问题

深度学习中有很多优化函数，常见的那些你还记得它的定义以及优缺点吗？

背景知识

深度学习网络训练中，有很多可供选择的优化函数如SGD、Adam等等，到底用哪个好呢？其实这个问题没有确切的答案，优化函数是需要配合损失函数使用的，说白了，优化函数也是一种超参数，是需要尝试的，哪个效果好就用哪个……

这些优化函数其实差别不大，都是基于一个基本框架来演进的，所以下面先介绍下优化算法的基本框架：

1、优化算法基本框架

（记住这个框架！！！）

假设当前时刻待优化的参数为 θ_t ，损失函数为 $J(\theta)$ ，学习率为 η ，参数更新的框架为：

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：

$$\begin{aligned} m_t &= \phi(g_1, g_2, \ldots, g_t) \\ V_t &= \psi(g_1, g_2, \ldots, g_t) \end{aligned} \tag{1}$$

   即一阶动量为包含当前梯度在内的历史梯度的一次函数，而二阶动量是历史梯度的二次函数。

3. 计算当前时刻的下降梯度：

$$\Delta\theta_t = -\eta * \frac{m_t}{\sqrt{V_t}} \tag{2}$$

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

2、指数加权移动平均值

SGD只计算当前梯度更新参数，完全没有考虑历史梯度，但这样有一个问题是假如当前参数处在损失函数的局部最低点处，即梯度为0，因为梯度为0，所以参数不再更新，也就是说不管你之前历史梯度多大，下降地多快，只要你当前梯度为0，那就只能停在这里，也就意味着冲不出这个局部最低点。要解决这个问题就需要将历史梯度考虑进来，但是这里又有一个问题：历史梯度那么多，全部都考虑吗，还是只考虑一部分？其实我们只要考虑最近的一段历史梯度即可，这段历史梯度怎么截就用到了**指数加权移动平均值**的概念。

假设 υ_{t-1} 是 $t-1$ 时刻的指数加权移动平均值，θ_t 是当前 t 时刻的观测值，那么 t 时刻的指数加权移动平均值为：

$$\begin{aligned} \upsilon_t &= \beta\upsilon_{t-1} + (1-\beta)\theta_t \\ &= (1-\beta)\theta_t + (1-\beta)\beta\theta_{t-1} + \beta^2\upsilon_{t-2} \\ &= (1-\beta)\theta_t + (1-\beta)\beta\theta_{t-1} + (1-\beta)\beta^2\theta_{t-2} + \beta^3\upsilon_{t-3} \\ &\ldots(\text{递推}) \\ &= (1-\beta)\theta_t + \sum_{i=1}^{t-1}(1-\beta)\beta^i\theta_{t-i} \end{aligned} \tag{3}$$

其中 $0 \leq \beta \leq 1$ ，从指数加权移动平均值的最终形式可以看出，i 表示的是距离当前时刻的时间长短，i 越大代表着距离当前时刻越久远，且由于其系数中指数部分的影响，其系数越小，也就是说距离当前时刻越远的历史梯度对当前时刻的指数加权移动平均值的贡献越少，这时候若我们设置一个阈值来对贡献量进行筛选，便使得当前时刻的指数加权移动平均值只考虑距离当前时刻较近的那些历史梯度，这就对应了名字中的"移动"这个概念。

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，所以忽略掉那些权重小于这个阈值的观测值之后，上式就可以看做是在求指数加权**移动**平均值。

下面我们计算一下什么时候权重 $(1-\beta)\beta^i$ 等于$\frac{1}{e}$ 的。

高数中有一个重要极限公式：

$$\lim_{n\to\infty}(1+\frac{1}{n})^n = e \tag{4}$$

其实这个极限无论是对于 $+\infty$还是 $-\infty$都是成立的，因此我们令 $t=-n$，得：

$$lim_{t\to-\infty}(1-\frac{1}{t})^{-t} = e \quad \rightarrow \quad lim_{n\to\infty}(1-\frac{1}{n})^n = \frac{1}{e} \approx 0.3679$$

令 $n = \frac{1}{1-\beta}$，则：

$$lim_{n\to\infty}(1-\frac{1}{n})^n = lim_{\beta\to1}(\beta)^{\frac{1}{1-\beta}} = \frac{1}{e}$$

