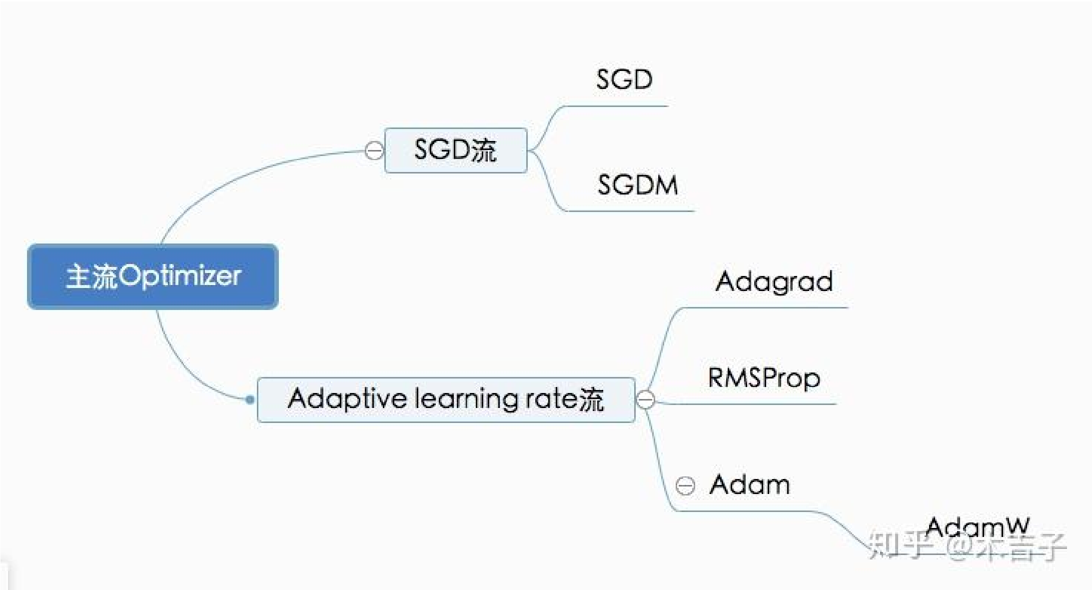
优化函数



GD 梯度下降

遍历全部训练集，太慢

BGD 批量梯度下降

一次更新一个 batch 的数据，速度较快，但是不一定是全局最优，最终结果在最优解附近。

SGD 随机梯度下降

随机抽取一部分数据来更新，有可能到迭代完成也没有完全遍历完样本。速度较快，但是不一定是全局最优，最终结果在最优解附近。

对于后面的优化函数，我的理解是向量有方向和大小两个指标，Momentum 解决了方向的问题。AdaGrad，RMSProp 解决大小问题。SGD 是一阶导，Adagrad 是二阶导数，但是二阶导数不好求，所以用其它求法替代。Adam 结合两者。打个比方，一阶导数是速度，二阶导数是加速度。

Momentum

在 SGD 的基础上引入了一阶动量： $m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$

一阶动量是各个时刻梯度方向的指数移动平均值。也就是说，t 时刻的下降方向，不仅由当前点的梯度方向决定，而且由此前累积的下降方向决定。β1 的经验值为 0.9，这就意味着下降方向主要是此前累积的下降方向，并略微偏向当前时刻的下降方向。想象高速公路上汽车转弯，在高速向前的同时略微偏向，急转弯可是要出事的。

Adagrad

$$\eta^t = \frac{\eta}{\sqrt{t+1}} \qquad g^t = \frac{\partial L(\theta^t)}{\partial w} \qquad w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t$$

怎么样去度量历史更新频率呢？那就是二阶动量——该维度上,迄今为止所有梯度值的平方和。一般为了避免分母为 0，会在分母上加一个小的平滑项。参数更新越频繁，二阶动量越大,学习率就越小。分母是在模拟二次微分,因为计算二次微分在实际问题中是开销很大的。

$$w^1 \leftarrow w^0 - \frac{\eta^0}{\sigma^0} g^0 \qquad \sigma^0 = \sqrt{(g^0)^2}$$

$$w^2 \leftarrow w^1 - \frac{\eta^1}{\sigma^1} g^1 \qquad \sigma^1 = \sqrt{\frac{1}{2}[(g^0)^2 + (g^1)^2]}$$

$$w^3 \leftarrow w^2 - \frac{\eta^2}{\sigma^2} g^2 \qquad \sigma^2 = \sqrt{\frac{1}{3}[(g^0)^2 + (g^1)^2 + (g^2)^2]}$$

$$\vdots$$

$$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t \qquad \sigma^t = \sqrt{\frac{1}{t+1}\sum_{i=0}^{t}(g^i)^2}$$

$$w^{t+1} \leftarrow w^t - \frac{\boxed{\eta^t}}{\boxed{\sigma^t}} g^t$$

$$\eta^t = \frac{\eta}{\cancel{\sqrt{t+1}}} \quad \text{1/t decay}$$

$$\sigma^t = \sqrt{\frac{1}{\cancel{t+1}}\sum_{i=0}^{t}(g^i)^2}$$

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^{t}(g^i)^2}} g^t$$

RMSprop

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。

$$\begin{aligned} s_{dw} &= \beta s_{dw} + (1-\beta) dW^2 \\ s_{db} &= \beta s_{db} + (1-\beta) db^2 \\ W &= W - \alpha \frac{dW}{\sqrt{s_{dw}+\varepsilon}} \\ b &= b - \alpha \frac{db}{\sqrt{s_{db}+\varepsilon}} \end{aligned}$$

Adam

既要考虑方向，又要考虑大小。

在训练的最开始我们需要初始化梯度的累积量和平方累积量。

$$v_{dw} = 0, v_{db} = 0; s_{dw} = 0, s_{db} = 0$$

$$\begin{aligned} v_{dw} &= \beta_1 v_{dw} + (1-\beta_1) dW \\ v_{db} &= \beta_1 v_{db} + (1-\beta_1) db \\ s_{dw} &= \beta_2 s_{dw} + (1-\beta_2) dW^2 \\ s_{db} &= \beta_2 s_{db} + (1-\beta_2) db^2 \end{aligned} \qquad \begin{aligned} v^c_{dw} &= \frac{v_{dw}}{1-\beta_1^t} \\ v^c_{db} &= \frac{v_{db}}{1-\beta_1^t} \\ s^c_{dw} &= \frac{s_{dw}}{1-\beta_2^t} \\ s^c_{db} &= \frac{s_{db}}{1-\beta_2^t} \end{aligned} \qquad \begin{aligned} W &= W - \alpha \frac{v^c_{dw}}{\sqrt{s^c_{dw}+\varepsilon}} \\ b &= b - \alpha \frac{v^c_{db}}{\sqrt{s^c_{db}+\varepsilon}} \end{aligned}$$

Batch size 的选择

Batch size 太大，对于优化器来说，会造成陷入较差的 local optimization，影响最终性

能。过小会导致下降过程剧烈波动。

实际使用的选择

SGD+momentum 一般能达到最好的效果，Adam 和 AdamW 之类的优化器收敛效果非常快，但是最终的结果往往会比 SGD+Momentum 差一些。可以前期用 Adam，后期换成 SGD，这样既保证了速度又保证了精度。Adam 达不到最优的原因是因为后期衰减太快了。