数字图像处理报告三：卷积神经网络简述

姓名：鲁国锐　　学号：17020021031　　专业：电子信息科学与技术

2020 年 3 月 18 日

目录

1 题目描述

请简述卷积神经网络的基本原理。

2 深度学习

由于卷积神经网络的工作是建立在深度学习的大框架之下的，所以在简述卷积神经网络之前首先要全面了解一下什么是深度学习。本节将通过三个模块对其进行描述。

2.1 深度学习的发展

机器学习是人工智能领域的一个重要学科，自其产生以来在诸多方面都取得了巨大的成功。深度学习则是机器学习的一个重要分支，也是目前实现人工智能的一条重要途径。

深度学习是神经网络发展到一定时期的产物 [6]。尽管它近年来才逐渐为人们所熟知，但其起源可追溯到 1943 年 McCulloch 等人 [3] 提出的 McCulloch-Pitts 计算结构，它通过模拟人体神经元的工作原理来解决问题，但需要手动设置权重，十分不便。之后虽然由 Rosenblatt 教授 [1] 提出了可以自动设置权重的感知机模型，但随后又被 Minsky 教授和 Papert 教授于 1969 年证明感知机模型只能解决线性可分问题，并且否定了多层神经网络训练的可能性 [4]，致使此后很长一段时间内该领域的研究基本处于停滞状态。

直到 20 世纪 80 年代，Rumelhart 团队提出反向传播算法（Back Propagation, BP）[5]，开启了机器学习的“浅层学习”时代 [7]，神经网络才再次焕发生机。而 2006 年机器学习泰斗 Hinton 及其团队在 Science 上发表论文并首次提出” 深度学习” 概念 [2]，则又一次将关于神经网络的研究推向高潮。

随着“互联网 +”模式的发展以及计算机性能的不断提高，人工智能开始逐渐与各个领域融合，而深度学习由于其可以利用大数据来填补专业知识的特性而一跃成为其中的佼佼者。各大公司争相成立深度学习的研究院并发起与其相关的开发项目，高校之中也纷纷建立人工智能相关的博士、硕士点甚至本科专业。这一切都表明，深度学习正飞速地发展着，并不断壮大。

回顾深度学习的发展历史，其起源虽然久远，但它真正的兴起应当是在反向传播算法提出之后，所以也有文献在总结深度学习历史时直接从 20 世纪 80 年代开始 [7]。有鉴于此，本节中将重点介绍反向传播相关的概念及其数学原理。

2.2 前向传播

在讲反向传播之前首先要提一下前向传播，如图1所示。

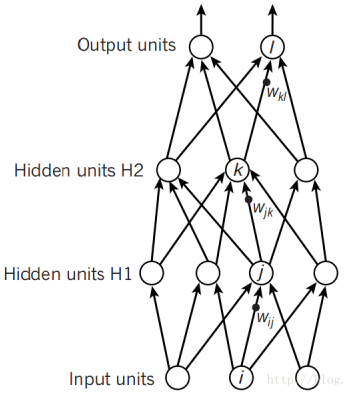

图 1: 全连接神经网络示意图[1]

[1]图片来自网络

所谓前向传播，其实就是计算一个十分复杂的复合函数。图1中每一个节点都代表着两次计算：

$$z^{[l]} = w^{[l]} \cdot a^{[l-1]} + b^{[l]} \tag{1}$$

$$a^{[l]} = \sigma^{[l]}\left(z^{[l]}\right) \tag{2}$$

其中 $\sigma(x) = \frac{1}{1+e^{-x}}$，我们称其为“激活函数”。

从这里我们可以看出，从输入节点开始，沿任意路径到一个输出节点，就构成了一个复合函数，且每一层函数都有着相同的形式。同时我们也可以看出，这若干个复合函数在一起就构成了一个神经网络。

2.3 反向传播

不难想见，若不加任何处理，随机设置公式2中的参数 w 和 b 的值，得到的结果肯定也是随机的，我们无法用其来进行任何决策或预估。所以我们需要减少误差，而要减小误差我们必须要有真值才行，于是这里又涉及到有监督学习与梯度下降法的问题，我们将分别在2.3.1节和2.3.2节讲述。