所以当 $\beta \to 1$ 时，那些 $i \geq \frac{1}{1-\beta}$ 的 θ_{t-i} 的权重 $(1-\beta)\beta^i$ 的权重肯定小于 $\frac{1}{e}$ 。β **通常取0.9，也就是说 $i \geq 10$ 的那些观测值都会被忽略，也就相当于只考虑包括当前时刻在内的最近10个时刻的指数加权移动平均值。**

但是还有一个问题是：当 t 比较小时，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 θ_1 相差太大，所以通常会加上一个修正因子 $1-\beta^t$ ，加上修正因子后的公式为：

$$v_t = \frac{\beta v_{t-1} + (1-\beta)\theta_t}{1-\beta^t} \tag{5}$$

显然，当 t 较小时，修正因子 $1-\beta^t$ 会起作用，当 t 足够大后，$\beta^t \to 0, (1-\beta^t) \to 1$ ，修正因子自动退场。**加修正因子的这个做法只有在 Adam 和 Nadam 中使用到，其他算法并没有考虑。**

SGD (Stochastic Gradient Descent)

SGD不考虑历史梯度，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：

$$\begin{aligned} \Delta\theta_t &= -\eta * \frac{g_t}{\sqrt{E}} = -\eta * g_t \\ \theta_{t+1} &= \theta_t + \Delta\theta_t = \theta_t - \eta * g_t \end{aligned} \tag{6}$$

缺点：容易陷入局部最优。由于SGD只考虑当前时刻的梯度，在局部最优点的当前时刻梯度为 0 ，根据上面计算公式可知，此时参数不再进行更新，故陷入局部最优的状态。

但是虽然SGD有陷入局部最优的缺陷，但还是很常用。我的理解是：以上分析是针对一个参数 θ_i 来说的，即使其中一个参数陷入局部最优，但其他参数还是会继续更新，所以大概率会将陷入局部最优的那个参数拖离局部最优点，于是该参数可以继续更新。 所以整体来说并不会像单个参数那样陷入局部最优就出不来，所以还是可以work的。

改进策略及算法

1. **引入历史梯度的一阶动量**，代表算法：Momentum、NAG
2. **引入历史梯度的二阶动量**，代表算法：AdaGrad、RMSProp、AdaDelta
3. **同时引入历史梯度的一阶动量及二阶动量**，代表算法：Adam、Nadam

Momentum

为了抑制SGD的震荡（有点不理解这句话），Momentum认为梯度下降过程可以加入惯性，也就是在SGD的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，而由于此时仍然没有用到二阶动量，所以 $V_t = E$ ，所以Momentum的参数更新公式如下：

$$
\begin{aligned}
&m_t = \beta m_{t-1} + \eta g_t \\
&\Delta\theta_t = -\eta * \frac{m_t}{\sqrt{E}} = -\eta * m_t = -(\beta m_{t-1} + \eta g_t) \\
&(这里m_t乘以\eta后可以视为不变，因为乘上后，系数同样是大于0小于1的) \\
&\theta_{t+1} = \theta_t - (\beta m_{t-1} + \eta g_t)
\end{aligned} \tag{7}
$$

可以看到上面式子的**第一行 g_t 前面的系数并不是严格按照我们上面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 η** ，这点需要注意。

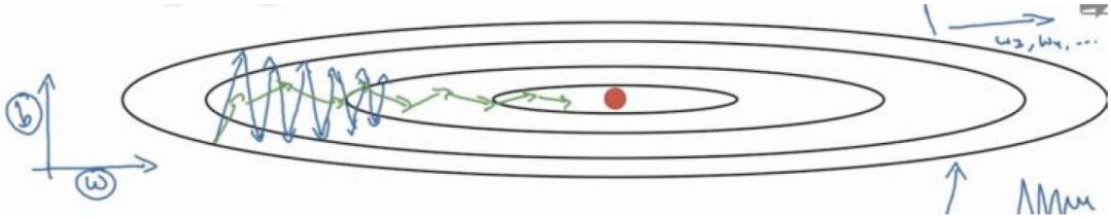
NAG (Nesterov Accelerated Gradient)

