深度学习笔记 | 第2讲：神经网络的过拟合与正则化

louwill 狗熊会 2018-08-13

新朋友点蓝色字免费订阅

大家好！第二期的深度学习笔记又和大家见面了。在上一期中，小编和大家介绍了机器学习和深度学习的基本关系以及感知机和神经网络的基本原理。本期推送将在上一期的基础上，继续和大家学习探讨神经网络的一些特性。深度学习之所以冠以深字，很大程度上要归功于神经网络包含有很多的隐藏层，这使得深度神经网络能够提取图像和语音等输入不同层级的特征。那么是不是隐藏层越多神经网络的表现就越好呢？

1

机器学习的核心要义

神经网络的隐藏层越多、网络越深是不是神经网络的表现就越好？相信有着机器学习敏锐嗅觉的你肯定会说不。若是神经网络隐藏层越多模型效果越好，那我们还担心啥数据量不够、数据预处理、网络结构和超参数调优等问题呢，直接加隐藏层就好。用脚趾头想就知道肯定不是，虽然较多的隐藏层能够提取输入不同层次的特征，最大程度上拟合输入数据，但是在数据量一定的情况下，盲目加深隐藏层必然导致模型效果

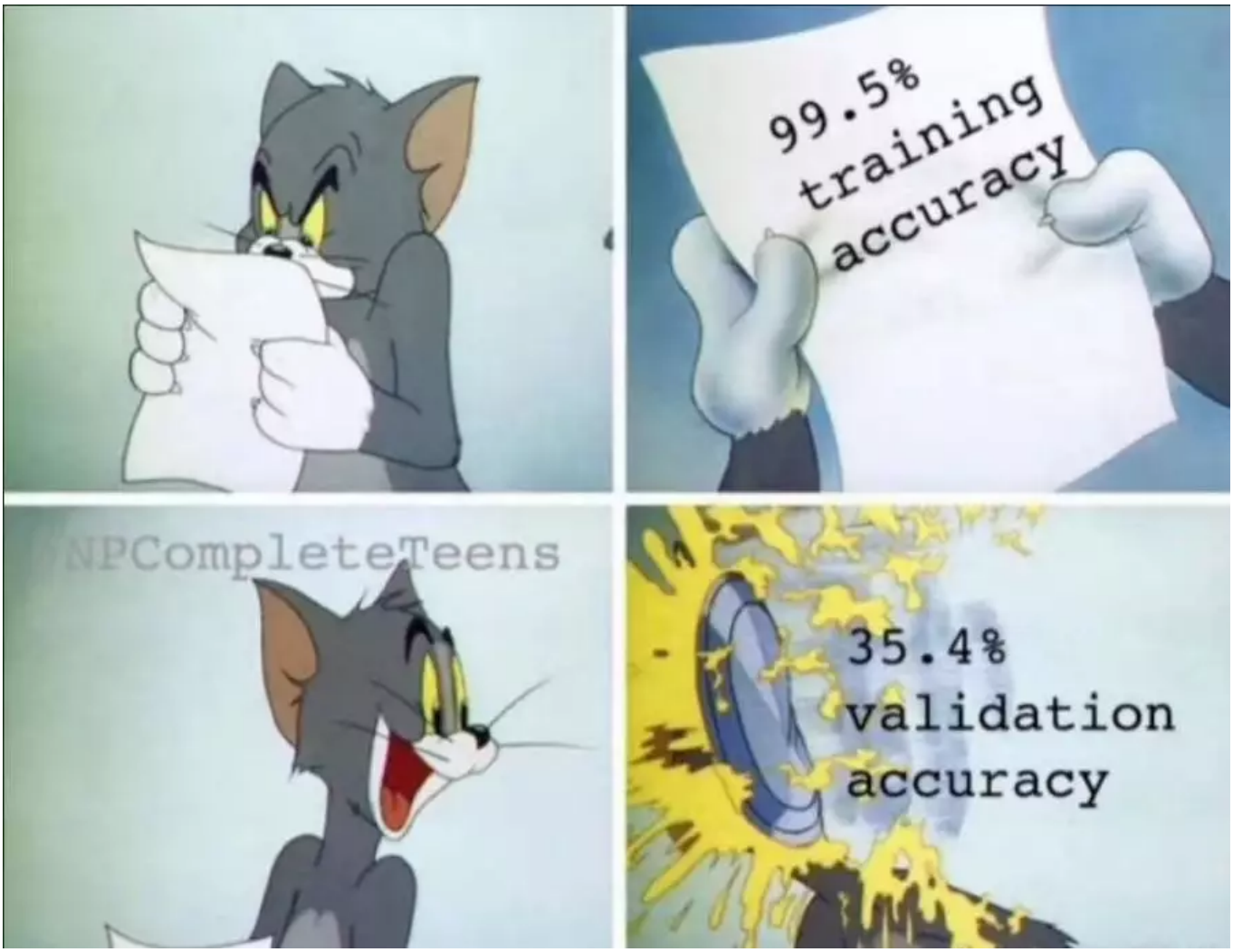
变差。变差的直接原因便是**过拟合**。所以，在探究深度神经网络的性能之前，我们先来好好掰扯一下机器学习中核心问题——**防止过拟合**。

