**学习进度**

本周学习任务：

学习吴恩达深度学习课程v5.72，关于神经网络优化模块的内容。

1. 神经网络实践过程中的常见问题
2. 数据集

在构建网络模型的初期，除了需要明确学习目标之外，应该首先考虑数据集的划分、分布及特征归一化输入问题。

在默认情况下，应将**数据集划分**为训练集、验证集及测试集，其中训练集用来训练模型，验证集用来模型优化，测试集用来对最终选定的神经网络做出无偏估计。当数据体量较小时，划分比例常为6：2：2；当数据体量较大时，验证及测试集的规模应小于总量的10%。

关于**数据集分布**问题，在假设基本误差很小的前提下，理想状态是确保训练、验证及测试集的数据应来自同一分布，这使我们能够通过训练、验证及测试集误差，更轻松地判断出模型是否存在偏差或方差问题。然而，在实际应用场景中，更契合学习目标的数据集可能不多，这些“优质”数据集应优先划分给验证及测试集，从而保证验证及测试集的数据来自同一分布。那训练集可以不与验证及测试集保证数据的同一分布吗？以猫识别任务为例，如果符合实际应用情景下的模糊猫图像较少，你应该将它们优先划分给验证及测试集，以确保模型优化的是基于实际情景下常碰到的模糊图像识别能力，这从而使得训练集中引入了来自不同数据分布的清晰猫图像网图，变成了一种可以接收的妥协行为。

采用统一的**归一化输入**操作，将特征分布规范为均值为0，方差为1。这使得数据在空间中分布更均匀，能适应更大的学习率步长，从而加快了训练速度。其中，零均值化公式为：；归一化方差公式为：。

1. 过拟合问题

在基本误差很小的前提下，如果训练集误差较小，而验证集误差相之较大，则认为发生了过拟合现象。此类高方差问题一般通过正则化、数据扩增及early stopping等方法来解决。

其中，正则化是较为常见的解决方法，它通过在损失函数J(w,b)中引入**L2正则化**惩罚项的方式，从而削减模型曲线大角度拉动的能力。此外，**dropout正则化**采用反向随机失活的方式，使得神经网络不再依赖某个具体特征，另一种程度上削减了异常样本对于模型曲线的扰动。这种通过遍历所有结点，每次都以特定概率来决定是否删除副本网络中结点及其连接的方式。虽然使得投入训练的神经网络，得到了随机简化。但是也使得代价函数不再被明确定义，对梯度下降性能检查工作造成了困难。

1. 权重初始化

随意地将权重初始化为零向量，将造成梯度消失或爆炸问题。为此，可以采用Pre-training预训练、Xavier初始化及He初始化等方式，完成权重的初始化工作。其中，**Pre-training**不用从零开始训练新模型，它引用能够解决类似问题且训练好的模型，继承其权重值并基于此进行微调训练。此外，随着网络深度地增加，将参数初始化为符合高斯分布（均值为0，方差为1）的方式，并不能有效解决梯度消失问题。**Xavier初始化**则可通过保持一层网络中的输入输出符合正态分布且方差相近的方式，避免输出的激活值趋向于零，从而解决梯度消失问题。然而，对于含有ReLU等非线性激活函数的神经网络层，Xavier初始化效果极不理想。这种情况实际上是由于ReLU函数让一半的Z值变为零，移除了大约一半的方差造成的。而**He初始化**则通过加倍权重方差的方式，有效克服了这一问题。

1. 梯度检验

我们知道数值法通过导数定义求导时，如果步长足够小，可求得较精确的梯度值，但求解速度较慢；解析法按照导数公式计算梯度，当问题较复杂时，往往容易出错。为此，针对于backprop准确性问题，梯度检验是一个行之有效的策略。

常规做法是将解析法计算的梯度与数值法计算的梯度进行对比，以检验解析法公式的正确性。在数学逻辑上，我们引入相对误差的概念，来衡量正确性程度，即。

当相对误差在1e-7及以下时，表明正确性结果理想。值得注意的是，不要在训练过程中进行梯度检验，它只用于调试环节，且很少频繁使用。

1. Batch Normalization

在训练的过程中，网络中的可学习参数都在持续变化，使得隐藏层中的激活值分布也在不断变换，极大影响训练效果。为此，需要将神经网络模型上一层的输出Z归一化处理，从而限制前层输出的数据分布，以达到提高后层计算效率的目的。

在数学逻辑上。首先，需要计算某层的均值及方差；然后，对该层的Z进行归一化处理，即；最后，还需要引入可学习参数：缩放因子及偏移量，对归一化结果进行缩放偏移操作，即。值得注意的是，对每层输出值进行归一化操作将损坏学习到的原始特征分布，而可学习参数、的引入，能一定程度上恢复该层输出值中所蕴含的特征分布。