除了利用惯性（一阶动量）跳出局部沟壑外，我们还可以尝试往前看一步，即：在Momentum考虑历史梯度的基础上，把当前梯度转换为未来梯度。

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，那不如不用管当前梯度，而是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换成下一时刻的梯度。由于此时仍然没有用到二阶动量，所以 $V_t = E$ ，所以NAG的参数更新公式为：

$$
\begin{aligned}
&Momentum中原本下一个时刻的梯度计算公式为：\theta_{t+1} = \theta_t - (\beta m_{t-1} + \eta g_t) \\
&不考虑当前梯度即令；g_t = 0 \\
&所以下一个时刻的梯度的计算公式为：\quad \theta_{t+1} = \theta_t - \beta m_{t-1} \\
&所以将当前时刻的梯度换成下一时刻的梯度即：g_t = \Delta J(\theta_t - \beta m_{t-1}) \\
&上式代入到Momentum的参数更新公式中：\theta_{t+1} = \theta_t - (\beta m_{t-1} + \eta \Delta J(\theta_t - \beta m_{t-1}))
\end{aligned} \tag{8}
$$

> 以上的两个概念只引入了一阶动量。而二阶动量的出现，才意味着**“自适应学习率”**优化算法时代的到来。在SGD及其引入一阶动量的改进算法中，均以相同的学习率去更新参数。但是，以相同的学习率进行变化经常是不合理的。
>
> 在神经网络中，参数需要用不同的学习率进行更新。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；对于那些偶尔更新的参数，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。
>
> 以神经网络中的 W 及 b 为例，如下图为损失函数等高线图，W 为横轴，b 为纵轴。发现每次b 变化很大，而 W 每次仅更新一小步。但是，纵观整个损失函数我们发现，W 其实可以迈开步子往前走，而 b 则不用那么活跃。



于是，出现了以下针对不同维度的参数采用不同学习率的二阶动量改进算法。

AdaGrad

从数学的角度来看，更新幅度很大的参数，通常其历史累计梯度的平方和会很大；相反，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。因此，我们可以考虑让学习率除以历史梯度的平方和，这样之前更新幅度大的参数的学习率分母也大，之前更新幅度小的参数的学习率分母也小，从而起到调节学习率的效果。

我们上面讨论的 θ_t 指的是网络中的参数，但是参数有很多个，所以其实 θ_t 是一个向量，我们假设网络中有 d 个参数，那么 $\theta_t = [\theta_{t,1}, \theta_{t,2}, .. \theta_{t,d}]^T$ 。那么针对其中的第 i 维度的参数梯度更新公式为：

$$
\begin{aligned}
v_{t,i} &= \sum_{t=1}^{t} g_{t,i}^2 \\
\Delta\theta_{t,i} &= -\frac{\eta}{\sqrt{v_{t,i} + \varepsilon}} * g_{t,i} \\
\theta_{t+1,i} &= \theta_{t,i} - \frac{\eta}{\sqrt{v_{t,i} + \varepsilon}} * g_{t,i}
\end{aligned} \tag{9}
$$

其中 $g_{t,i}$ 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（常取一个很小的值例如 1e-8） 。显然，此时上式中的 $-\frac{\eta}{\sqrt{v_{t,i}+\varepsilon}}$ 这个整体可以看作是学习率，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 时刻所有维度的参数的更新公式如下：

$$
\begin{aligned}
V_t &= diag(v_{t,1}, v_{t,2}, \ldots, v_{t,d}) \in R^{d*d} \\
\Delta\theta_t &= -\frac{\eta}{\sqrt{V_t + \varepsilon}} * g_t \\
\theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{V_t + \varepsilon}} * g_t
\end{aligned} \tag{10}
$$

也就是构造成矩阵相乘的形式： V_t 是对角矩阵（除了对角线有非零值外其他地方都是 0 ） 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：不累计全部历史梯度，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。

首先看最简单直接版的 **RMAProp** ，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：

$$
\begin{aligned}
v_{t,i} &= \beta v_{t-1,i} + (1-\beta) g_{t,i}^2 \\
V_t &= diag(v_{t,1}, v_{t,2}, \dots, v_{t,d}) \in R^{d*d} \\
\Delta\theta_t &= -\frac{\eta}{\sqrt{V_t + \varepsilon}} * g_t \\
\theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{V_t + \varepsilon}} * g_t
\end{aligned} \tag{11}
$$

