深层神经网络

什么是深度学习网络

在logistic回归,以及单隐层神经网络之上,还有双隐层和五隐层的神经网络
我们通常认为logistic回归是一个浅层模型,而我们在上面提到的最后一种就要深得多,**所以模型的深浅实际上是一个程度的问题**

为什么需要深层神经网络

在最近的人工智能领域,**人们发现有些函数只有非常深层的神经网络才能学习**,而浅一些的模型通常无法学习

在处理具体问题时,**我们需要先判断究竟需要多深的神经网络**,所以先尝试使用logistic回归是一个不错的选择,然后依次尝试双层,三层,以此类推,把隐层数量当成一个**可以自由选**

择数值大小的超参数,然后在保留交叉验证数据上评估,或者使用你自己的开发集进行评估

用来描述深度神经网络的符号

我们首先来看一个三隐层的神经网络,隐层中单元的数量分别为5,5,3,然后输出层有一个输出单元

那么我们就用一个大写的符号L表示层数,用$n^{[l]}$表示第l层的单元数量,特别地输出层的单元个数,也就是$n^{[0]}$,通常认为是样本的特征个数,也就是$n^{[0]} = n_x$在这个例子当中$n^{[0]} = n_x$
对于第l层,我们使用$a^{[l]}$来表示第l层中的激活函数,所以有$a^{[l]} = g(z^{[l]})$,并且我们使用$w^{[l]}, b^{[l]}$来计算$z^{[l]}$,并且对于输入层,$a^{[0]} = x$,对于输出层,$a^{[L]} = \hat{y}$

深层网络中的前向传播

深层网络中的前向传播过程

如同之前在单隐层神经网络中讲过的
我们输入样本X(也就是$A^{[0]}$),在经过与权重参数矩阵$W^{[1]}, b^{[1]}$以及激活函数$g^{[1]}$的计算后,得到激活值$A^{[1]}$,再将激活值输入第二层节点,以此类推,直到输出层的激活值$A^{[L]}$,将其作为预测值$\hat{y}$

而对于这一传播过程的向量化计算,我们已经在前面的章节进行过学习,深层和浅层的计算没有区别

但是,在这一过程中,我们很显然会在不同节点中经历许多类似的重复计算,这一过程是无法被向量化的,所以这个过程我们可以使用显式的**for**循环进行计算

关于反向传播中导数矩阵的维数(没太多内容但感觉比较重要)

对于W, b,假设所在层有n个节点,上一层有n_x个节点,W的维数为(n,n_x),b为(n,1)
对于A而言,假设计算出该激活值的层有n个节点,X中有m个样本,其维数为(n,m),而Z则与上标相同的A维数相同

对于反向传播中dW, db的维数，这些矩阵通常与正向传播中与其上标相同的W, b的维数相同,对于dZ, dA也是同理

为什么使用深度表示

深度网络究竟在计算什么

举例

人脸识别

假设我们在建设一个人脸识别或人脸检测系统,我们可以来具体看一看深度神经网络究竟能做什么

当我们输入一张脸部的照片,**我们就可以把深度神经网络的第一层当成一个特征探测器,或者是边缘探测器**
在这一个例子中,我们会建立并了解一个大概有20个隐藏单元的深度神经网络是怎么针对这张图进行计算的

由于这里的一些细节知识涉及到卷积神经网络的内容,所以不做详细阐述

我们可以将第一层的单元理解为寻找图片中的每一个边缘部分,然后组合起来,在第二层又寻找身体的每一个部分,再组合起来,最后就可以进行对人脸的识别

我们在直觉上可以把这种神经网络的前几层当做探测简单的函数,比如边缘,然后再与后面几层结合起来,就可以学习**更多的复杂函数**
在这里我们需要理解一个技术上的细节,那就是边缘探测器实际上相对来说都是针对照片上非常小块的面积,而面部探测器就会针对大一些的区域