2.3.1 数据集与有监督学习

深度学习得以进行的一大前提条件就是数据集。我们通过给模型大量的数据进行训练使其能够完成特定的任务。

数据集通常会分为训练集和测试集，训练集和测试集中又会被分为样本和真值两个部分。而对样本或真值的处理就分出了有监督和无监督学习。

有监督学习，顾名思义，就是需要人为的“监控”。这里所说的“监控”其实就是指为每一个训练样本和测试样本标注真值，或者是对输入样本进行一些手工处理。

相应的，如果不对数据做任何处理，则称为无监督学习。

2.3.2 梯度下降法

关于梯度下降法的作用我们可以形象地理解为“找谷底”，如图2所示。

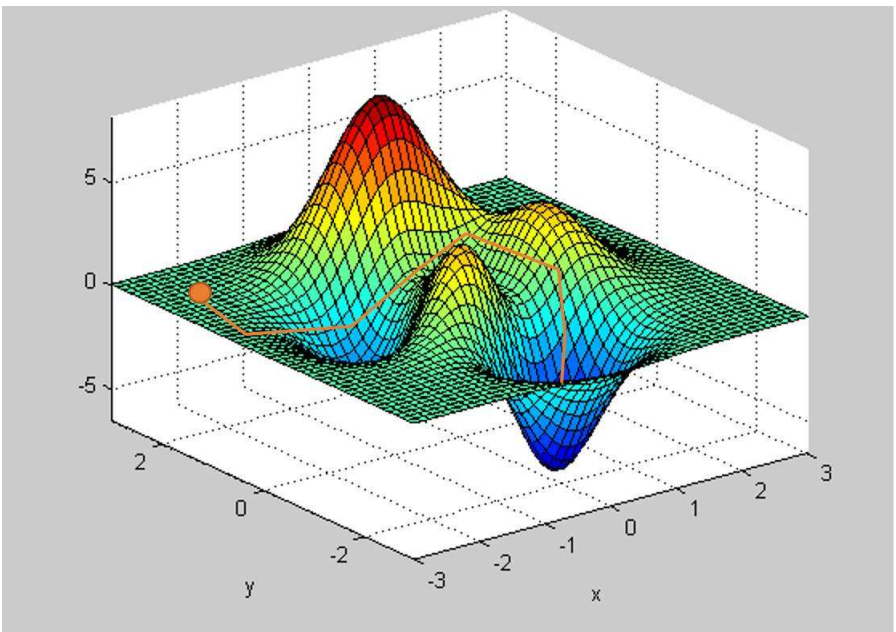

图 2: 梯度下降示例[1]

[1]图片来自网络

假设我们现在正处在一个山谷的某处，需要走到谷底，但却不知道路线，怎么办？这种情况下一种理论上可行的办法是朝着当前位置最陡峭的方向向下迈一步，之后再向新位置最陡峭的方向向下迈一步，如此反复，不出意外的话，我们肯定可以到达谷底。

上述过程其实就是梯度下降法的原理。现在我们有了一个很大很深的神经网络，可以通过前向传播算出一个结果，同时对于每一个输入都有一个真值与之对应，所以接下来要做的就是如何利用这些条件来找出输入与输出之间的对应关系。

在第2.2节中曾说过，一个神经网络其实就是若干个复合函数并列放在一起，并且只要这些复合函数足够“复杂”，我们就可以用这个神经网络来拟合数据集中输入与输出的关系。但如果仅仅去随机初始化网络中的参数，很难得到一个令人满意的结果，所以在初始化之后我们还需要去优化这些参数，使得神经网络输出的结果尽可能接近真值。所用的方法就是梯度下降。

对于一个多元函数，我们想要求它的极小值，常用的做法是求它的导函数，令其等于 0。然而这种做法难以编程实现，因而可以借鉴上述“找山谷”的思路：先求出当前点的梯度，然后令坐标向梯度的负方向移动一小步，之后再求新位置的梯度，并重复之前的步骤，这样就可以逐渐移至极小值点附近。而在深度学习中，我们就是用这样的方式优化每一个节点中的参数值，最终使得整个网络可以较好地去拟合输入输出之间的关系。