在讨论过拟合之前，我们再先来掰扯下有监督机器学习的核心任务。总的来说，有监督机器学习都可以用如下公式来概括：

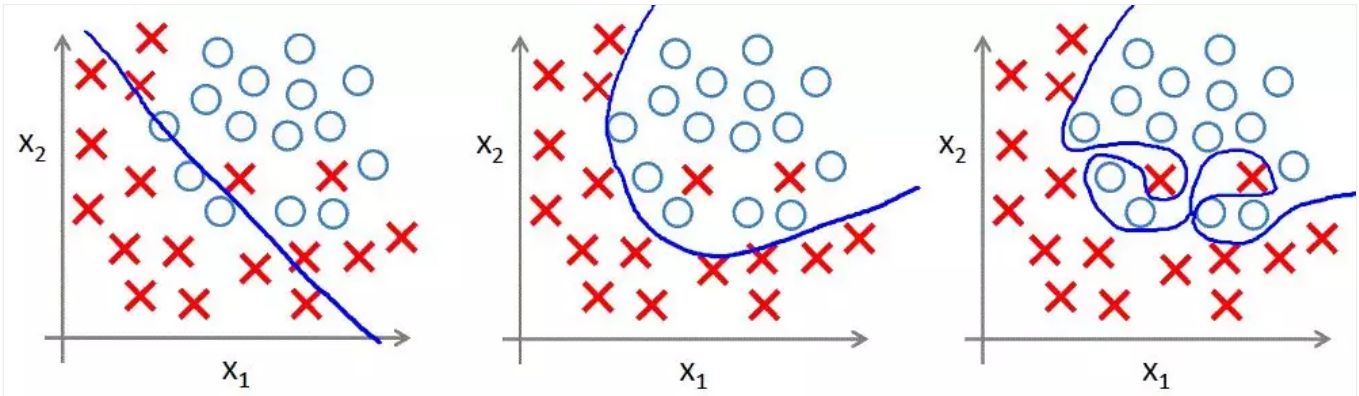
$$\min \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f(x_i)\right) + \lambda J(f)$$

这个公式便是监督机器学习中的损失函数，其中第一项为针对于训练数据集的经验误差项，也就是我们常说的训练误差，第二项为正则化项，也叫惩罚项，用于对模型复杂度的约束和惩罚，正则化项的具体形式小编在下文会细说。所以，所有的监督机器学习的核心任务无非就是正则化参数的同时最小化经验误差。多么简约的哲学啊！在各类机器学习模型中，其中的差别无非就是变着法儿改变经验误差项，也就是我们常说的损失函数。不信你看：当第一项损失函数是平方损失(suqare loss)时，机器学习模型便是线性回归了，当变成指数损失(exp loss)时，模型则是牛哄哄的Adaboost，而当损失函数为合页损失(hinge loss)时，便是大名鼎鼎的支持向量机了！呵~多么直白的领悟。

所以，综上所言，第一项经验损失项很重要，它能变着法儿改变模型形式，我们在训练模型时要最大程度的把它变小。但在很多时候，决定我们机器学习模型生死的关键通常不是第一项，而是第二项正则化项。正则化项通过对模型参数施加约束和惩罚，让我们的模型时时刻刻保持被过拟合的警惕。所以，我们再回到前面提到的有监督机器学习的核心任务：正则化参数的同时最小化经验误差。翻译成人话就是训练集误差小，测试集误差也小，模型有着较好的泛化能力。或是是模型偏差小，方差也小。



