串讲深度学习中的优化算法

分享人：Sm1les

目录
contents

基本框架

Datawhale

定义当前时刻待优化参数为$\theta_t \in \mathbb{R}^d$，损失函数为$J(\theta)$，学习率为$\eta$，参数更新框架为：

1. 计算损失函数关于当前参数的梯度：$g_t = \nabla J(\theta_t)$
2. 根据历史梯度计算一阶动量和二阶动量：

$$m_t = \phi(g_1, g_2, ..., g_t), V_t = \psi(g_1, g_2, ..., g_t)$$

3. 计算当前时刻的下降梯度：

$$\Delta\theta_t = -\eta \cdot \frac{m_t}{\sqrt{V_t}}$$

4. 根据下降梯度更新参数：$\theta_{t+1} = \theta_t + \Delta\theta_t$

SGD

SGD（Stochastic Gradient Descent）：由于SGD没有动量的概念，也即没有考虑历史梯度，所以当前时刻的一阶动量即为当前时刻的梯度$m_t = g_t$，且二阶动量$V_t = E$，所以SGD的参数更新公式为

$$\Delta\theta_t = -\eta \cdot \frac{g_t}{\sqrt{E}} = -\eta \cdot g_t$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t = \theta_t - \eta \cdot g_t$$

指数加权移动平均值（Exponentially Weighted Moving Average，EWMA）：假设v_{t-1}是$t-1$时刻的指数加权移动平均值，θ_t是t时刻的观测值，那么t时刻的指数加权移动平均值为

$$
\begin{aligned}
v_t &= \beta v_{t-1} + (1-\beta)\theta_t \\
&= (1-\beta)\theta_t + \sum_{i=1}^{t-1}(1-\beta)\beta^i\theta_{t-i}
\end{aligned}
$$

其中$0 \leq \beta < 1, v_0 = 0$。显然，由上式可知，$t$时刻的指数加权移动平均值其实可以看做前$t$时刻所有观测值的指数加权平均值，除了第$t$时刻的观测值权重为$1-\beta$外，其他时刻的观测值权重为$(1-\beta)\beta^i$。由于通常对于那些权重小于$\frac{1}{e}$的观测值可以忽略不计，所以忽略掉那些观测值以后，上式就可以看做在求指数加权**移动**平均值。

那么哪些项的权重会小于$\frac{1}{e}$呢？由于

$$\lim_{n\to+\infty}\left(1-\frac{1}{n}\right)^n=\frac{1}{e}\approx 0.3679$$

若令$n=\frac{1}{1-\beta}$，则

$$\lim_{n\to+\infty}\left(1-\frac{1}{n}\right)^n=\lim_{\beta\to 1}(\beta)^{\frac{1}{1-\beta}}=\frac{1}{e}\approx 0.3679$$

所以，当$\beta\to 1$时，那些$i\geq\frac{1}{1-\beta}$的θ_{t-i}的权重$(1-\beta)\beta^i$一定小于$\frac{1}{e}$。例如当$t=20,\beta=0.9$时，$\theta_1,\theta_2,..,\theta_9,\theta_{10}$的权重都是小于$\frac{1}{e}$的，因此可以忽略不计，那么此时就相当于在求$\theta_{11},\theta_{12},..,\theta_{19},\theta_{20}$这最近10个时刻的加权移动平均值。所以指数移动平均值可以近似看做在求最近$\frac{1}{1-\beta}$个时刻的加权移动平均值，β常取≥ 0.9。

由于当t较小时，指数加权移动平均值的偏差较大，例如：设$\theta_1 = 40, \beta = 0.9$，那么$v_1 = \beta v_0 + (1-\beta)\theta_1 = 0.9 * 0 + 0.1 * 40 = 4$，显然$v_1$和$\theta_1$相差太大，所以通常会加上一个修正因子$1 - \beta^t$，加了修正因子后的公式为

$$v_t = \frac{\beta v_{t-1} + (1-\beta)\theta_t}{1-\beta^t}$$

显然，当t很小时，修正因子$1 - \beta^t$会起作用，当t足够大时$\beta^t \to 0, (1-\beta^t) \to 1$，修正因子会自动退场。

