1 深度卷积神经网络

1.1 引言

人工神经网络[?] 是一种通过模仿大脑神经元行为进行信息处理的数学模型，但由于其无法承受大规模的参数和训练样本并且具有泛化能力差等问题。Fukushima[?] 于 1982 年首次提出了卷积神经网络模型，Lecun 等人对神经网络传统算法在训练上面临的计算复杂度高等问题进行了改进，提出了基于梯度下降的优化算法[?] 和 BP 算法[?]。2003 年，Simard 对卷积神经网络进行了简化[?]。Hinton 在 2006 年的两篇文章[?, ?] 可以作为深度学习 (Deep Learning,DL) 正式提出的里程碑。Ciren 等人[?] 在 2011 年对神经网络进行改造使其可以通过 GPU 进行训练计算，并在大量的图像数据集对深度学习算法进行了测试，并且都取得了最好的成绩。其在工业界以及学术界掀起了巨大的浪潮，被应用于语音识别[?]、图像识别[?] 和自然语音处理[?] 等各种方面。专门针对于深度学习的芯片不断问世，例如谷歌公司的 TPU[?]，大大提高了深度学习的应用场景。深度神经网络模型是一种非常强大的深度学习模型，他同时可以处理有监督和无监督学习任务。并且随着科技的发展，数据量越来越大，计算机并行能力也有了很大的提高。针对于海量数据，简单的线性模型由于无法充分利用计算能力，不再适用，可以预见在将来会有越来越多的工作应用到深度学习。目前，其内涵已经超出了传统的多层神经网络，甚至机器学习的范畴，逐渐朝着人工智能的方向快速发展[?]。

1982 年，Kunihiko 等人[?] 首次将卷积神经网络模型的概念引入深度学习。后来许多学者在实践中对 CNN 的发展和理论分析作出了重大贡献。1989 年，LeCun 等人将基于梯度的学习方法[?] 和 BP 算法[?] 引入到 CNN。2003 年，Behnke 写了一本总结 CNN[?] 的书。同年，Simard 等人[?] 对卷积神经网络进行了简化。2011 年，Ciren 等[?] 进一步改进 CNN 并实现了 GPU 版本，使得 CNN 的训练识别速度有了巨大的提升，并使用 CNN 框架对多个图像数据库进行实验，并取得了最佳成果。

本章安排如下：2.2 节对深度学习的基本分类进行了介绍，2.3 节详细地阐述了传统人工神经网络结构的基本定义和神经元之间信息传递的过程，2.4 节对本文主要利用的深度卷积神经网络的特性进行了介绍，2.5 节进行本章总结。

1.2 基本分类

传统上可以把深度学习分为卷积神经网络（Convolutional Neural Networks，CNN）、递归神经网络（Recurrent Neural Networks，RNN）、长短时记忆网络（Long short-term memory, LSTM）、深度信念网络（Deep Belief NetWorks，DBN）、自编码器 (AutoEncoder)、稀疏编码 (Sparse Coding)、限制波尔兹曼机 (Restricted Boltzmann Machine, RBM) 等。

其中卷积神经网络是最流行的一种深度学习模型，通过使用卷积层极大地减少了中间层的参数数目，使学习效率更高并较少过拟合，同时卷积操作独有的局部感受野（local receptive fields）、共享权重（shared weights）和池化（pooling）三种特性也是处理序列元素分类识别的很重要的一点，权重共享策略减少了需要训练的参数，相同的权重可以让卷积核不受信号位置的影响来检测信号的特性，使得训练出来的模型泛化能力更强；池化运算可以降低网络的空间分辨率，从而消除信号的微小偏移和扭曲。

递归神经网络是一种包含循环的，允许信息持久化的神经网络模型。传统的前馈神经网络中，单独的输入完全确定了余下层的神经元的激活值。而对于递归神经网络，隐藏层和输出层的神经元的激活值不仅由当前的网络输入决定，而且包含了前面的输入的影响。长短时记忆网络是一种特殊的递归神经网络，主要用于解决递归神经网络前期模型难以训练的问题。其通过刻意设计的单元结构，在递归神经网络的基础上添加了元胞状态（cell state）用来保存长期的状态，然后通过门函数来控制此长期状态。

深度信念网络是一个概率生成模型，是由多个限制玻尔兹曼机组成，这些网络被“限制”为一个可见层和一个隐藏层，层间存在连接，但是层内的单元间不存在连接。隐藏单元被训练来捕捉在可见层表现出来的高阶数据的相关性。

通过对于以上几种最常用的深度学习方法的介绍，我们可以发现卷积神经网络是最适合处理本文这种静态类型的数据，循环或者说是不同时刻的输入对于地海杂波类型的识别并没有提高，故递归神经网络和长短时记忆网络显然不适合本问题。另一方面，深度信念网络的生成模型并不关心不同类别之间的最优分类面的位置，故其用于分类问题时，分类精度没有判别模型高。且其学习的是数据的联合分布，相比其他算法具有更高的复杂性。

