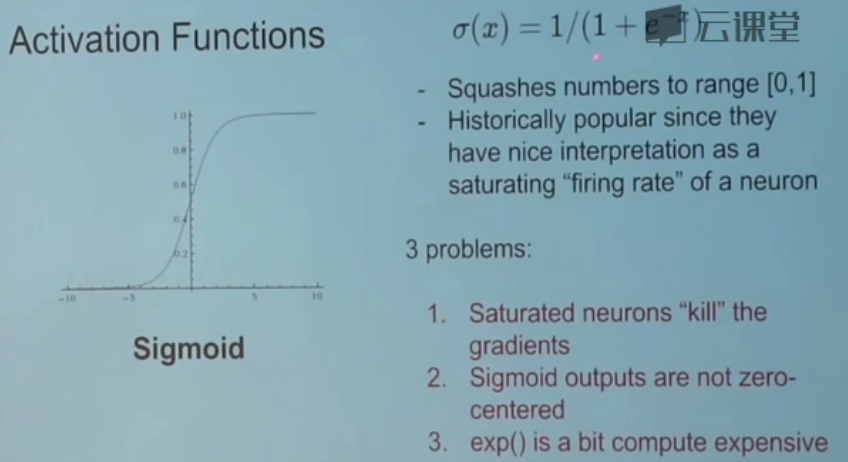
# 神经网络的训练

**激活函数**

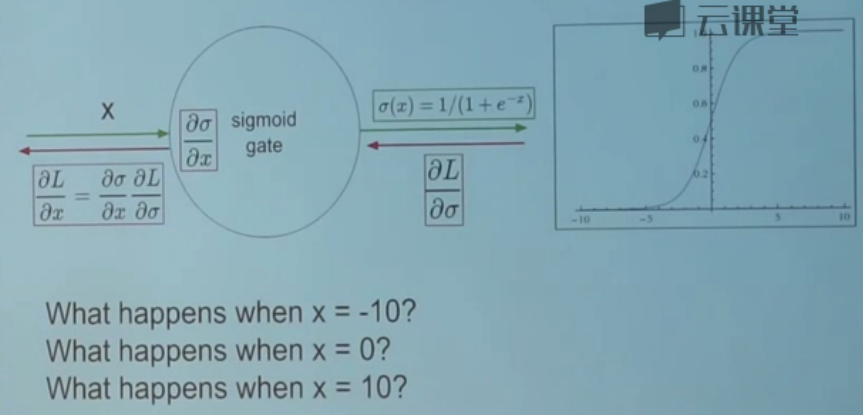
一、Sigmoid



三个问题：

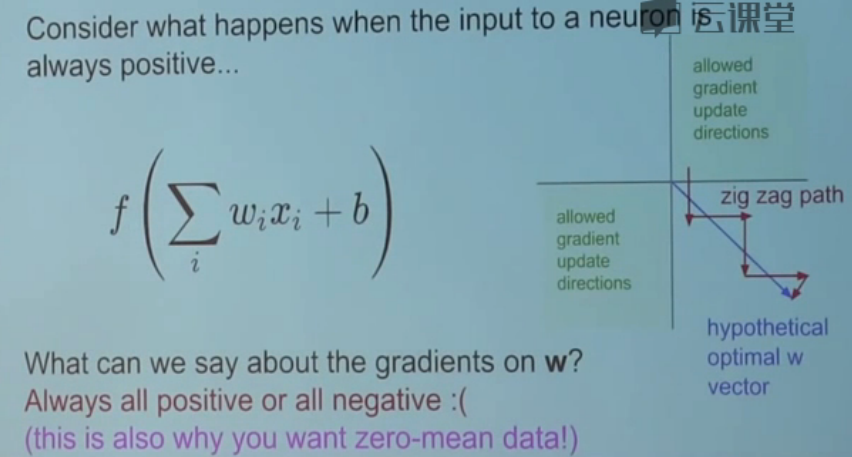
1. 梯度消失

在安全区以外的值，sigmoid值接近0/1时，斜率趋于0，因此梯度也趋于0



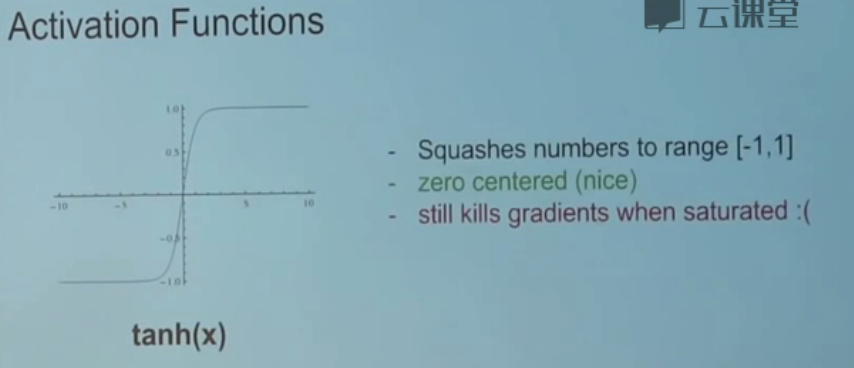