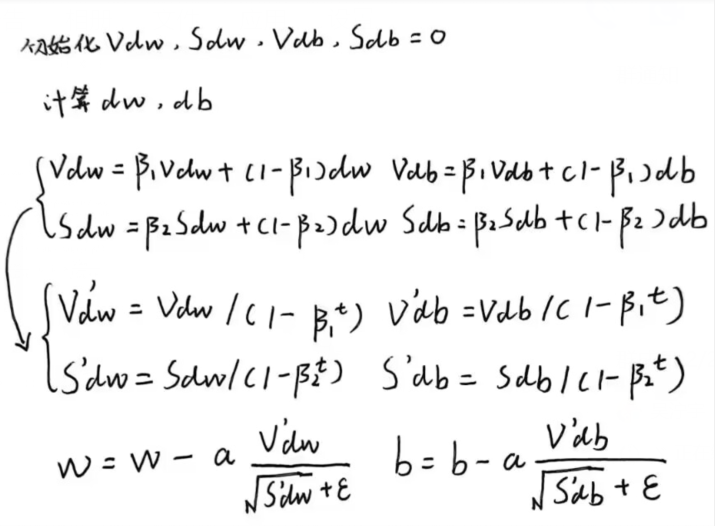
为什么说Batch Normalization能够起到一定的正则化作用？BN将上一层的输出Z进行归一化处理，使网络每一层的输入更接近正态分布，有助于加速训练过程、降低梯度消失和爆炸风险。此外，也有助于缓解传输数据内部协变量偏移问题，一定程度上降低了网络对输入数据中一些异常特征的敏感性，从而起到了正则化作用。

1. 优化算法

优化算法提供了一种有效的方式来搜索损失函数的最优解，使得神经网络模型能够更好地学习和适应输入数据的特征。常见的优化算法有，Batch梯度下降、Mini-batch梯度下降及Adam优化算法。

当训练集较小时，常推荐使用**Batch梯度下降法**，一次性处理完所有训练样本。而针对海量数据，则推荐使用**Mini-batch梯度下降法**，它把训练集切割为若干mini-batch，并分批投喂给算法走一遍完整流程，每一批的训练成果都是1 epoch，有效地提高了模型的训练速度。此类方法的关键问题是：如何适宜地选取mini-batch的大小？经验是，mini-batch的大小一般取，且在[64,512]区间内最佳，具体多少需要根据实际情况自行调试。

值得注意的是，传统梯度下降法都需慢慢摆动到最小值，这种左右波动减慢了梯度下降速度，从而无法使用更大的学习率。为此，**Adam优化算法**结合了Momentum和RMSprop梯度下降法，通过对梯度的一阶动量和二阶动量进行估计，并与学习率相乘来调整每次参数更新的步长，有效减缓了梯度下降的左右波动幅度。其公式推演如下图所示：



经验是：参数推荐0.9，参数推荐0.999，参数推荐，实践中极少调整这些参数的值。

1. 学习率衰减

当学习率固定不变时，梯度下降很难精确收敛，由于受到噪声影响，最终会在最优值附近摆动。为此，我们希望有这样理想的学习率衰减：在学习初期，放心大胆地走；当开始收敛时，小一些的学习率将使最终摆动幅度减小。

其中，是一种常见的学习率衰减方式，该公式中需要调参的有：初始学习率及衰减率decay\_rate。此外，指数衰减也是一种常见的学习率衰减方式，即。

值得注意的是，Adam优化算法本身包含有学习率衰减功能，当使用Adam优化算法时，无需额外添加学习率衰减的实现方式。

1. 超参调试经验

神经网络中存在大量的可调参数，在实践过程中，首先需要猜测调哪些参数可能更有用。经验是，学习率、网络深度、不同层中的隐藏单元数量、激活函数及mini-batch大小等参数，都是值得尝试的调试目标。

明确可调参数后，又该如何取值来调试呢？常见的方法有：网格取值、随机取值及聚焦式取值等方式。其中，网格取值是指在明确取值间隔的情况下，遍历所有可能的取值方式。随机取值，顾名思义是在限定的取值范围内，每次调试都随机取一个参数值，考虑了更多的可能性。而聚焦式取值则由粗糙到精细，试图以参数值的实际效果为标准，逐渐缩小取值区域并愈加逼近最优取值附近。

当明确调试哪些参数且如何取值后，我们的实践思路又是什么呢？典型的实践思路，有“熊猫式”及“鱼子酱式”这两种。前者适合计算资源有限的情况下，进行单线作战，每天都会观察训练效果并不断调试参数。后者则情况相反，它需要更多的计算资源，思路是分别基于不同的参数，并行运行模型。