1.3 传统人工神经网络结构

人工神经网络是一种模拟大脑神经系统的算法，其可以从海量的训练样本中学习到一个权重函数，用来进行模式识别、分类等。神经网络主要是利用将许多个单一神经元 (也称作感知器) 联结在一起，一个神经元的输出就可以是另一个神经元的输入，形成一

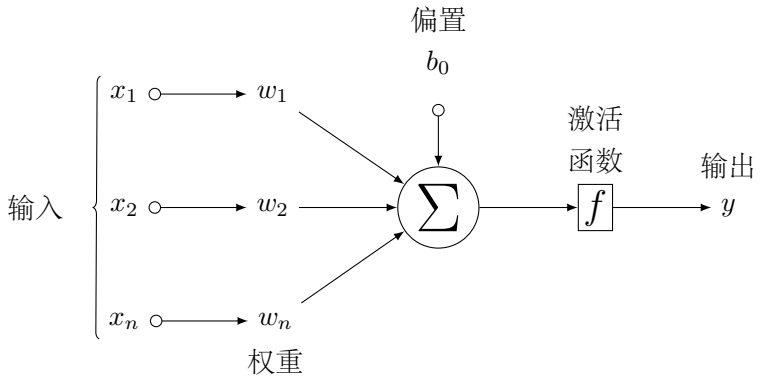

图 1-1　神经元

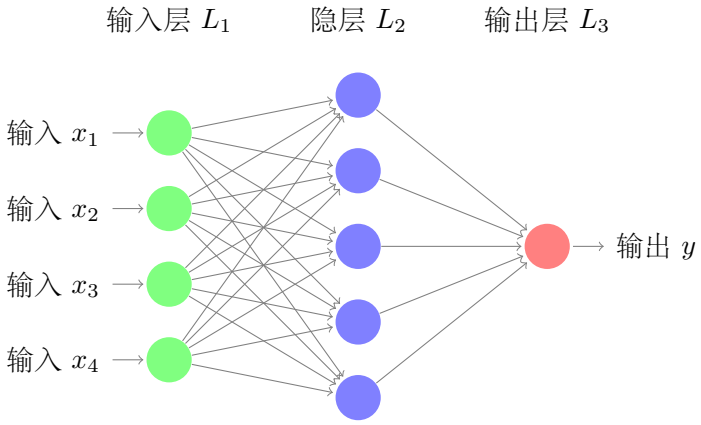

图 1-2　神经网络

个有向无环的网络结构。首先介绍单一神经元的结构，如图 1-1所示，神经元一般具有多个输入 $x = [x_1, x_2, \dots\ x_n]$，这些输入可以取 0 和 1 中的任意值。神经元对于每一个输入有权重 $w = [w_1, w_2, \dots, w_n]$ 和一个总的偏置 b_0 ，其输出为:

$$y = f(w \cdot x + b) \tag{1-1}$$

这里的 f 为该神经元的激活函数，我们在后面进行描述。

上述多个神经元分层互联即可以形成神经网络。图 1-2是一个简单的神经网络，其中圆圈表示神经网络的节点，也即神经元。对于一个具有 n_l 层的神经网络，如将第 l 层用 L_l 表示，则有最左边的一层 L_1 为输入层，最右的一层 L_{n_l} 为输出层。由于不能在训练样本集中观测到中间所有节点的值，故将这些层称做隐藏层。本例中网络的层数 $n_l = 3$，输入层具有 4 个输入单元，隐层具有 5 个隐藏单元，输出层只有一个输出单元。

根据输入 x_i 求解输出 y 的主要过程为前向传播。我们用 $w_{ij}^{(l)}$ 表示第 l 层的第 j 单元与第 $l+1$ 层的第 i 单元之间的权重，$b_i^{(l)}$ 是第 $l+1$ 层中第 i 单元的偏置项。因此，$w^{(l)}$ 为第 l 层权重组成的向量，$w^{(1)} \in \Re^{4\times3}$ ，$w^{(2)} \in \Re^{1\times5}$ 。根据图 1-1和公式 1-1可以