1. 不是原点中心对称的

因此w的梯度要么全是正数，要么全是负数，也会限制收敛的速度

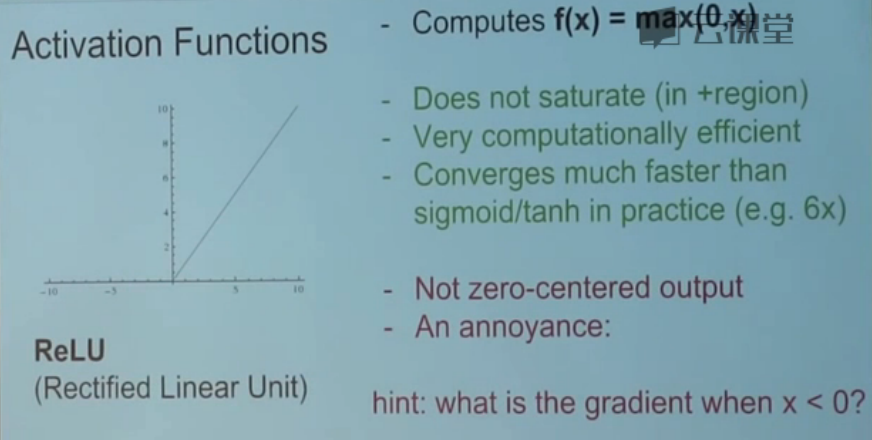


1. 计算复杂度高

二、tanh



三、ReLU



优点：

1. 在正半区，不会出现梯度消失
2. 收敛速度快，计算简单（6倍）

确定：

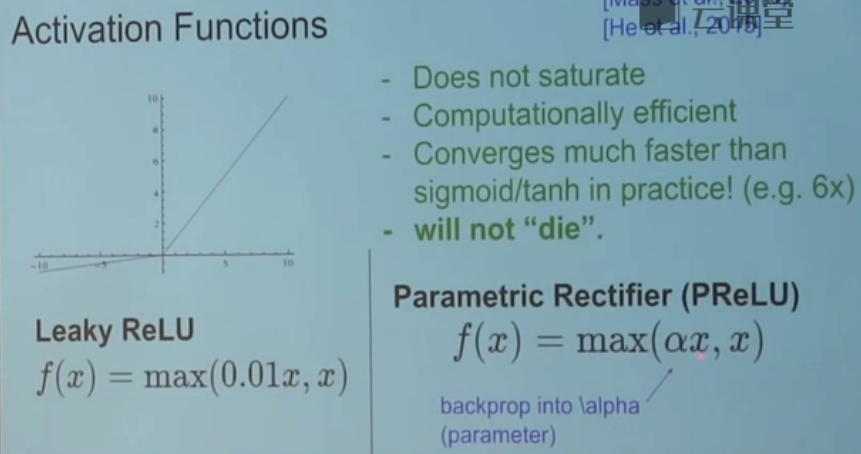
1. 非中心对称
2. 当输入为负数时，梯度=0，因此不会进行反向传播

Note:

学习率过高时，可能导致出现dead神经元

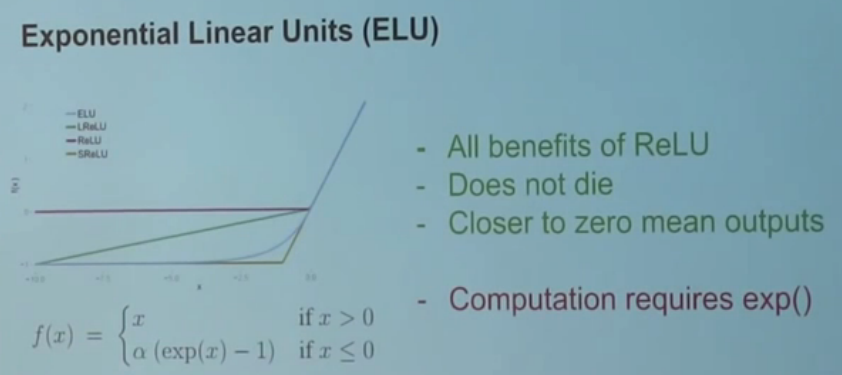
四、Leaky ReLU

解决梯度=0的问题



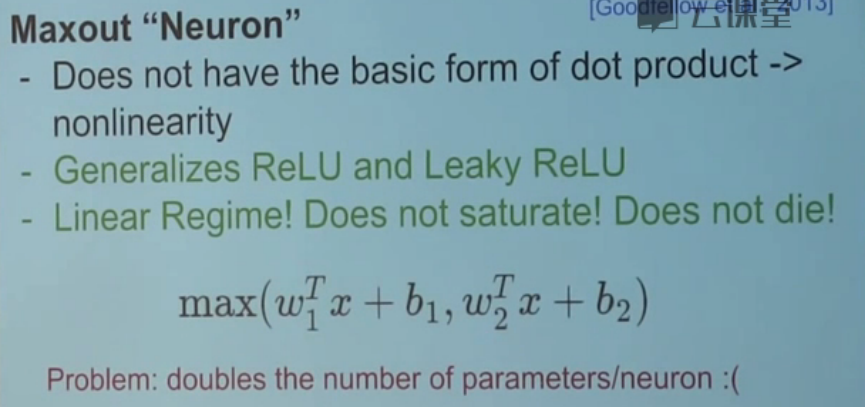
五、ELU

解决非0均值输出问题

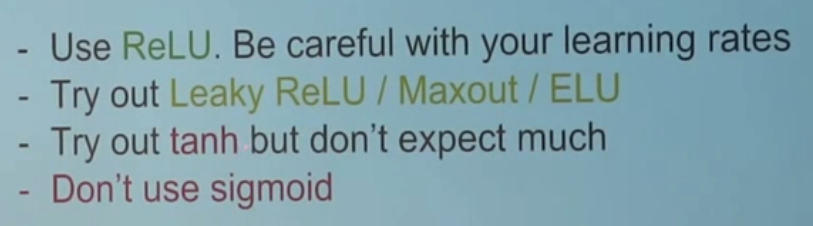


六、Maxout

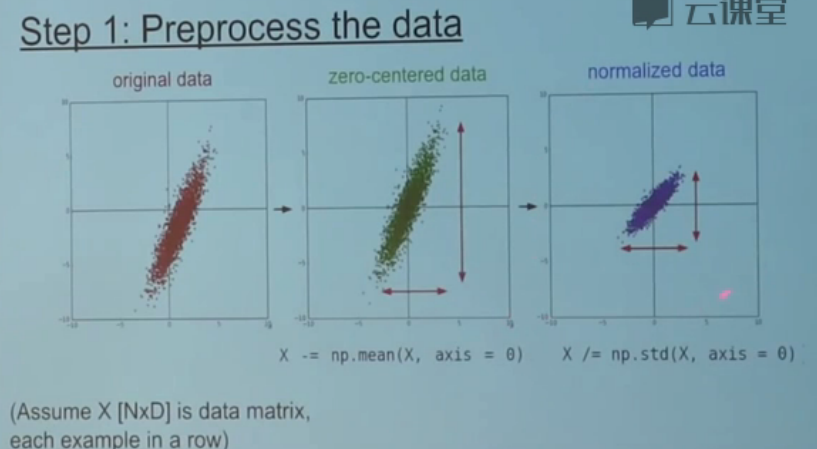
两套参数，取max

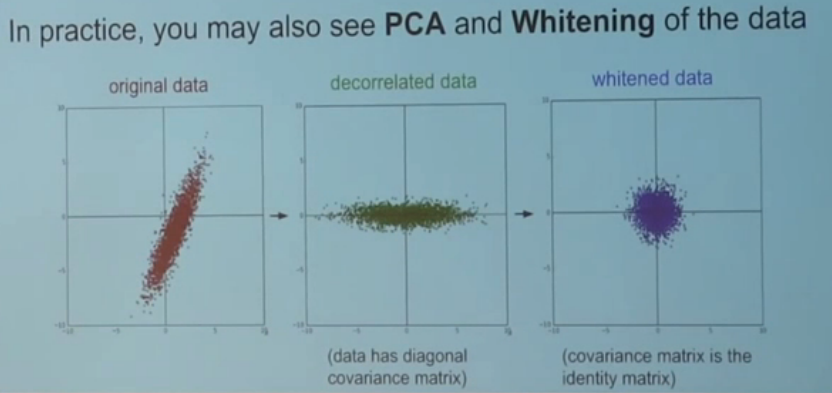