但很多时候模型的训练并不如我们愿。当你是名在机器学习领域摸爬滚打已久的选手时，想必你更能体会到模型训练的艰辛和训练集测试集误差同时小的不容易。很多时候，我们都会把经验损失，也就是训练误差降到了极低，但模型一到测试集上，瞬间天崩地裂，模型表现一塌糊涂。这种情况便是小编今天要谈的主题——过拟合。所谓过拟合，**指在机器学习模型训练过程中，模型对训练数据学习过度，将数据中包含的噪声和误差也学习了，使得模型在训练集上表现很好，而在测试集上表现很差的一种现象**。再回顾一下小编在笔记1中的说法：机器学习简单而言就是归纳学习数据中的普遍规律，一定得是普遍规律，像这种将数据中的噪声也一起学习了的，归纳出来的便不是普遍规律，那就是过拟合。



欠拟合、正常拟合与过拟合（图片来自吴恩达机器学习课程）

鉴于过拟合的普遍性和关乎模型的生死性，小编认为，在机器学习实践中，与过拟合作长期的坚持不懈的斗争是机器学习的核心要义。没有之一。其他的诸如特征工程、扩大训练集数量、算法设计和超参数调优等等都是为防止过拟合这个核心问题而服务的。

2
范数与正则化

既然小编说机器学习模型训练的核心要义在于与过拟合作坚持不懈的斗争，那么你肯定要问了：咋斗争啊？除了加大训练数据量和做更加精细化的特征工程之外，最常用的斗争技术便是正则化了。这也是前文所提到的有监督机器学习的第二项正则化项。第二项中 λ 为正则化系数，通常是大于 0 的，是一种调整经验误差项和正则化项之间关系的系数。λ = 0 时相当于该公式没有正则化项，模型全力讨好第一项，将经验误差进行最小化，往往这也是最容易发生过拟合的时候。随着 λ 逐渐增大，正则化项在模型选择中的话语权越来越高，对模型的复杂性的惩罚也越来越厉害。所以，在实际的训练过程中，λ 作为一种超参数很大程度上决定了模型生死。

那除了正则化系数 λ ，正则化项到底长什么样呢？这就需要引入向量和矩阵的 L 范数的概念了。范数在数学上指的是是泛函分析中向量长度的度量，延伸到机器学习中，我们也可以将其理解为向量和矩阵的长度。在机器学习的正则化中，最常用的范数形式莫过于 L1 范数和 L2 范数。

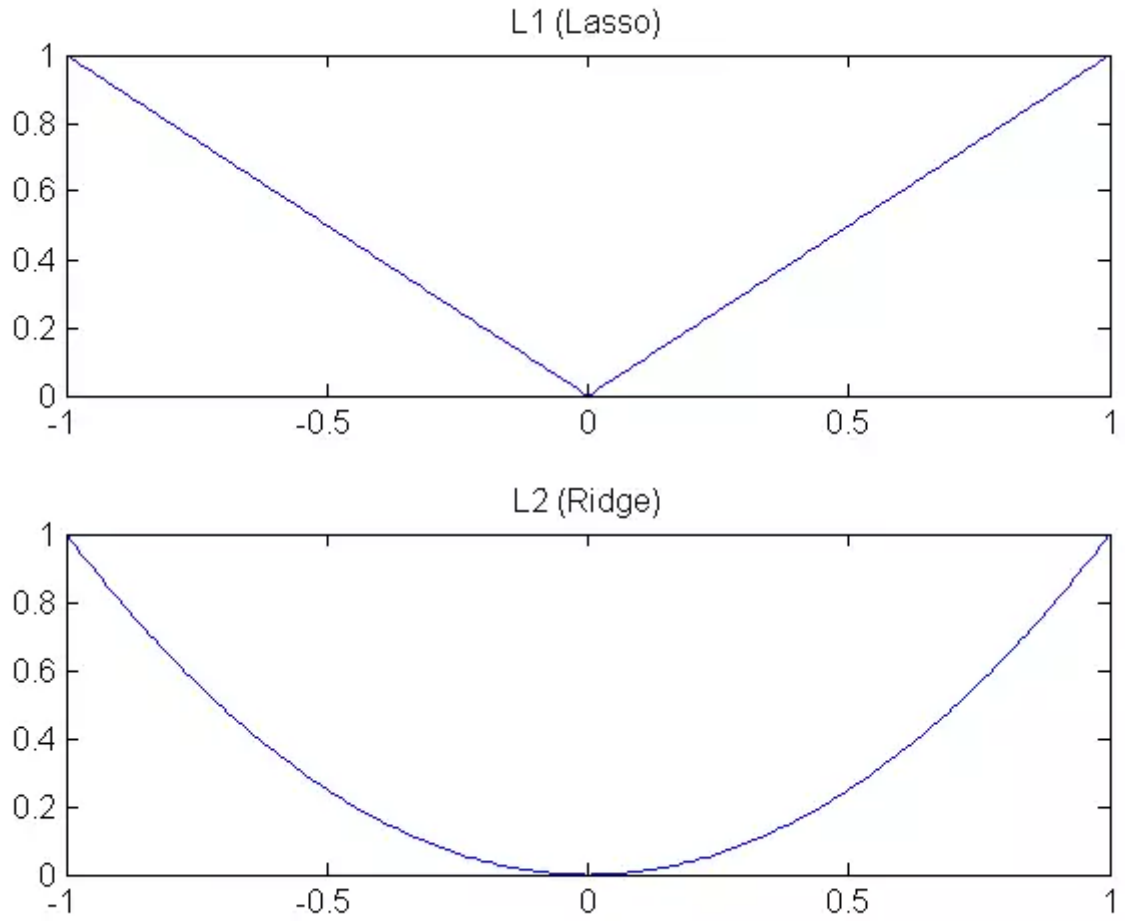
在说常见的 L1 和 L2 之前，咱们有必要先来看一下 L0 正则化。L0 正则化也就是 L0 范数，即矩阵中所有非 0 元素的个数。如何我们在正则化过程中选择了 L0 范数，那该如何理解这个 L0 呢？其实非常简单，L0 范数就是希望要正则化的参数矩阵 W 大多数元素都为 0。如此简单粗暴，让参数矩阵 W 大多数元素为 0 就是实现稀疏而已。说到这里，权且打住，想必同样在机器学习领域摸爬滚打的你一定想问，据我所知稀疏性不通常都是用 L1 来实现的吗？这里个中缘由小编就不去深究了（会涉及到很多数学理论问题），简单说结论：在机器学习领域，L0 和 L1 都可以实现矩阵的稀疏性，但在实践中，L1 要比 L0 具备更好的泛化求解特性而广受青睐。先说了 L1，好像还没解释 L1 范数是什么，L1 范数就是矩阵中各元素绝对值之和，正如前述所言，L1 范数通常用于实现参数矩阵的稀疏性。至于为啥要稀疏，稀疏有什么用，通常是为了特征选择和易于解释方面的考虑。