知道每个神经元节点的输出是经过激活函数的激活值，可以用 $y_i^{(l)}$ 表示第 l 层的第 i 单元的输出值。当 $l=1$ 时，$y_i^{(1)}$ 为就是第 i 个输入值，也即有 $y_i^{(1)}=x_i$。当权重已知的时候，我们就可以根据公式 1-1求解此传播过程，进而求解最终的输出值 y。从 L_1 层到 L_2 层的过程为：

$$y_1^{(2)}=f(w_{11}^{(1)}x_1+w_{12}^{(1)}x_2+w_{13}^{(1)}x_3+w_{14}^{(1)}x_4+b_1^{(1)}) \tag{1-2}$$

$$y_2^{(2)}=f(w_{21}^{(1)}x_1+w_{22}^{(1)}x_2+w_{23}^{(1)}x_3+w_{24}^{(1)}x_4+b_2^{(1)}) \tag{1-3}$$

$$y_3^{(2)}=f(w_{31}^{(1)}x_1+w_{32}^{(1)}x_2+w_{33}^{(1)}x_3+w_{34}^{(1)}x_4+b_3^{(1)}) \tag{1-4}$$

$$y_4^{(2)}=f(w_{41}^{(1)}x_1+w_{42}^{(1)}x_2+w_{43}^{(1)}x_3+w_{44}^{(1)}x_4+b_4^{(1)}) \tag{1-5}$$

$$y_5^{(2)}=f(w_{51}^{(1)}x_1+w_{52}^{(1)}x_2+w_{53}^{(1)}x_3+w_{54}^{(1)}x_4+b_5^{(1)}) \tag{1-6}$$

根据同样的计算过程可以得到：

$$y=y_1^{(3)}=f(w_{11}^{(2)}y_1^{(2)}+w_{12}^{(2)}y_2^{(2)}+w_{13}^{(2)}y_3^{(2)}+w_{14}^{(2)}y_4^{(2)}+w_{15}^{(2)}y_5^{(2)}+b_1^{(2)}) \tag{1-7}$$

我们只需要把公式 1-6代入 1-7就可以计算得到输入 x_i 与输出 y 之间的关系。为了使表达更清晰，我们用 $z_i^{(l)}$ 表示第 l 层第 i 单元输入加权和，则有：

$$z_i^{(l)}=\sum_{j=1}^{n}w_{ij}^{(l-1)}x_j+b_i^{(l)} \tag{1-8}$$

因此公式 1-6变为

$$y_i^{(l)}=f(z_i^{(l)}) \tag{1-9}$$

则整个前向传播过程可以简化为：

$$z^{(l+1)}=w^{(l)}y^{(l)}+b^{(l)} \tag{1-10}$$

$$h^{(l+1)}=f(z^{(l+1)}) \tag{1-11}$$

其中，$b^{(l)}$ 为第 $l+1$ 层偏置项组成的向量，$h^{(l)}$ 为第 l 层各个单元输出值组成的向量。

根据上述计算过程，对于一个具有 n_l 层的前馈神经网络（在网络中没有闭环或者回路），其第 1 层是输入层，第 n_l 层是输出层，中间的所有隐藏层 L_l 均与下一层 L_{l+1} 全连接。我们只需要逐个计算每一个层中的节点的输出值，然后据此得到下一层的输出值。对于一个复杂的神经网络一般会具有多个隐藏层以及输出层中可以有多个输出单元。图 1-3的神经网络有三层隐藏层：L_2、L_3 及 L_4 ，输出层 L_5 有三个输出单元。

1.4 深度卷积神经网络

卷积神经网络是深度学习中的一种重要算法，在分类等很多领域具有很大的优势，该方法经常被用于图像识别、语音识别等问题。卷积神经网络是一种特殊的人工神经网

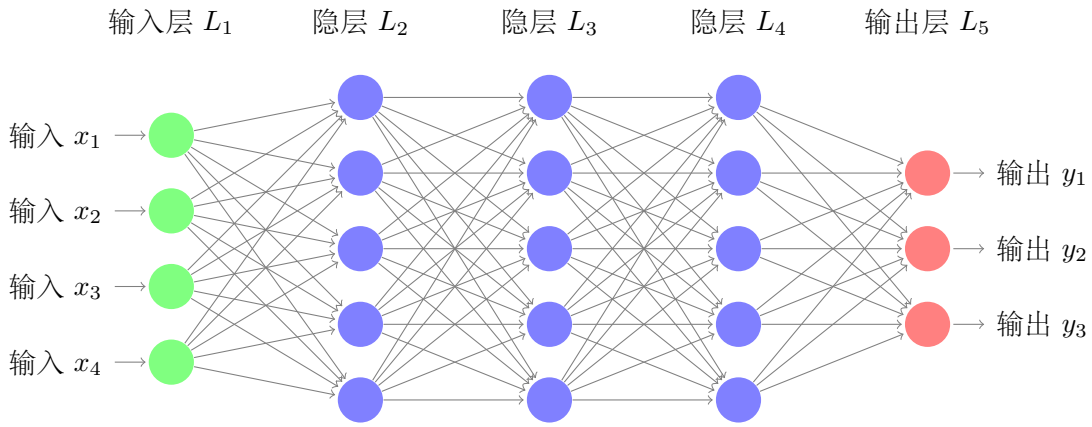

图 1-3　具有多个输出单元的神经网络示意图

络，由多对卷积层 (Convolutional layer) 和池化层 (Pooling layer)，以及全连接层 (Full-connected layer) 组成。图 1-4为一个典型的卷积神经网络，最左边的为输入层，其中的神经元为输入神经元。最右边的输出层包含有一个或者多个输出神经元，其余的是多组卷积和池化层，有限数量的完全连接的隐藏层。