2.3.3 反向传播的具体实现

有了梯度下降的概念后，我们就不难想出具体的实现方法，也就是反向传播。

由于每一个节点的函数形式都是固定的，那么它的导函数形式也是固定的，根据式1和式2我们可以很容易写出（公式中的 J 表示神经网络最终的输出）：

$$\frac{\partial J}{\partial z^{[l]}} = \frac{\partial J}{\partial a^{[l]}} * \sigma^{[l]\prime}(z^{[l]}) \tag{3}$$

$$\frac{\partial J}{\partial w^{[l]}} = \frac{\partial J}{\partial z^{[l]}} \cdot a^{[l-1]} \tag{4}$$

$$\frac{\partial J}{\partial b^{[l]}} = \frac{\partial J}{\partial z^{[l]}} \tag{5}$$

$$\frac{\partial J}{\partial a^{[l-1]}} = \frac{\partial J}{\partial z^{[l]}} \cdot w^{[l]} \tag{6}$$

在优化时，我们令：

$$w^{[l]} = w^{[l]} - \alpha \cdot \frac{\partial J}{\partial w^{[l]}} \tag{7}$$

$$b^{[l]} = b^{[l]} - \alpha \cdot \frac{\partial J}{\partial b^{[l]}} \tag{8}$$

这里 α 叫学习率，相当于我们在2节中说的所跨一步的步长，是一个需要人根据经验来调节的超参数。

3 卷积神经网络

在介绍完了深度学习工作的基本过程之后，再来讲述卷积神经网络的基本原理就要容易得多了。同样，我们将分三个小节来对其进行描述。

3.1 全连接网络的缺陷

在2.2节中我们曾给出了全连接神经网络的示意图。然而在现实情况中，比如处理一张图片时，这样做会产生巨大的运算量。例如我们要处理一张 1024×1024 的图片，那么仅仅在输入层就会有超过 100 万个节点，这样势必会产生大量的开销。另外，由于在输入图像时，会将一张图片“拉”成一个向量，这样在一定程度上还会损失结构信息。

3.2 卷积神经网络的优势及工作过程

为了应对上述情况，人们提出了卷积神经网络，通过对卷积核的复用来减少网络参数。这里的卷积核其实就相当于一个滤波器，逐步扫描整幅图像，把卷积核中的各点值与对应的图片像素相乘再相加，得到的结果放在卷积核中心对应的图像位置上，如图3所示。

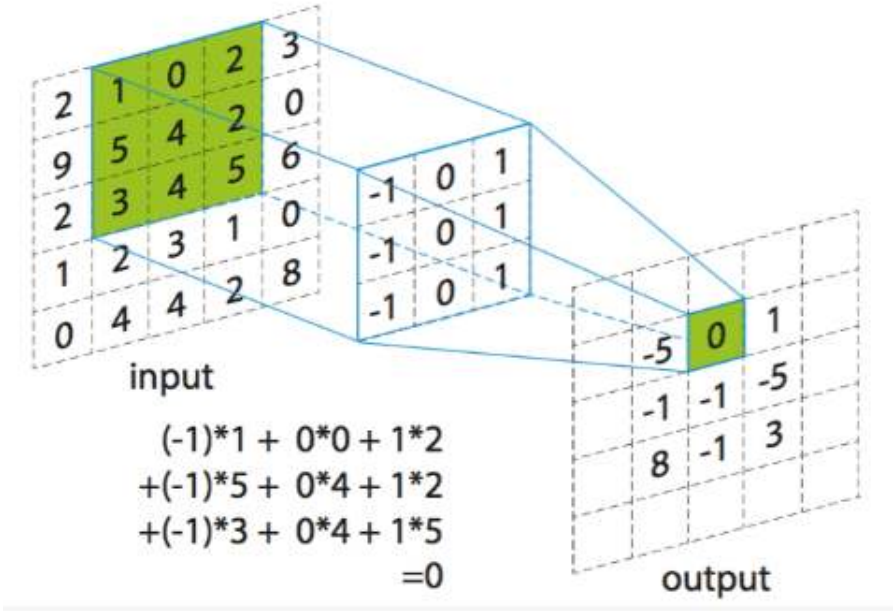

图 3: 卷积核的工作过程[1]

在这种情况下，网络需要学习的参数就变成了卷积核（或者说滤波器）每个点的值。但由于每一层的卷积核数量往往不会太多，所以需要学习的参数也会大大减少。

3.3 卷积神经网络的基本思想

到这里我们可能会有些疑问：为何每一层的卷积核数量不用太多？

其实正如3.2节所说的，每一个卷积核可以被理解为一个滤波器，而滤波器的功能就是提取图像中的某一类信息，且能够被同一个滤波器提取出来的信息都具有某种相同的“模式”。**更确切地说，图像中具有某种相同模式的信息都能够被同一个滤波器提取出来**，例如变化剧烈的边缘可以用拉普拉斯算子等提取，而这也正是卷积核数量不用太多的原因！