Momentum

Momentum（SGD with Momentum）：为了抑制SGD的震荡，Momentum认为梯度下降过程可以加入**惯性**，也就是在SGD基础上引入了一阶动量。而所谓的一阶动量就是该时刻梯度的指数加权移动平均值：$\eta \cdot m_t := \beta \cdot m_{t-1} + \eta \cdot g_t$（其中$g_t$并不严格按照指数加权移动平均值的定义采用权重$1-\beta$，而是使用我们自定义的学习率$\eta$）。由于此时仍然没有用到二阶动量，所以$V_t = E$，那么Momentum的参数更新公式为

$$\Delta\theta_t = -\eta \cdot \frac{m_t}{\sqrt{E}} = -\eta \cdot m_t = -\left(\beta m_{t-1} + \eta g_t\right)$$

$$\theta_{t+1} = \theta_t - \left(\beta m_{t-1} + \eta g_t\right)$$

NAG

NAG（Nesterov Accelerated Gradient）：除了利用惯性跳出局部沟壑以外，我们还可以尝试往前看一步。想象一下你走到一个盆地，四周都是略高的小山，你觉得没有下坡的方向，那就只能待在这里了。可是如果你爬上高地，就会发现外面的世界还很广阔。因此，我们不能停留在当前位置去观察未来的方向，而要向前多看一步。我们知道Momentum在时刻t的主要下降方向是由历史梯度（惯性）决定的，当前时刻的梯度权重较小，那不如先看看如果跟着惯性走了一步，那个时候外面的世界是怎样的。也即在Momentum的基础上将当前时刻的梯度g_t换成下一时刻的梯度$\nabla J(\theta_t - \beta m_{t-1})$，由于此时仍然没有用到二阶动量，所以$V_t = E$，NAG的参数更新公式为

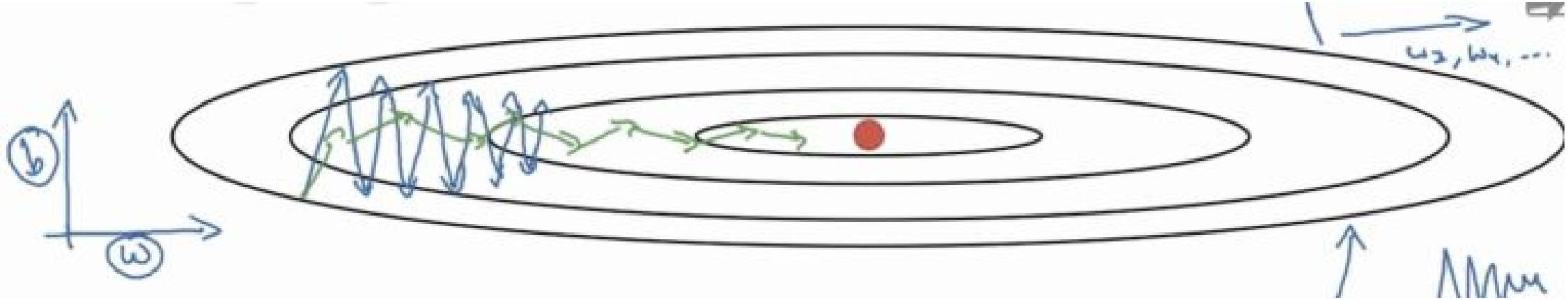
$$\Delta\theta_t = -\eta \cdot \frac{m_t}{\sqrt{E}} = -\eta \cdot m_t = -\left(\beta m_{t-1} + \eta \nabla J(\theta_t - \beta m_{t-1})\right)$$

$$\theta_{t+1} = \theta_t - \left(\beta m_{t-1} + \eta \nabla J(\theta_t - \beta m_{t-1})\right)$$

AdaGrad

Datawhale

此前我们都没有用到二阶动量。二阶动量的出现，才意味着“自适应学习率”优化算法时代的到来。SGD及其变种以同样的学习率更新每个维度的参数（因为θ_t通常是向量），但深度神经网络往往包含大量的参数，这些参数并不是总会用得到。对于经常更新的参数，我们已经积累了大量关于它的知识，不希望被单个样本影响太大，希望学习速率慢一些；对于偶尔更新的参数，我们了解的信息太少，希望能从每个偶然出现的样本身上多学一些，即学习速率大一些。因此，AdaGrad则考虑对于不同维度的参数采用不同的学习率。