$$\min \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f(x_i; w)\right) + \lambda||w||_1$$

再来看 L2 范数。相较于 L0 和 L1，其实 L2 才是正则化中的天选之子。在各种防止过拟合和正则化处理过程中，L2 正则化可谓风头无二。L2 范数是指矩阵中各元素的平方和后的求根结果。采用 L2 范数进行正则化的原理在于最小化参数矩阵的每个元素，使其无限接近于 0 但又不像 L1 那样等于 0，也许你又会问了，为什么参数矩阵中每个元素变得很小就能防止过拟合？这里我们就拿深度神经网络来举例说明吧。在 L2 正则化中，如果正则化系数变得比较大，参数矩阵 W 中的每个元素都在变小，**线性计算的和 Z 也会变小，激活函数在此时相对呈线性状态**，这样就大大简化了深度神经网络的复杂性，因而可以防止过合。

$$\min \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f(x_i; w)\right) + \frac{\lambda}{2}||w||^2$$

至于 L1 和 L2，江湖上还有一些混名，L1 就是江湖上著名的 **lasso**，L2 则是**岭回归**。二者都是对回归损失函数加一个约束形式，lasso 加的是 L1 范数，岭回归加的是 L2 范数。可以从几何直观上看看二者的区别。



lasso 和 Ridge

3

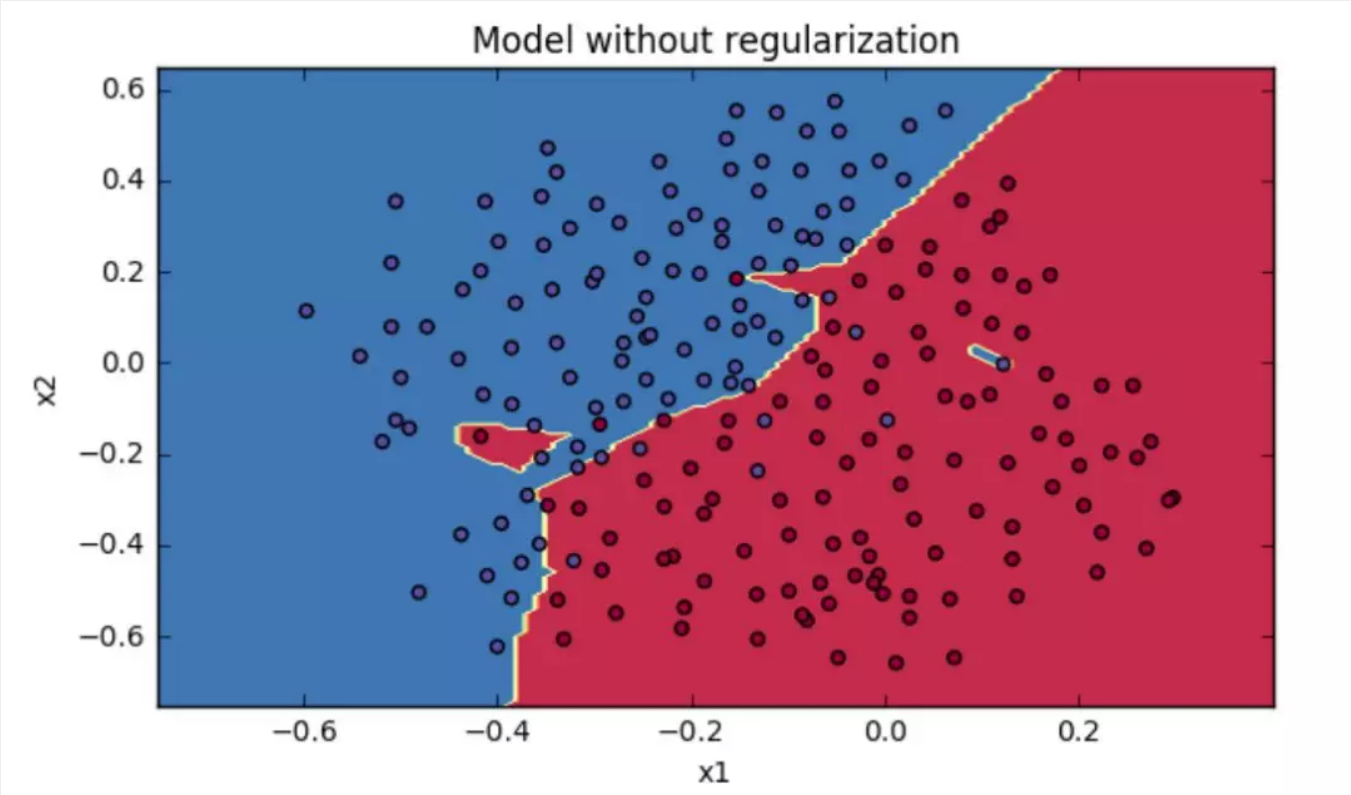
神经网络的正则化和dropout

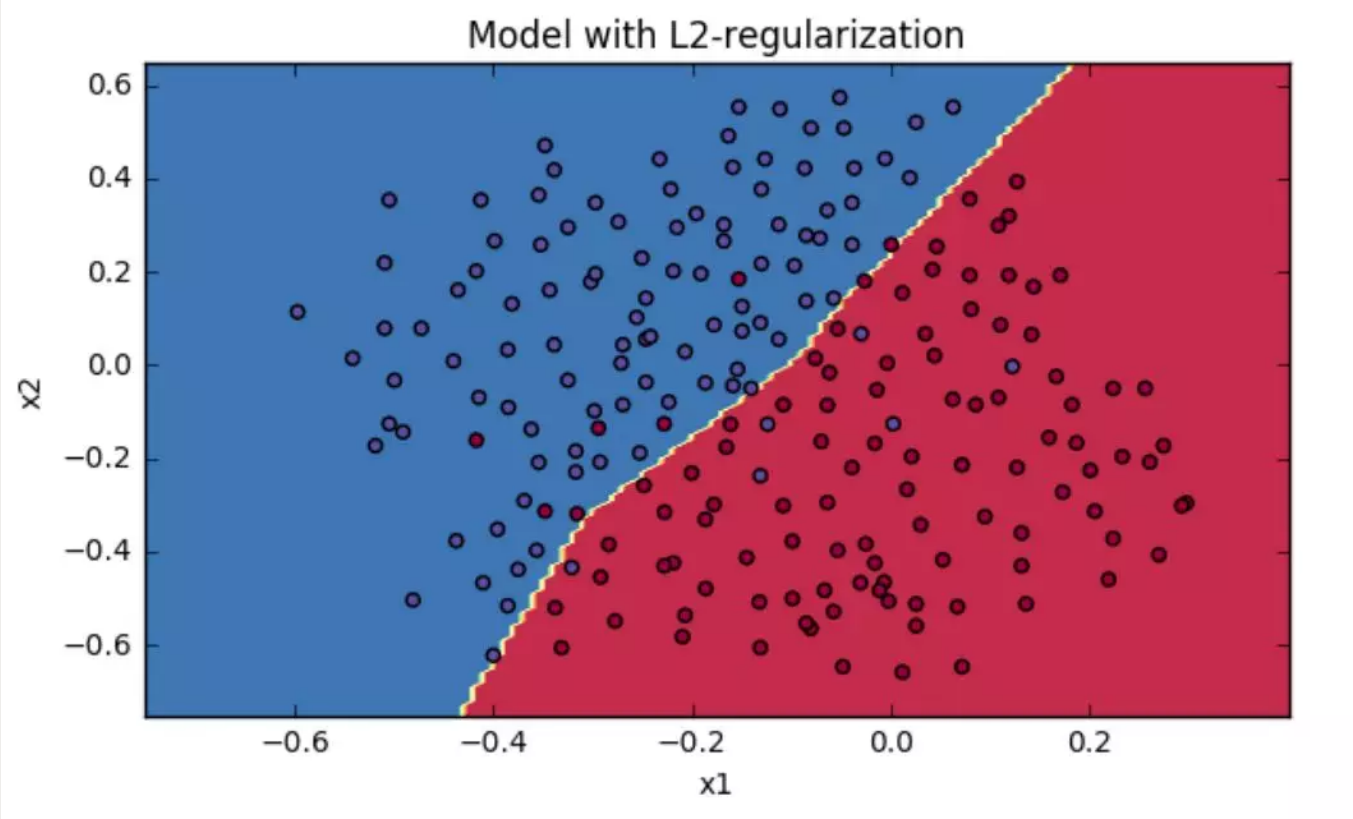
铺垫了两节的内容，就是我们说今天的实际主题：如何防止神经网络的过拟合或者说如何给神经网络加正则化项。正如前文所说的监督机器学习的核心公式，假设我们已经给神经网络采用了交叉熵损失函数作为第一项了，现在我们需要给核心公式加上正则化项，假设我们就加 L2 正则化项吧。如此一来，带正则化的交叉熵损失函数就变为如下形式：