网络的前向传播主要是卷积和池化操作，其网络参数的更新采用误差反向传播 (Error Backpropagation) 算法。相比于传统人工网络，卷积神经网络有以下几个特点：

- 输入为原始数据，对原始数据进行卷积运算，然后提取从最后一步生成的卷积数据的特征，丰富了识别中使用的特征。
- 多个卷积核可以利用输入数据中的局部结构，将整个输入空间划分成很小的隐藏单元。将各个隐藏单元的权重构建得到的卷积核作用于整个输入空间，从而得到特征向量。利用这种机制，我们不仅大大减少了参数数量同时提高了数据的平移不变性。
- 具有超过两个的隐藏层以及可以逐层初始化模型的参数，通过更多的隐藏层可以更好地对事物的特征进行学习表达。

1.4.1　卷积

深度卷积神经网络在特征提取过程中一个主要操作为卷积，在前向计算过程中，对于输入的一定区域的隐藏层的特征向量 x 和滤波器（权重）w 点乘后得到新的特征向量，然后滑过一个个滤波器，以产生输出特征向量。权重实际上是一个四维张量，第一维 F 是卷积输入层的特征向量的数量，另一维 F_p 是卷积输出层的特征向量的数量，另外两维是在宽度和高度方向给出的卷积核的大小，也称作感受野。

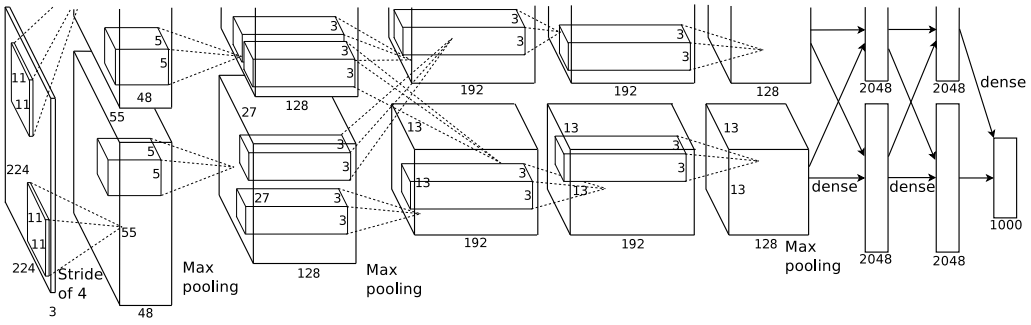

图 1-4　AlexNet 卷积神经网络模型[?]

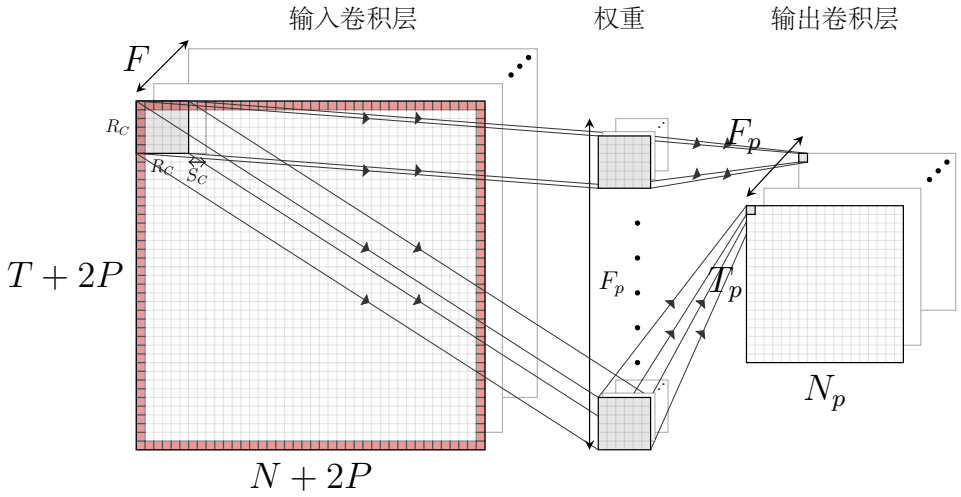

图 1-5　卷积示意图

对于传统的神经网络，每个输入元素会连接到每个隐藏神经元。然而，在卷积神经网络中，我们只是把输入数据进行小的、局部区域的连接，也即第一个隐藏层中的每个神经元会连接到一个输入神经元的一个区域。这个输入向量的区域被称为隐藏神经元的局部感受野。它是输入向量上的一个小窗口，对于每个连接学习一个权重而隐藏神经元同时也学习一个总的偏置。通过在整个输入频谱数据上交叉移动局部感受野，可以构建起第一个隐藏层。