AdaGrad

具体地，对于那些更新幅度很大的参数，通常历史累计梯度的平方和会很大，相反的，对于那些更新幅度很小的参数，通常其累计历史梯度的平方和会很小。所以在一个固定学习率的基础上除以历史累计梯度的平方和就能使得那些更新幅度很大的参数的学习率变小，同样也能使得那些更新幅度很小的参数学习率变大，所以AdaGrad的参数更新公式为

$$v_{t,i} = \sum_{t=1}^{t} g_{t,i}^2$$

$$\Delta\theta_{t,i} = -\frac{\eta}{\sqrt{v_{t,i} + \epsilon}} g_{t,i}$$

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{v_{t,i} + \epsilon}} g_{t,i}$$

其中$g_{t,i}^2$表示第t时刻第i维度参数的梯度值，ϵ是防止分母等于0的平滑项（常取一个很小的值$1e-8$）。显然，此时上式中的$\frac{\eta}{\sqrt{v_{t,i}+\epsilon}}$这个整体可以看做是学习率，分母中的历史累计梯度值$v_{t,i}$越大的参数学习率越小。

上式仅仅是第t时刻第i维度参数的更新公式，对于第t时刻的所有维度参数的整体更新公式为

$$V_t = \operatorname{diag}\left(v_{t,1}, v_{t,2}, \ldots, v_{t,d}\right) \in \mathbb{R}^{d \times d}$$

$$\Delta\theta_t = -\frac{\eta}{\sqrt{V_t + \epsilon}} \cdot g_t$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \cdot g_t$$

注意，由于V_t是对角矩阵，所以上式中的ϵ只用来平滑V_t对角线上的元素。

缺点：随着时间步的拉长，历史累计梯度平方和$v_{t,i}$会越来越大，这样会使得所有维度参数的学习率都不断减小（单调递减），无论更新幅度如何。

RMSProp/AdaDelta

由于AdaGrad单调递减的学习率变化过于激进，我们考虑一个改变二阶动量计算方法的策略：不累积全部历史梯度，而只关注过去一段时间窗口的下降梯度，采用Momentum中的指数加权移动平均值的思路。这也就是AdaDelta名称中Delta的来历。首先看最简单直接版的RMSProp，RMSProp就是在AdaGrad的基础上将普通的历史累计梯度平方和换成历史累计梯度平方和的指数加权移动平均值，所以只需将AdaGrad中的$v_{t,i}$的公式改成指数加权移动平均值的形式即可，也即

$$v_{t,i} = \beta v_{t-1,i} + (1-\beta) g_{t,i}^2$$

$$V_t = \operatorname{diag}\left(v_{t,1}, v_{t,2}, ..., v_{t,d}\right) \in R^{d \times d}$$

$$\Delta \theta_t = -\frac{\eta}{\sqrt{V_t + \epsilon}} \cdot g_t$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \cdot g_t$$

而AdaDelta除了对二阶动量计算指数加权移动平均以外，还对当前时刻的下降梯度$\Delta\theta_t$的**平方**也计算一个指数加权移动平均，具体地

$$\mathrm{E}[\Delta\theta^2]_{t,i} = \gamma\,\mathrm{E}[\Delta\theta^2]_{t-1,i} + (1-\gamma)\Delta\theta_{t,i}^2$$

由于$\Delta\theta_{t,i}^2$目前是未知的，所以只能用$t-1$时刻的指数加权移动平均来近似替换，也即

$$\mathrm{E}[\Delta\theta^2]_{t-1,i} = \gamma\,\mathrm{E}[\Delta\theta^2]_{t-2,i} + (1-\gamma)\Delta\theta_{t-1,i}^2$$

