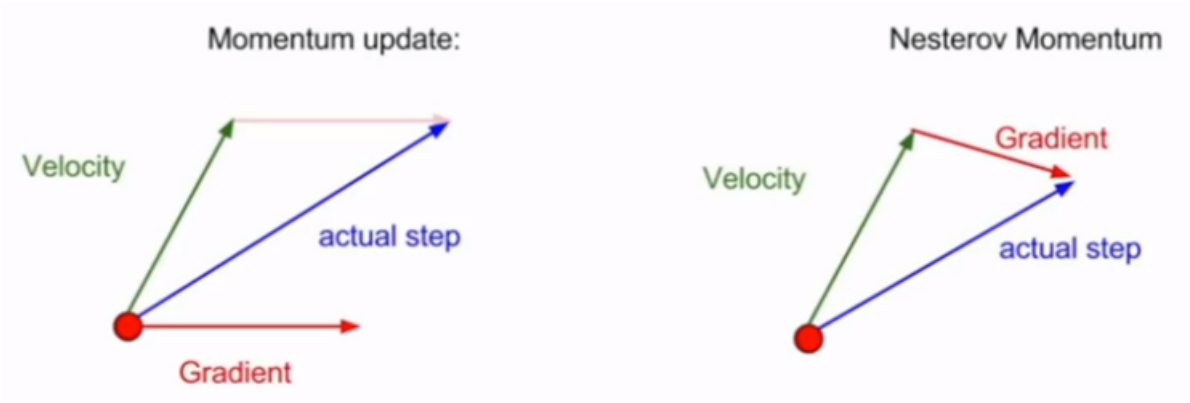
1.1.2 Nesterov Momentum

Require: Learning rate ε, momentum parameter α

Require: Initial parameter θ^1, initial velocity v^0

$$v^t = \alpha v^{t-1} - \varepsilon g(\theta^t + \alpha v^{t-1})$$
$$\theta^{t+1} = \theta^t + v^t$$

Nesterov Momentum 和普通的Momentum的区别：



1.2 权重自适应 (Adaptive Learning Rates)

1.2.1 AdaGrad

Require: Global learning rate ε, small contant δ (perhaps 10^{-7}) used for numerical stabilization

Require: Initial parameter θ^1, initial gradient accumulation $r^0 = \mathbf{0}$

$$r^t = r^{t-1} + g(\theta^t) \odot g(\theta^t) \ (\odot\text{: Multiplication applied element-wise})$$
$$\theta^{t+1} = \theta^t - \frac{\varepsilon}{\delta + \sqrt{r^t}} \odot g(\theta^t) \ (\text{Division and square root applied element-wise})$$

适用于凸优化问题。

1.2.2 RMSProp

The RMSProp algorithm modifies AdaGrad to perfrom better in the non-convex setting by changing the gradient accumulation r into an exponentially weighted moving average.

AdaGrad shrinks the learning rate according to the entire history of the squared gradient and may have made the learning rate too small before arriving at such a convex structure. RMSProp uses an exponentially decaying average to discard history from the extreme past so that it can converge rapidly after finding a convex bowl, as if it were an instance of the AdaGrad algorithm initialized within that bowl.

Require: Global learning rate ε, decay rate ρ controls the length scale of the moving average, small contant δ (usually 10^{-6}) used for numerical stabilization

Require: Initial parameter θ^1, initial gradient accumulation $r^0 = \mathbf{0}$

$$r^t = \rho r^{t-1} + (1-\rho) g(\theta^t) \odot g(\theta^t)$$

$$\theta^{t+1} = \theta^t - \frac{\varepsilon}{\sqrt{\delta + r^t}} \odot g(\theta^t) \ (\frac{1}{\sqrt{\delta + r^t}} \text{applied element-wise})$$

1.3 动量法 + 权重自适应

1.3.1 RMSProp with Nesterov momentum

Require: Global learning rate ε, decay rate ρ, momentum cofficient α

Require: Initial parameter θ^1, initial velocity v^0, initial gradient accumulation $r^0 = \mathbf{0}$

$$r^t = \rho r^{t-1} + (1-\rho) g(\theta^t + \alpha v^{t-1}) \odot g(\theta^t + \alpha v^{t-1})$$

$$v^t = \alpha v^{t-1} - \frac{\varepsilon}{\sqrt{r}} \odot g(\theta^t + \alpha v^{t-1}) \ (\frac{1}{\sqrt{r^t}} \text{applied element-wise})$$

$$\theta^{t+1} = \theta^t + v^t$$

1.3.2 Adam

Adam 算法十分健壮，是首选。

Require: Global learning rate ε (Suggested default: 0.001), decay rates ρ_1 and ρ_2 for moment estimates (Suggested default: 0.9 and 0.999 respectively), small contant δ used for numerical stabilization (Suggested default: 10^{-8})

Require: Initial parameter θ^1, initial 1st and 2nd moment variables $s^0 = \mathbf{0}, r^0 = \mathbf{0}$

$$\text{Biased 1st moment estimate: } s^t = \rho_1 s^{t-1} + (1-\rho_1) g(\theta^t)$$

$$\text{Biased 2nd moment estimate: } r^t = \rho_2 r^{t-1} + (1-\rho_2) g(\theta^t) \odot g(\theta^t)$$

$$\text{Corrected 1st moment estimate: } \hat{s}^t = \frac{s^t}{1-(\rho_1)^t}$$

$$\text{Corrected 2nd moment estimate: } \hat{r}^t = \frac{r^t}{1-(\rho_2)^t}$$

$$\theta^{t+1} = \theta^t - \frac{\varepsilon}{\delta + \sqrt{\hat{r}^t}} \hat{s}^t$$

其中，$(\rho_1)^t, (\rho_2)^t$ 表示 t 次方而不是第 t 次迭代。

Adam 名称来源于 Adaptive moments (自适应矩)，可视为 RMSProp 和 momentum 相结合的变体：

1. 上式中动量 (momentum) v 对应一阶矩 (moment) s，只不过前面我们没有明确地初始化动量 v^0，而这里 $s^0 = 0$；
2. 考虑到初始化时令 $s^0 = \mathbf{0}, r^0 = \mathbf{0}$，Adam 还包含对一阶矩和二阶矩的偏差修正，比如 $t = 1$ 时：

   $$s^1 = (1-\rho_1) g(\theta^1)$$

   $$r^1 = (1-\rho_2) g(\theta^1) \odot g(\theta^1)$$

   修正过后：

$$\hat{s}^1 = \frac{s^1}{1-\rho_1} = g(\theta^1)$$

$$\hat{r}^1 = \frac{r^1}{1-\rho_2} = g(\theta^1) \odot g(\theta^1)$$

而 RMSProp 虽然包含形式相同的二阶矩，且 $r^0 = \mathbf{0}$，但 RMSProp 未对其进行修正，因此，相比 Adam，训练初始阶段 RMSProp 对二阶矩的估计可能存在较大的偏差。

2. 二阶优化算法

2.1 牛顿法