从直观感受来看，不论图像有多复杂，都能够被看作是由一系列基本的要素构成的，而这些要素正是图案的模式。比如一个图像的轮廓可以看作是由若干条不同角度的小线段组成，所以为了提取它的轮廓，我们可以用对应数量的滤波器来进行卷积，每一种滤波器都专门用来提取某个特定的要素。更关键的是，**在某一个局部图案中可以找到的要素，往往也能在其它图案总找到**，比如桌子和课本都有可能会包含垂直的小线段。

基于这样的假设，我们就可以用少量的滤波器，即卷积核，来提取一整幅图像中所有需要的信息，从而达到减少参数的目的。而从实际应用的效果来看，该假设是完全成立的。

4 深度学习中的一些问题

第2节和第3节介绍了深度学习及卷积神经网络的工作方式及基本原理，本节我们将讨论深度学习中存在的部分问题作为补充。

[1]图片来自网络

4.1 梯度消失与梯度爆炸

我们在第2.2节中曾提到过激活函数 $\sigma(x)$，同时在第2.3中不难看出在优化网络参数时无法回避对激活函数求导，那么我们现在来考察一下 $\sigma(x)$ 导函数的性质：

$$\begin{aligned}\sigma'(x) &= \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2} \\ &= \sigma(x) - [\sigma(x)]^2 \\ &\leq \frac{1}{4}\end{aligned} \tag{9}$$

由于我们在进行梯度反传时实际上是在用链式法则求导数，所以随着层数的加深，导数的数值会以指数形式下降，这就导致了反向传播越往前，效果越微弱，致使前后层训练的步调不一致，尤其是靠近输入层的参数，几乎无法得到优化。

相应的，如果激活函数的导数大于 1，则在反向传播中导数值会以指数形式上升，造成梯度爆炸。不过这种情况很少见。

这个问题到现在应该说还没有彻底解决，但经过前人的努力，也有了许多不错的方案。其中一个效果比较好的是用其它激活函数来替代 $\sigma(x)$，而在所有替代品中，*relu* 函数脱颖而出，有效地缓解了梯度消失，并得到广泛认可。它的表达式如下：

$$relu(x) = max(0, x) \tag{10}$$

4.2 局部极值点

从图2中我们可以看出，一个函数很可能有多个极值点，但实际需要的只有一个全局的最值点。然而，梯度下降法只能让我们找到某一个极值点，却无法判断这个点是否为全局最优。

针对这个问题也有许多解决方式被提出，不过很难说哪种方式可以彻底解决它。一种常见做法是用不同的权重来初始化整个网络，尤其是用别人已经在某些相近数据集上训练好的参数。这样做可以使得我们有较大的概率将整个网络参数初始化在最优点附近，从而避免落入其它局部极值点。

5 总结：一点个人理解

通过梳理神经网络的相关知识，个人感觉深度学习（主要是有监督学习）更多地是一个极大似然估计的过程。所谓极大似然估计，就是在已知实验结果的情况下，反推最有可能导致这种结果的所有参数。换句话说，就是要找出最优参数，使得似然函数输出的结果能够最大程度接近已知的结果。而梯度下降法所做的，正是这样的事情：不断优化参数，使得网络的输出尽可能接近标记的值。所以在我看来，所谓深度学习，就是极大似然估计的一种实现；而深度神经网络，其实就是一个及其复杂的似然函数。

参考文献

[1] Rosenblatt F. Two theorems of statistical separability in the perceptron. 1958. 2

[2] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006. 2

[3] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133, 1943. 2

[4] Marvin Minsky and Seymour Papert. An introduction to computational geometry. Cambridge tiass., HIT, 1969. 2

[5] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. 323(6088):399–421, 1986. 2

[6] 付文博，孙涛，梁藉，闫宝伟，范福新. 深度学习原理及应用综述. 计算机科学, (s1), 2018. 2

[7] 余凯，贾磊，陈雨强，徐伟. 深度学习的昨天、今天和明天. 计算机研究与发展, 50(9):1799–1804, 2013. 2