$$J = -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\log\left(a^{[L](i)}\right) + (1-y^{(i)})\log\left(1-a^{[L](i)}\right)\right)$$

$$J_{regularized} = \underbrace{-\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\log\left(a^{[L](i)}\right) + (1-y^{(i)})\log\left(1-a^{[L](i)}\right)\right)}_{\text{cross-entropy cost}} + \underbrace{\frac{1}{m}\frac{\lambda}{2}\sum_{l}\sum_{k}\sum_{j}W_{k,j}^{[l]2}}_{\text{L2 regularization cost}}$$

不加正则化项和加了 L2 正则化项的损失函数进行神经网络的模型训练和进行训练的分类模型效果可如下图所示：



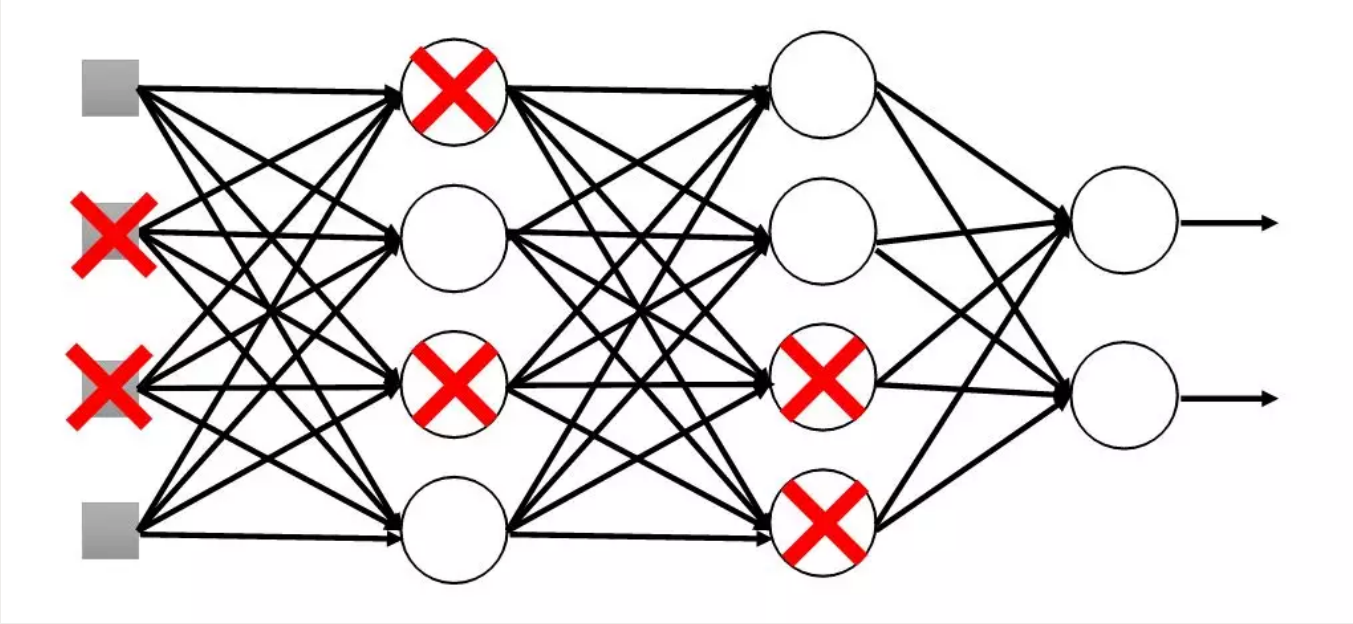


图片截图于吴恩达deeplearningai课程作业