Summary：



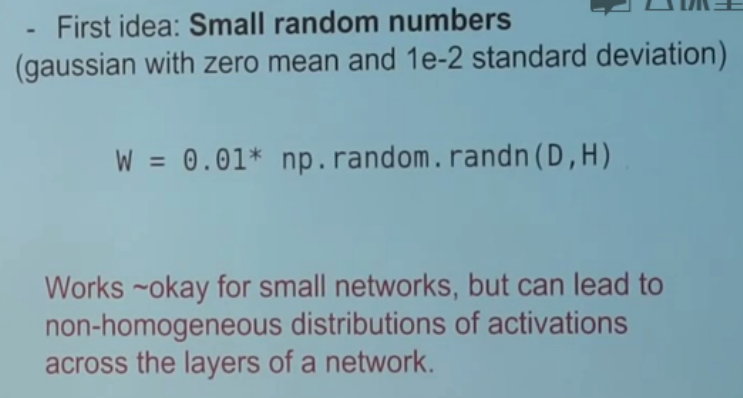
**预处理**





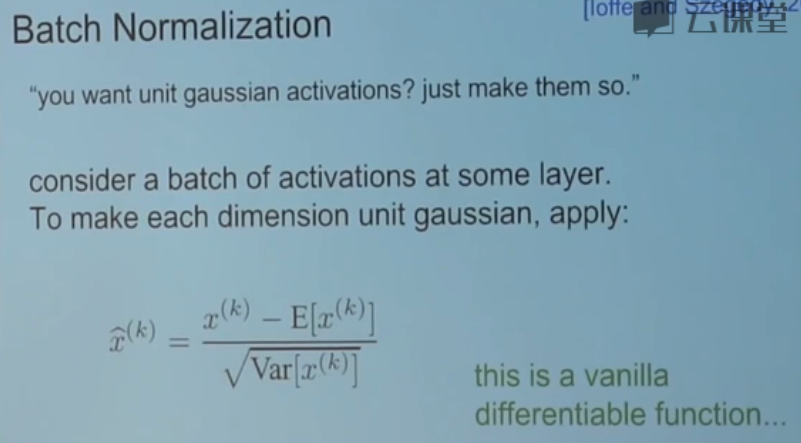
**权值初始化**

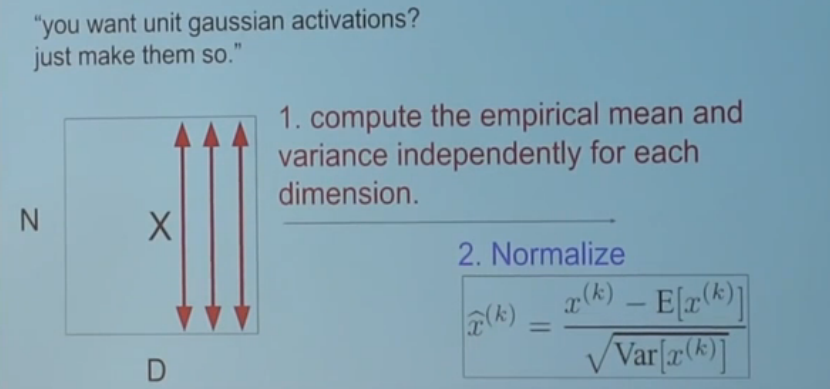
小随机数（如高斯噪声）

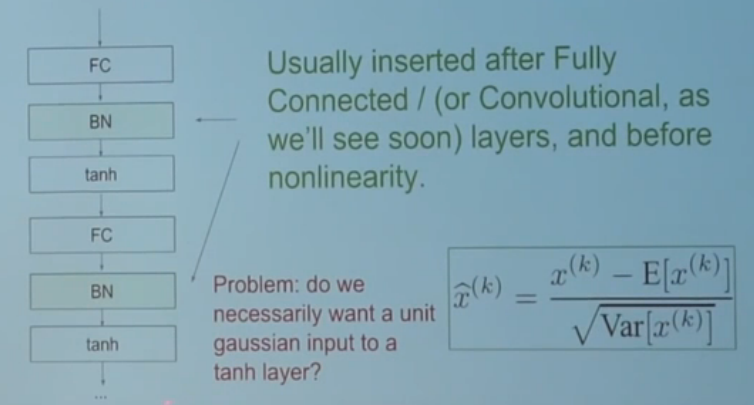


Note：高斯噪声的标准差太小或者太大都会导致出现问题（随着层数的增加），理想的权值应该是服从高斯分布的。权值初始化工作本质上是一项数据驱动型工作，应该根据不同的数据，不同的激活函数，进行不同的调整。