而 **AdaDelta** 又在 RMSProp 的基础上进行改进：它除了对二阶动量计算指数加权移动平均值之外，还对学习率动了手脚，即要达到的目标是不需要我们人为设定固定的学习率，而是让模型根据历史经验将学习率给换掉。所以另外它会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，具体的：

$$
E[\Delta\theta^2]_{t,i} = \gamma E[\Delta\theta^2]_{t-1,i} + (1-\gamma)\Delta\theta_{t,i}^2 \tag{12}
$$

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值来近似替换，也即：

$$
E[\Delta\theta^2]_{t-1,i} = \gamma E[\Delta\theta^2]_{t-2,i} + (1-\gamma)\Delta\theta_{t-1,i}^2 \tag{13}
$$

接着 AdaDelta 将此值替换我们预先设置的学习率 η。

因此，AdaDelta 的参数更新公式如下：

$$
\begin{aligned}
v_{t,i} &= \beta v_{t-1,i} + (1-\beta) g_{t,i}^2 \\
V_t &= diag(v_{t,1}, v_{t,2}, \dots, v_{t,d}) \in R^{d*d} \\
E[\Delta\theta^2]_{t-1,i} &= \gamma E[\Delta\theta^2]_{t-2,i} + (1-\gamma)\Delta\theta_{t-1,i}^2 \\
\Theta_t &= diag(E[\Delta\theta^2]_{t-1,1}, E[\Delta\theta^2]_{t-1,2}, \dots, E[\Delta\theta^2]_{t-1,d}) \in R^{d*d} \\
\Delta\theta_t &= -\frac{\sqrt{\Theta_t + \varepsilon}}{\sqrt{V_t + \varepsilon}} * g_t \\
\theta_{t+1} &= \theta_t - \frac{\sqrt{\Theta_t + \varepsilon}}{\sqrt{V_t + \varepsilon}} * g_t
\end{aligned} \tag{14}
$$

显然，对于 AdaDelta 算法来说，已经不需要我们自己预设学习率了，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。

> 下面的两个算法对SGD的改进策略是同时引入一阶动量和二阶动量。Adam和Nadam都是前述方法的集大成者。

Adam

在 RMSProp 的基础上再考虑一阶动量(Momentum)。具体如下：

首先计算一阶动量：（注意这个公式中 g_t 前面的系数与Momentum是不同的）

$$
m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{15}
$$

然后类似 RMSProp 和 AdaDelta 计算二阶动量：

$$
\begin{aligned}
v_{t,i} &= \beta_2 v_{t-1,i} + (1-\beta_2) g_{t,i}^2 \\
V_t &= diag(v_{t,1}, v_{t,2}, \dots, v_{t,d}) \in R^{d*d}
\end{aligned} \tag{16}
$$

但是这里要加上修正因子，即：（如果忘了这个回到前面去再看看指数加权移动平均值概念的最后部分）

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{17}$$
$$\hat{v}_{t,i} = \frac{v_{t,i}}{1-\beta_2^t}$$
$$\hat{V}_t = diag(\hat{v}_{t,1}, \hat{v}_{t,2}, \ldots, \hat{v}_{t,d}) \in R^{d*d}$$

所以，Adam的参数更新公式为：

$$\Delta\theta_t = -\frac{\eta}{\sqrt{\hat{V}_t + \varepsilon}} * \hat{m}_t \tag{18}$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \varepsilon}} * \hat{m}_t$$

Nadam

从这名字也能看出，Nadam = Nestrov + 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" $\Delta J(\theta_t - \beta m_{t-1})$ ，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 "未来梯度" ，就算达到了 Nestrov 的效果。既然如此，我们不一定非要在计算 g_t 时使用 "未来梯度" ，可以考虑在其他地方使用未来梯度。

具体的，首先在 Adam 的基础上将 $\hat{m}_t$ 展开：

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \varepsilon}} * \hat{m}_t \tag{19}$$
$$= \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \varepsilon}} * (\frac{\beta m_{t-1}}{1-\beta_1^t} + \frac{(1-\beta_1)g_t}{1-\beta_1^t})$$

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" ，所以便可以得到 Nadam 的表达式为：

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{V}_t + \varepsilon}} * (\frac{\beta m_t}{1-\beta_1^t} + \frac{(1-\beta_1)g_t}{1-\beta_1^t}) \tag{20}$$

参考资料

1、深度学习中的优化算法串讲
2、以上资料的视频讲解

By Yee

2020.05.14