语音识别

这实际上就是一个从简单到复杂的过程,这种组成方法也可以应用在图像,或者说人脸识别以外的其他数据上,比如语音识别——前面的层可以探测一些低层次的音频波形的一些特征,比如音调的变化,以及白噪音的声音,然后将这些波形组合在一起,就能去探测声音的基本单元——在语言学中叫做音位,用英语来做比喻就是各种各样的元音和辅音,将这些组合起来,就能识别各种各样的单词,再来就能识别词组,句子

深层神经网络究竟是如何做到的

所以深度神经网络的许多隐层中,**前几层能学习一些低层次的简单特征,后几层就能把简单的特征组合起来,去探测更加复杂的东西**
与此同时,我们所计算的前几层,到了网络深层时,就能做许多复杂的事情

一些神经科学家觉得这与人的大脑很类似,因为人的大脑也是先探测简单的东西,然后组合起来才能探测复杂的物体

为什么神经网络有效

这一理论来自于电路理论,并且与我们可以使用电路元件,通过各种逻辑门计算哪些函数有着分不开的关系
在非正式的情况下,这些函数都能通过相对单元数量较少,但是在网络中较深的层来实现,**而如果是在较浅的层中,可能就需要指数级别增长的单元数量**

搭建深层神经网络块

搭建一层的函数

我们首先选择神经网络中的一层来进行搭建

在第l层,我们有参数$W^{[l]}$和$b^{[l]}$
在正向传播里,有输入的激活函数,以及前一层传来的输入$a^{[l-1]}$和输出$a^{[l]}$,对于我们通过计算得到的$z^{[l]}$,我们可以通过一个字典"cache"储存起来,因为这里的对以后得正向反向传播的步骤都非常有用

在反向传播中,我们会需要实现一个函数,输入为$da^{[l]}$,输出$da^{[l-1]}$的函数,(实际上的输入还包括了储存在"cache"中的$z^{[l]}$),还需要输出$dw^{[l]}$,$dz^{[l]}$,以实现梯度下降学习

神经网络在函数中的计算过程

在实现了这些函数后,神经网络的计算过程就是这样的:
首先进行正向传播,输入$a^{[0]}$,通过第一个正向函数得到$w^{[l]}, b^{[l]}, a^{[l]}$,在第二层中也计算出

$w^{[2]}, b^{[2]}, a^{[2]}$,然后以此类推,直到我们算出最终输出值,在这一过程中我们缓存了所有的z

然后进行反向传播,输入da,得到各层的梯度,并将dw, db缓存,更新参数,完成一次迭代

参数和超参数

什么是超参数

首先让我们来看看模型里的W和b
除此之外,我们还需要输入其它参数到学习算法之中,比如学习率α,以及梯度下降法循环的次数,也就是迭代次数$iterations$,或者说还有隐层数L,或者隐藏单元数$n^{[l]}$,也可以选择激活函数,$ReLU, tanh$**或是**$sigmoid$
这些都需要我们来设置

这些数字实际上控制了最后参数W, b的值,**所以我们称这些数为超参数**

除了这些,实际上还存在着其他的超参数,比如$momentum, minibatch$的大小,几种不同的正则化参数等等

如何选择超参数

所以我们可以发现,超参数的选择有很多可能性,**所以我们需要不停尝试**

在今天的深度学习应用领域,**超参数的选择仍然是一个十分经验性的过程**,我们可能大致知道最好的学习率的值,在实际尝试得到结果后,我们可能发现,这个值可能需要更改

所以,如果不知道应当选择什么学习率,那么可以先选择一个进行尝试,看看损失函数的值有没有下降,通过不断的尝试找到一个最佳的值
类似地,对于其他超参数也是如此

对于同一个模型,最优的超参数也不是一成不变的,通常来说,可能由于硬件的变化等原因,每几个月可能就需要调整一次参数,**所以我们可能需要尝试保留交叉检验或者类似的检验方法**,然后选择一个对于这个问题效果比较好的数值