除了计算出$t-1$时刻的指数加权移动平均以外，AdaDelta还用此值替换我们预先设置的学习率η

因此，AdaDelta的参数更新公式为

$$v_{t,i} = \beta v_{t-1,i} + (1-\beta) g_{t,i}^2$$

$$V_t = \mathrm{diag}\left(v_{t,1}, v_{t,2}, ..., v_{t,d}\right) \in R^{d \times d}$$

$$\mathrm{E}[\Delta\theta^2]_{t-1,i} = \gamma\, \mathrm{E}[\Delta\theta^2]_{t-2,i} + (1-\gamma)\Delta\theta_{t-1,i}^2$$

$$\Theta_t = \mathrm{diag}\left(\mathrm{E}[\Delta\theta^2]_{t-1,1}, \mathrm{E}[\Delta\theta^2]_{t-1,2}, ..., \mathrm{E}[\Delta\theta^2]_{t-1,d}\right) \in R^{d \times d}$$

$$\Delta\theta_t = -\frac{\sqrt{\Theta_t + \epsilon}}{\sqrt{V_t + \epsilon}} \cdot g_t$$

$$\theta_{t+1} = \theta_t - \frac{\sqrt{\Theta_t + \epsilon}}{\sqrt{V_t + \epsilon}} \cdot g_t$$

显然，对于AdaDelta算法来说，已经不需要我们自己预设学习率η了，只需要预设β和γ这两个指数加权移动平均值的衰减率即可。

谈到这里，Adam和Nadam的出现就很自然而然了——它们是前述方法的集大成者。我们看到，Momentum在SGD基础上增加了一阶动量，AdaGrad在SGD基础上增加了二阶动量。把一阶动量和二阶动量都用起来，就是Adam了。

Adam

Datawhale

具体地，首先计算一阶动量：$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$

然后类似RMSProp/AdaDelta计算二阶动量

$$v_{t,i} = \beta_2 v_{t-1,i} + (1-\beta_2) g_{t,i}^2$$

$$V_t = \mathrm{diag}\left(v_{t,1}, v_{t,2}, \ldots, v_{t,d}\right) \in R^{d \times d}$$

然后分别加上指数加权移动平均值的修正因子

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

$$\hat{v}_{t,i} = \frac{v_{t,i}}{1-\beta_2^t}$$

$$\hat{V}_t = \mathrm{diag}\left(\hat{v}_{t,1}, \hat{v}_{t,2}, \ldots, \hat{v}_{t,d}\right) \in R^{d \times d}$$

所以，Adam的参数更新公式为

$$\Delta\theta_t = -\frac{\eta}{\sqrt{\hat{V}_t + \epsilon}} \cdot \hat{m}_t$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \epsilon}} \cdot \hat{m}_t$$

Nadam

由于Adam没有将Nesterov集成进来，而Nadam则是在Adam的基础上将Nesterov集成了进来，也即Nadam = Nesterov + Adam。具体思想如下：由于Nesterov的核心在于，计算当前时刻的梯度g_t时使用了「未来梯度」$\nabla J(\theta_t - \beta m_{t-1})$，NAdam 基于此提出了一种公式变形的思路，大意可以这样理解：只要能在梯度计算中考虑到「未来因素」，就算是达到了Nesterov的效果。既然如此，我们就不一定非要在计算g_t时使用「未来因素」，可以考虑在其他地方使用「未来因素」。

Nadam

具体地，首先NAdam在Adam的基础上将$\hat{m}_t$展开

$$\begin{aligned}\theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \epsilon}} \cdot \hat{m}_t \\ &= \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \epsilon}} \cdot \left(\frac{\beta_1 m_{t-1}}{1-\beta_1^t} + \frac{(1-\beta_1) g_t}{1-\beta_1^t}\right)\end{aligned}$$

此时，如果我们将第$t-1$时刻的动量m_{t-1}用第t时刻的动量m_t近似代替的话，那么我们就引入了「未来因素」，所以将m_{t-1}替换成m_t即可得到Nadam的表达式

$$\begin{aligned}\theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \epsilon}} \cdot \left(\frac{\beta_1 m_t}{1-\beta_1^t} + \frac{(1-\beta_1) g_t}{1-\beta_1^t}\right) \\ &= \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \epsilon}} \cdot \left(\beta_1 \hat{m}_t + \frac{(1-\beta_1) g_t}{1-\beta_1^t}\right)\end{aligned}$$

参考文献

https://ruder.io/optimizing-gradient-descent/index.html
https://zhuanlan.zhihu.com/p/32230623
https://zhuanlan.zhihu.com/p/32335746
https://zhuanlan.zhihu.com/p/29920135
https://zhuanlan.zhihu.com/p/32626442

Datawhale

一个专注于AI领域的开源组织