各种优化器的区别

笔记本:	深度学习		
创建时间:	2018/8/11 14:23	**更新时间:**	2018/8/11 15:22
作者:	beyourselfwb@163.com		
URL:	https://www.jianshu.com/p/d99b83f4c1a6		

作者：不会停的蜗牛
链接：https://www.jianshu.com/p/d99b83f4c1a6
來源：简书

keras optimizers.py 里的优化器

```
all_classes = {
    'sgd': SGD,
    'rmsprop': RMSprop,
    'adagrad': Adagrad,
    'adadelta': Adadelta,
    'adam': Adam,
    'adamax': Adamax,
    'nadam': Nadam,
    'tfoptimizer': TFOptimizer,
}
```

这么多优化器
trade - off　权衡 参数更新的准确率和运行时间 两者的关系

6. Adagrad

这个算法就可以对低频的参数做较大的更新，对高频的做较小的更新，也因此，对于稀疏的数据它的表现很好，很好地提高了 SGD 的鲁棒性，例如识别 Youtube 视频里面的猫，训练 GloVe word embeddings，因为它们都是需要在低频的特征上有更大的更新。**Adagrad 的优点是减少了学习率的手动调节**

梯度更新规则:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \cdot g_{t,i}.$$

缺点:

它的缺点是分母会不断积累，这样学习率就会收缩并最终会变得非常小。

7. Adadelta

这个算法是对 Adagrad 的改进，和 Adagrad 相比，就是分母的 G 换成了过去的梯度平方的衰减平均值，此外，还将学习率 η 换成了 RMS[Δθ]，这样的话，我们甚至都不需要提前设定学习率了.

梯度更新规则:

此外，还将学习率 η 换成了 RMS[Δθ]，这样的话，我们甚至都不需要提前设定学习率了:

$$\Delta\theta_t = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} g_t.$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t.$$

7. RMSprop

RMSprop 是 Geoff Hinton 提出的一种自适应学习率方法。

RMSprop 和 Adadelta 都是为了解决 Adagrad 学习率急剧下降问题的，

8. Adam

这个算法是另一种计算每个参数的自适应学习率的方法。

除了像 Adadelta 和 RMSprop 一样存储了过去梯度的平方 vt 的指数衰减平均值，也像 momentum 一样保持了过去梯度 mt 的指数衰减平均值:

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

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2.$$

如果 mt 和 vt 被初始化为 0 向量，那它们就会向 0 偏置，所以做了偏差校正，

通过计算偏差校正后的 mt 和 vt 来抵消这些偏差:

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

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}.$$

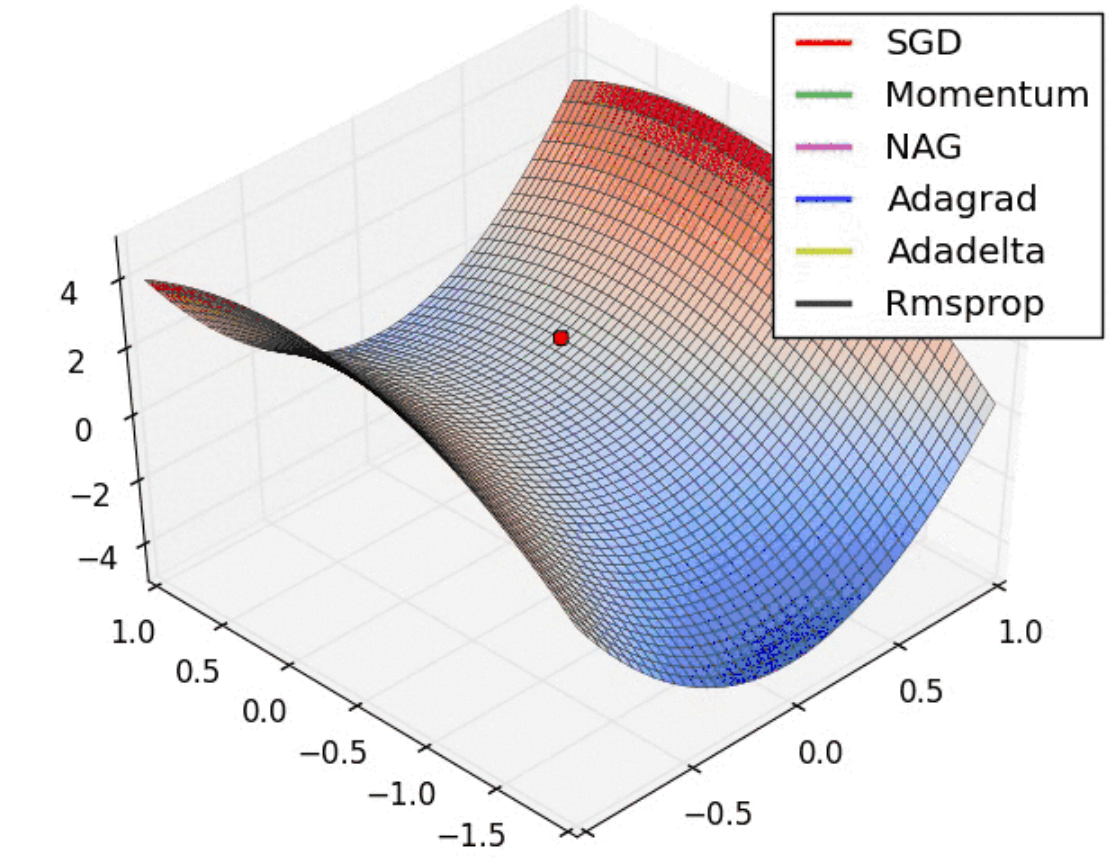
梯度更新规则:

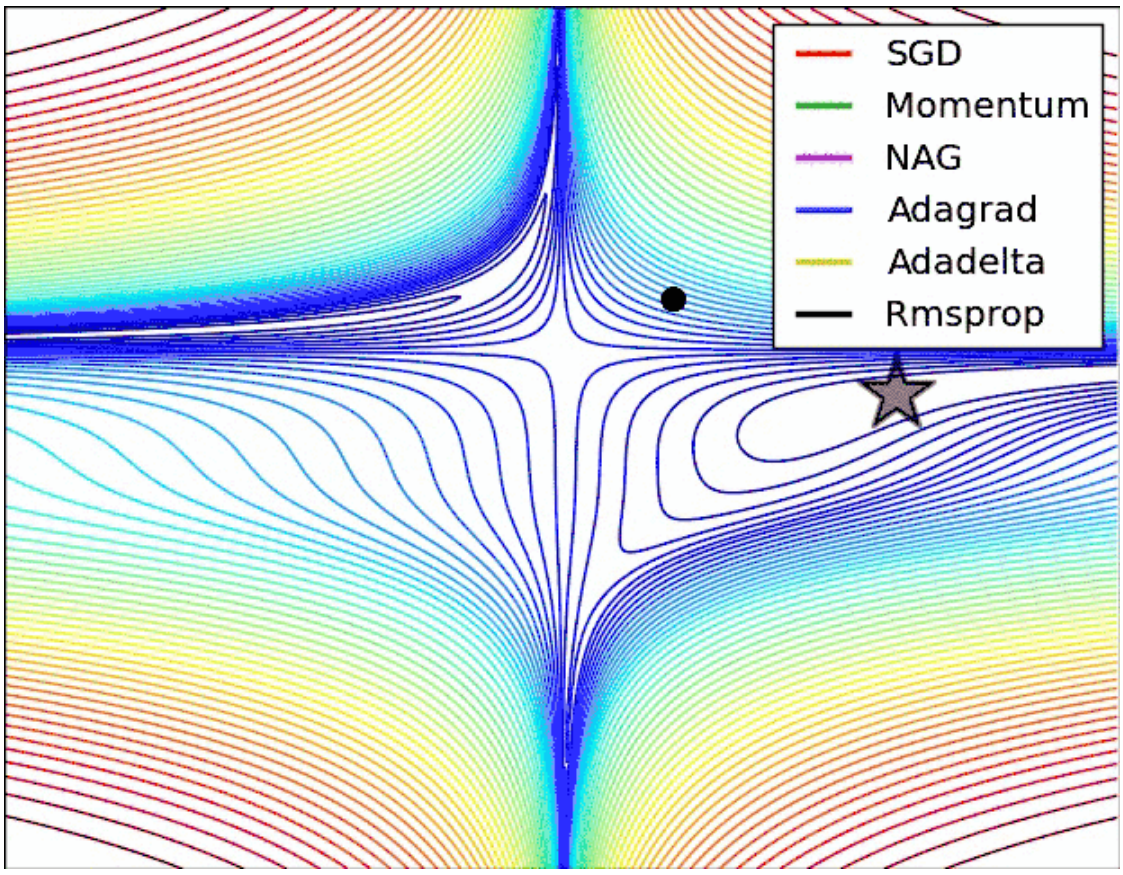
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t.$$

超参数设定值:

建议 β1 = 0.9，β2 = 0.999，ϵ = 10e−8

实践表明，Adam 比其他适应性学习方法效果要好。


SGD
Momentum
NAG
Adagrad
Adadelta
Rmsprop
4
2
0
−2
−4
1.0
0.5
0.0
−0.5
−1.0
−1.5
1.0
0.5
0.0
−0.5


SGD
Momentum
NAG
Adagrad
Adadelta
Rmsprop

整体来讲，Adam 是最好的选择。