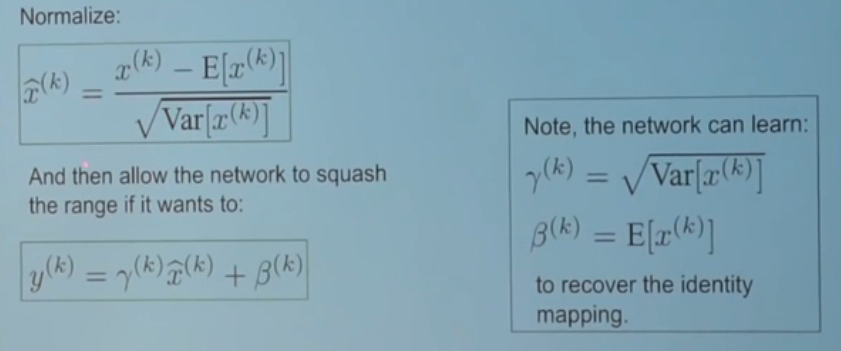
**批量数据正则化**



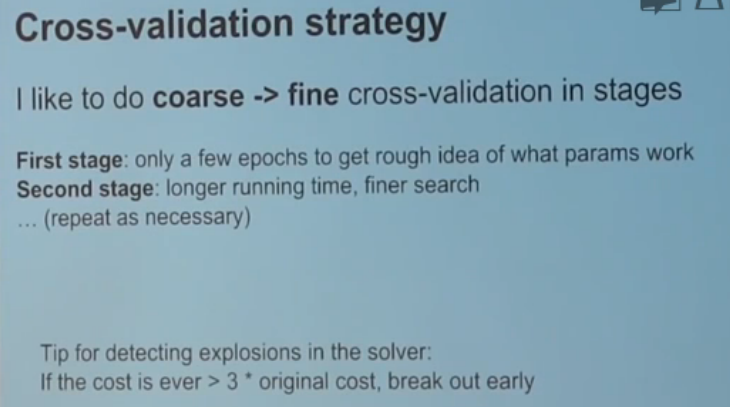




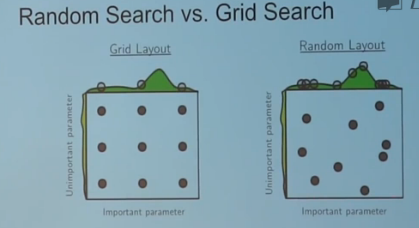
当不想要BN层影响后面的tanh层时，可以加上下面两个参数（参数如果为右侧，可以完全恢复）



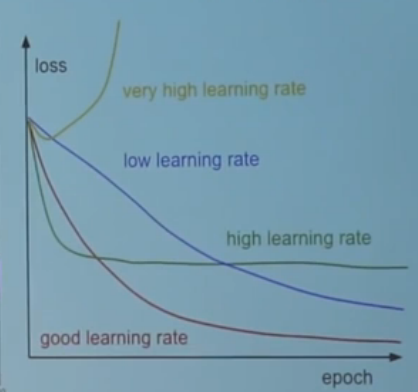
参数优化时，先粗粒度确定大致范围（数据量小、速度快），然后再在一个小范围内进行细粒度搜索