感受野允许用一个子集来代替整个输入数据。它的目的是在输入数据中搜索相似的模式，而不管模式在哪里，也即平移不变性。输出图像的宽度和高度也由步长（stride）确定，步长为在进行再次应用卷积运算之前在垂直、水平方向上滑动的像素的数目。我们以图 1-5这个对一个图像进行二维卷积的操作来介绍卷积运算。

其中 R_C 为卷积感受野的大小，S_C 为卷积的步长。根据输入高度 T 和宽度 N，可以计算输出图像的高度和宽度：

$$N_p = \frac{N+2P-R_C}{S_C}+1\ , \qquad T_p = \frac{T+2P-R_C}{S_C}+1 \tag{1-12}$$

通常对输入数据进行边界扩充（padding）来使得输入特征图的宽度和高度满足 $N = N_p = T = T_p$，这样当前进步长 $S_C = 1$ 时，可以得到

$$P = \frac{R_C - 1}{2} \tag{1-13}$$

对于一个给定的层 L_n，卷积运算的计算过程如下：

$$y_{f\,l\,m}^{(\nu)} = \sum_{f'=0}^{F_\nu - 1} \sum_{j=0}^{R_C - 1} \sum_{k=0}^{R_C - 1} w_{f'\,j\,k}^{(o)f} h_{f'\,S_C l + j\,S_C m + k}^{(\nu)} \tag{1-14}$$

其中 o 表示网络中的第 $o+1$ 个卷积核。$\nu \in [0, N-1]$ 表示网络中的第 ν 个隐层，$f \in [0, F_{\nu+1} - 1]$ 为输入卷积层的第 f 个特征图，$l \in [0, N_{\nu+1} - 1]$ 为输入卷积层的第 v 个特征图的宽度单位，$m \in [0, T_{\nu+1} - 1]$ 为输入卷积层的第 v 个特征图的高度单位。因此可以得到 $S_C l + j \in [0, N_\nu - 1]$，$S_C l + m \in [0, T_\nu - 1]$。于是可以通过添加激活函数计算隐藏单元的输出。考虑到边界填充可以得到下式

$$h_{f\,l+P\,m+P}^{(\nu+1)} = f\left(y_{f\,l\,m}^{(\nu)}\right) \tag{1-15}$$

每个滤波器只关心数据的部分特征，当出现它学习到的特征的时候，就会呈现激活态。

1.4.2 共享权重和偏置

上面已经说过对于每个隐藏神经元具有一个偏置和连接到它的局部感受野的权重，同时对于该层的所有的隐藏神经元中每一个使用相同的权重和偏置。也即对第 j 个隐藏神经元，输出为：

$$h_j = f(b + \sum_{m=1}^{M} w_m y_{j+m}) \tag{1-16}$$

这里 f 是神经元的激活函数，b 是偏置的共享值，w_m 是一个共享权重的 $1 \times M$ 向量，y_k 表示位置 k 的输入激活值。这意味着第一个隐藏层的所有神经元检测完全相同的特征，只是在输入数据的不同位置，因此卷积网络可以很好地适应数据偏移的情况。

1.4.3 池化

我们在通过卷积获得了特征之后，下一步我们希望利用这些特征去做分类。理论上讲，人们可以用所有提取得到的特征去训练分类器，例如 Softmax 分类器（多分类的逻辑回归分类器），但这样做面临着计算量的挑战，除此以外过多的特征向量，也容易导致过拟合。由于本文的雷达信号具有一种“静态性”属性，在一个数据区域有用的特征极有可能在另一个区域同样适用。因此，为了描述数据量较多的数据，一个很自然的想法就是对不同位置的特征进行聚合统计，例如，可以计算频谱数据上一段频率范围内的某

个特征的最大值 (或平均值)。这些经过采样的统计特征相比使用所有提取得到的特征不仅具有低得多的维度，同时还不容易过拟合，在一定程度上会改善结果。这种聚合的操作称为池化。池化操作是卷积神经网络中一种常用的用来对数据进行降维处理的方法。一般有平均池化和最大池化，主要过程为选取输入特征向量的 F 一个池化感受野 R_P 和步长 S_P 选取其最大或者平均元素来代表该区域，得到输出特征向量 $F_p = F$，其中宽度 $N_p < N$，高度 $T_p < T$。在池化操作过程中，我们一般不考虑输入隐藏层的边界扩充值，因此在公式 1-17中存在 $+P$ 这一部分。