正如上图所示，没加正则化的深度神经网络的训练结果存在着明显的过拟合现象，分类结果存在较大的误差。而加了正则化项的神经网络模型训练效果则要好得多。

除了给损失函数加正则化项外，神经网络还有自己独特的方法防止由过多的隐藏层带来的过拟合问题，这便是 dropout 。从字面意思上理解，你可以把它理解为丢弃、失活等含义，dropout 的正式含义是指在神经网络训练的过程中，对所有神经元按照一定的概率进行消除的处理方式。在训练深度神经网络时，dropout 能够在很大程度上简化神经网络结构，防止神经网络过拟合。所以，**从本质上而言，dropout 也是一种神经网络的正则化方法。**

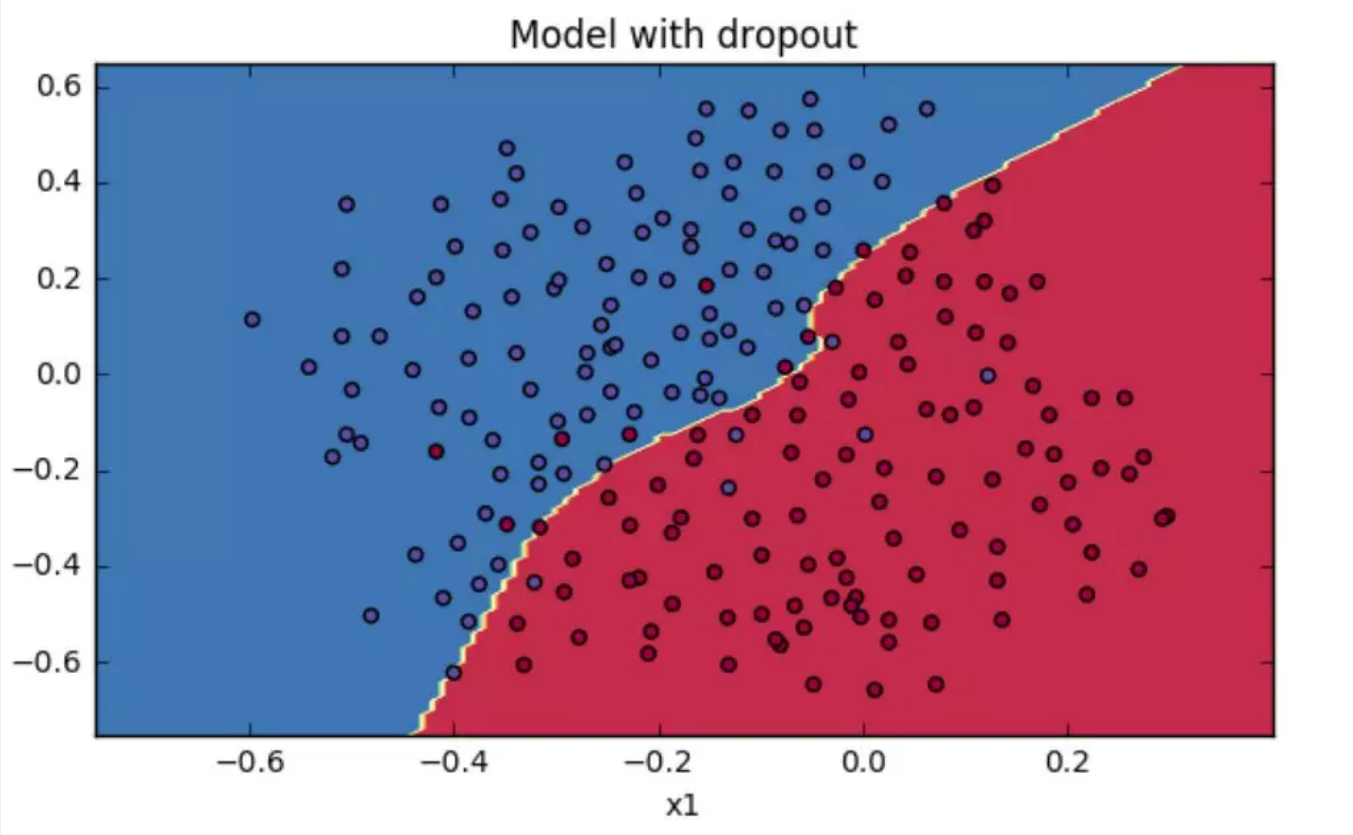
假设我们要训练一个4 层（3个隐层）的神经网络，该神经网络存在着过拟合。于是我们决定使用 dropout 方法来处理，dropout 为该网络每一层的神经元设定一个失活（drop）概率，在神经网络训练过程中，我们会丢弃一些神经元节点，在网络图上则表示为该神经元节点的进出连线被删除。最后我们会得到一个神经元更少、模型相对简单的神经网络，这样一来原先的过拟合情况就会大大的得到缓解。这样说似乎并没有将 dropout 正则化原理解释清楚，我们继续深究一下：为什么 dropout 可以可以通过正则化发挥防止过拟合的功能?