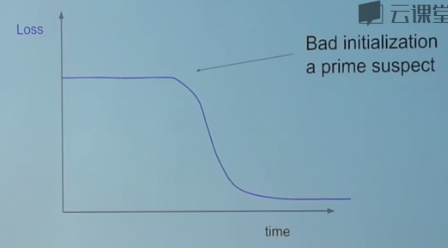


有时候，随机搜索可能比网格搜索来的好

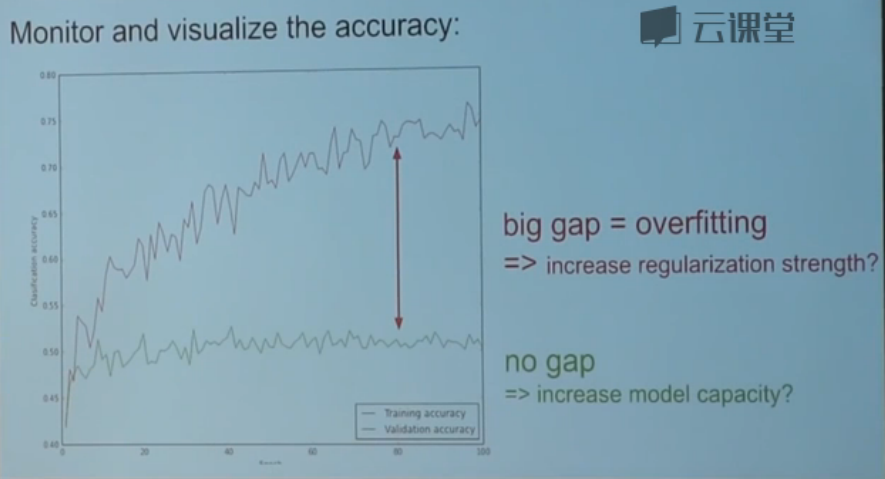


学习率曲线





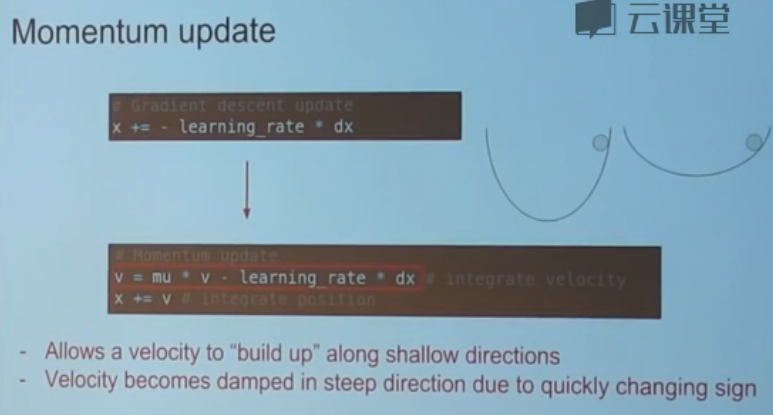
准确率参数，可能是过拟合，这个时候应该提高正则化参数

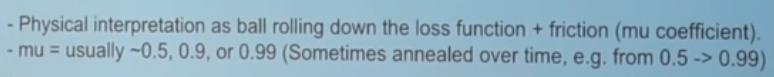


学习率的增量不能太大，也不能太小，大概千分之一左右

Momentum update

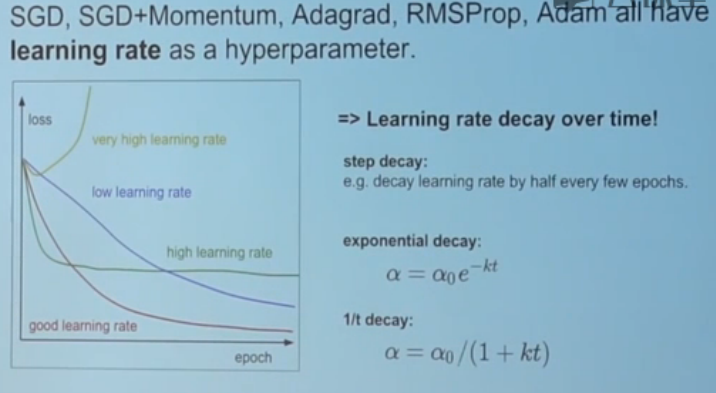
比SGD速度更快，引入动量





学习率

随时间衰减



Dropout

一种很简单，但是又非常有效的改进

训练时随机让一定比例p的神经元失活，使得训练时的网络只是整个网络的子网；

记得在测试时要对期望以p比例处理，使得训练和测试的期望相当；

失活包括了前向和反向

