batch

深度学习的优化算法，说白了就是梯度下降。每次的参数更新有两种方式。

第一种，遍历全部数据集算一次损失函数，然后算函数对各个参数的梯度，更新梯度。这种方法每更新一次参数都要把数据集里的所有样本都看一遍，计算量开销大，计算速度慢，不支持在线学习，这称为Batch gradient descent，批梯度下降。

另一种，每看一个数据就算一下损失函数，然后求梯度更新参数，这个称为随机梯度下降，stochastic gradient descent。这个方法速度比较快，但是收敛性能不太好，可能在最优点附近晃来晃去，hit不到最优点。两次参数的更新也有可能互相抵消掉，造成目标函数震荡的比较剧烈。

为了克服两种方法的缺点，现在一般采用的是一种折中手段，mini-batch gradient decent，小批的梯度下降，这种方法把数据分为若干个批，按批来更新参数，这样，一个批中的一组数据共同决定了本次梯度的方向，下降起来就不容易跑偏，减少了随机性。另一方面因为批的样本数与整个数据集相比小了很多，计算量也不是很大。

基本上现在的梯度下降都是基于mini-batch的，所以深度学习框架的函数中经常会出现batch\_size，就是指这个。

iterations

iterations（迭代）：每一次迭代都是一次权重更新，每一次权重更新需要batch\_size个数据进行Forward运算得到损失函数，再BP算法更新参数。1个iteration等于使用batchsize个样本训练一次。

epochs

epochs被定义为向前和向后传播中所有批次的单次训练迭代。这意味着1个周期是整个输入数据的单次向前和向后传递。简单说，epochs指的就是训练过程中数据将被“轮”多少次，就这样。

举个例子

训练集有1000个样本，batchsize=10，那么：   
训练完整个样本集需要：   
100次iteration，1次epoch。

具体的计算公式为：   
one epoch = numbers of iterations = N = 训练样本的数量/batch\_size