带dropout的神经网络结构

因为 dropout 可以随时随机的丢弃任何一个神经元，神经网络的训练结果不会依赖于任何一个输入特征，每一个神经元都以这种方式进行传播，并为神经元的所有输入增加一点权重，dropout 通过传播所有权重产生类似于 L2 正则化收缩权重的平方范数的效果，这样的权重压缩类似于 L2 正则化的权值衰减，这种外层的正则化起到了防止过拟合的作用。

所以说，总体而言，dropout 的功能类似于 L2 正则化，但又有所区别。你可以将 dropout 理解为对神经网络中每一个神经元加上一道概率流程，使得在神经网络训练时能够随机使某个神经元失效。还有需要值得注意的一点是，对于一个多层的神经网络，我们的 dropout 某层神经元的概率并不是一刀切的。对于不同神经元个数的神经网络层，我们可以设置不同的失活或者保留概率，对于含有较多权值的层，我们可以选择设置较大的失活概率（即较小的保留概率）。所以，总结来说就是如果你担心某些层所含神经元较多或者比其他层更容易发生过拟合，我们可以将该层的失活概率设置的更高一些。跟前面 L2 正则化一样，我们基于同样的数据对神经网络模型加上 dropout 后的效果如下所示：



图片截图于吴恩达deeplearningai课程作业

可见，带有 dropout 结构的神经网络模型的效果类似于 L2 正则化，二者同样可以防止神经网络的过拟合问题。在实际的深度学习试验中，除了 L2 正则化和 dropout 之外，我们还可以通过对验证集效果进行监测来执行 early stopping (早停) 策略来防止过拟合。

当然，过拟合作为一种同数据与生俱来的特性，几乎不可能完全避免，生产环境下数据脏乱差现象普遍，包含各种各样的噪声也在所难免，我们不能完全避免过拟合，但是我们可以通过本文所说的各种手段来缓解过拟合。**历史车轮滚滚向前，只要机器学习一直发展下去，与过拟合的斗争也就会一直持续下去。**

深度学习笔记第2讲的内容到这里就结束了，在笔记2中，小编和大家一起梳理了机器学习的核心任务和基本要义，明确了过拟合现象的普遍性和与之斗争的长期性。我们重点关注了神经网络缓解过拟合的方法，对 L2 正则化和 dropout 有了深入的理解。咱们下期见！

【参考资料】

[1] https://www.deeplearning.ai/

[2] Kukačka J, Golkov V, Cremers D. Regularization for Deep Learning: A Taxonomy[J]. 2017.

[3] Neyshabur B. Implicit Regularization in Deep Learning[J]. 2017.

作者简介

鲁伟，狗熊会人才计划一期学员。目前在杭州某软件公司从事数据分析和深度学习相关的研究工作，研究方向为贝叶斯统计、计算机视觉和迁移学习。

识别二维码，查看作者更多精彩文章

点击“**阅读原文**”，成为狗熊会会员！

阅读原文