对于给定的第 ν 个平均池化操作的公式如下:

$$y_{f\,l\,m}^{(\nu)} = \sum_{j,k=0}^{R_P-1} h_{f\,S_P l+j+P\,S_P m+k+P}^{(\nu)} \tag{1-17}$$

最大池化的公式为:

$$y_{f\,l\,m}^{(\nu)} = \max_{j,k=0}^{R_P-1} h_{f\,S_P l+j+P\,S_P m+k+P}^{(\nu)} \tag{1-18}$$

其中，$\nu \in [0, N-1]$ 表示网络中的第 ν 个隐层，$f \in [0, F_{\nu+1}-1]$, $l \in [0, N_{\nu+1}-1]$ 以及 $m \in [0, T_{\nu+1}-1]$。因此有 $S_P l + j \in [0, N_\nu - 1]$，$S_P l + m \in [0, T_\nu - 1]$。最大池化是最常用的一种池化方式，本文所用的神经网络结构均使用最大池化。对于第 t 次批采样，我们用 $j_{flm}^{(t)(p)}$, $k_{flm}^{(t)(p)}$ 来表示在 l, m 处特征向量图的最大值，则公式 1-18变为下式:

$$h_{f\,l+P\,m+P}^{(t)(\nu+1)} = y_{f\,l\,m}^{(t)(\nu)} = h_{f\,S_P l+j_{flm}^{(t)(p)}+P\,S_P m+k_{flm}^{(t)(p)}+P}^{(t)(\nu)} \tag{1-19}$$

深度卷积神经结构具有过拟合的自然趋势，虽然可以通过权重共享来减少参数的数量。但是由于大多数情况下，估计集的数量比训练集大一个数量级，使得神经网络模型的泛化能力不足。在每个训练迭代中，每个隐藏单元以预定概率被随机删除，删除后学习过程继续。这些被称作 dropout 的随机扰动有效地防止了神经网络学习过程的依赖关系，并在隐藏的单元之间创建了复杂的关系。这样增加了网络模型的复杂度，从而提高深度神经网络模型的泛化能力。

1.4.4 激活函数

在公式 1-7中，$f(x)$ 被用于每一个神经元节点，因此激活函数的选择是构建卷积神经网络模型的一个非常重要的方面。我们在本节介绍几种常用的激活函数。

sigmoid 激活函数

在传统人工神经网络中常用的激活函数为 sigmoid 激活函数（S 型激活函数），其取值范围为 $(0, 1)$，其定义如下式:

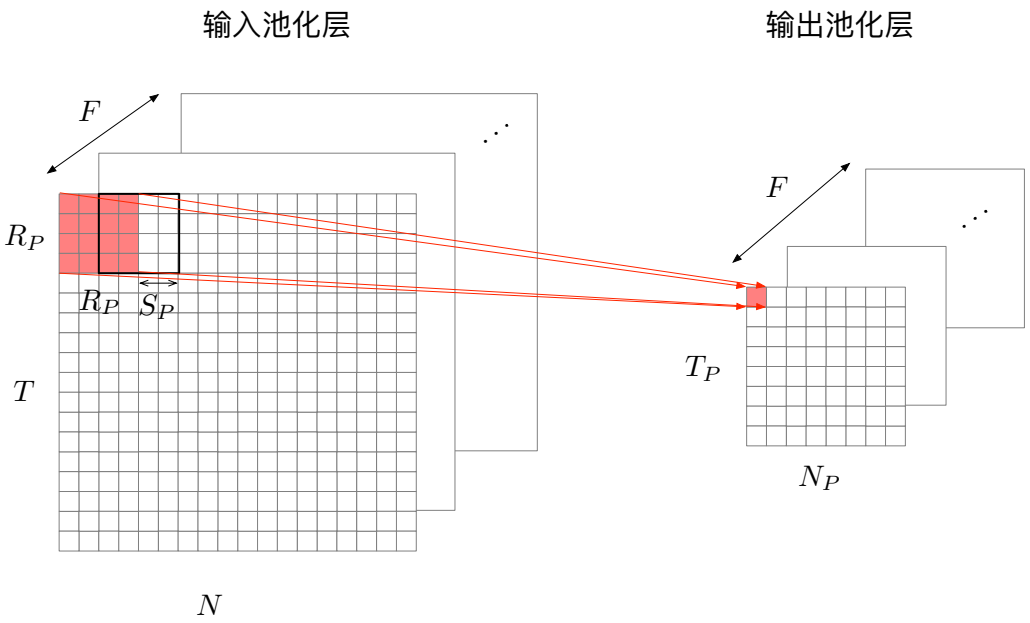

图 1-6　池化示意图

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{1-20}$$

对其求导可以得到：

$$\sigma'(x) = \sigma(x)\,(1 - \sigma(x)) \tag{1-21}$$

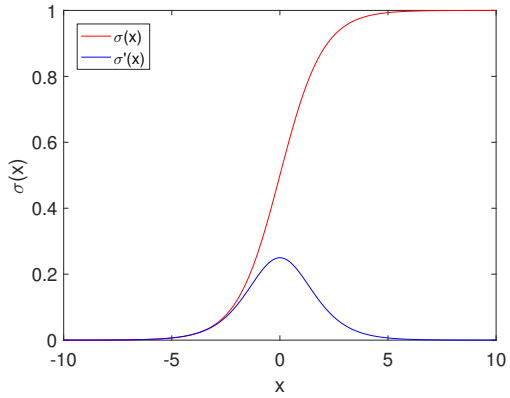

图 1-7　sigmoid 激活函数及其导数

tanh 激活函数

tanh 激活函数的取值范围为 $(-1, 1)$，其定义如下：

$$f(x) = \tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{1-22}$$

对其求导可以得到：

$$\tanh'(x) = 1 - \tanh^2(x) \tag{1-23}$$

ReLU 激活函数

ReLU(Rectified Linear Unit) 的取值范围为 $[0, +\infty)$，其定义如下式：

$$f(x) = \mathrm{ReLU}(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{1-24}$$

对其求导可以得到：

$$\mathrm{ReLU}'(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{1-25}$$

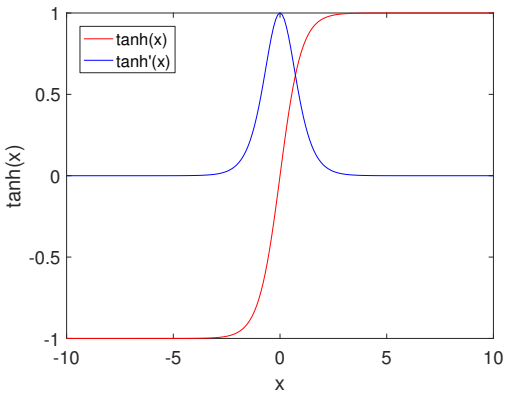

图 1-8　tanh2 激活函数及其导数

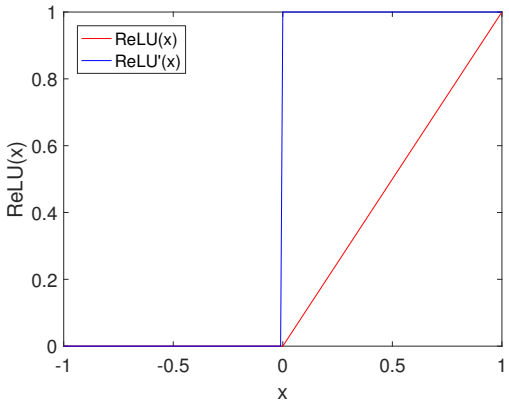

图 1-9　ReLU 激活函数及其导数

ReLU 具有优于传统激活函数的几个优点：更快的计算速度和更有效的梯度传播（它们不像 S 形单元那样饱和），生物学可能性和稀疏激活结构。尽管它结构简单，但仍然保持足够的辨别性质。其缺点之一是随机权重的初始状态，多个单位可能过早地落入死区（零输出的恒定梯度）。因此，当与整个连接层进行全连接时，Sigmod 激活函数的效果更好。

1.4.5　神经网络的学习算法

参照上文的符号定义，用符号 x 表示训练输入的向量集，w 表示所有的网络中权重的集合，b 是所有的偏置，用 $y = f(w \cdot x + b)$ 表示对应的期望输出。学习算法的主要目的是找到一个权重 w 和偏置 b 使得网络的输出 y 可以拟合所有的训练输入 x。为此

可以定义一个损失函数或代价函数 (loss function)$C(w,b)$。针对于不同的问题可以选取不同的损失函数，公式 1-26利用均方误差函数作为损失函数。

$$C(w,b) \equiv \frac{1}{2n}\sum_{x}||y-a||^2 \tag{1-26}$$

其中，n 是训练输入数据的个数，a 表示当输入为 x 时的输出向量。

从定义可以看出，$C(w,b)$ 越小说明分类越准确，那么训练神经网络的目的就是找到使得损失函数 $C(w,b)$ 最小的权重和偏置。为了求解该问题，将上述问题一般化，原问题变为最小化任意的具有 m 个变量的多元实值函数 $C(v)$，$v = v_1, v_2, \ldots, v_m$。这种具有大量变量的函数的解析解的求解是极其复杂的，一个比较合理的思路为利用数值计算的方法求取其极值点。每次对于 C 中的自变量添加一个微小的变化 Δv，根据此变化反映出来的 C 的变化 ΔC 更新下次的微小变化，从而使得 C 可以持续减小。对 C 中自变量的变化 $\Delta v = (\Delta v_1, \ldots, \Delta v_m)^T$ ，ΔC 将会变为

$$\Delta C \approx \nabla C \cdot \Delta v \tag{1-27}$$

，这里的梯度 ∇C 定义如下：

$$\nabla C \equiv (\frac{\partial C}{\partial v_1}, \ldots, \frac{\partial C}{\partial v_m})^T \tag{1-28}$$

其把 v 的变化关联为 C 的变化，假设选取 $\Delta v = -\eta \nabla C$，其中的 η 是学习率，一般取一个很小的正数，则公式 1-27变为：

$$\Delta C \approx -\eta \nabla C \cdot \nabla C = -\eta ||\nabla C||^2 \leq 0 \tag{1-29}$$

也即如果利用更新规则 $v \to v' = v - \eta \nabla C$，$C$ 会持续减小，此更新规则即为最基本的学习算法，梯度下降算法。图 1-10展示了梯度下降的示意图，其跟随当前位置最陡的下降的方向也即负梯度，其中学习率 η 描述在每个迭代步骤中下降的步长。可以根据选择不同的损失函数 C 或者通过计算来完成学习率的选择等各种技术对学习算法进行优化。根据梯度下降的思想，我们可以结合卷积神经网络中的反向传播，获得对于神经卷积神经网路的训练流程图，如图 1-11所示。

1.5 小结

本章介绍了深度学习的相关基础。首先是其基本分类，然后针对于本文利用的深度卷积神经网络，介绍了传统的神经元的结构和卷积神经网络基于此的改进。在最后讨论了深度卷积神经网络的学习算法。

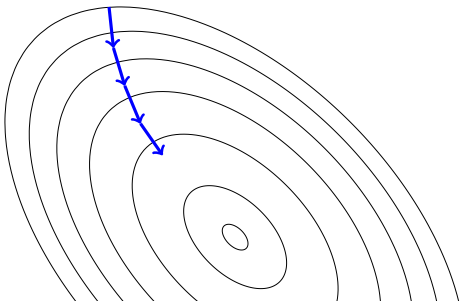

图 1-10　梯度下降示意图

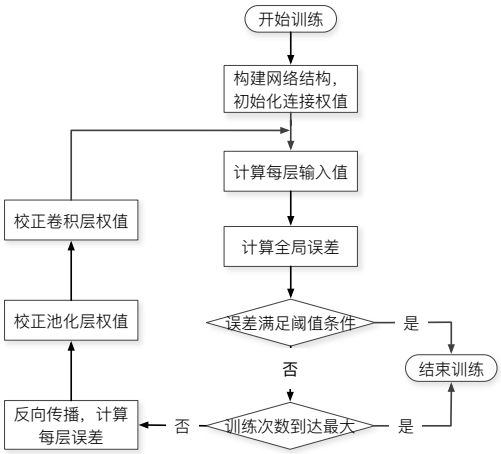

图 1-11　神经网络学习算法流程图

参考文献

[1] Hebb DO. The organization of behavior: A neuropsychological theory[M]. Psychology Press. 2005.

[2] Fukushima K, Miyake S. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition[C]. In Competition and cooperation in neural netsSpringer. 1982, :267–285.

[3] LeCun Y et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE. 1998, 86(11):2278–2324.

[4] LeCun Y et al. Backpropagation applied to handwritten zip code recognition[J]. Neural computation. 1989, 1(4):541–551.

[5] Simard PY et al. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis.[C]. In ICDAR. Citeseervol. 32003:958–962.

[6] Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks[J]. science. 2006, 313(5786):504–507.

[7] Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets[J]. Neural computation. 2006, 18(7):1527–1554.

[8] Ciresan DC et al. Flexible, high performance convolutional neural networks for image classification[C]. In Twenty-Second International Joint Conference on Artificial Intelligence. 2011.

[9] Hinton G et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups[J]. Signal Processing Magazine, IEEE. 2012, 29(6):82–97.

[10] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks[C]. In Advances in neural information processing systems. 2012:1097–1105.

[11] Collobert R et al. Natural language processing (almost) from scratch[J]. Journal of Machine Learning Research. 2011, 12(Aug):2493–2537.

[12] Jouppi NP et al. In-datacenter performance analysis of a tensor processing unit[J]. arXiv preprint arXiv:170404760. 2017.

[13] Silver D et al. Mastering the game of go without human knowledge[J]. Nature. 2017, 550(7676):354–359.

[14] Behnke S. Hierarchical neural networks for image interpretation[M]vol. 2766. Springer Science & Business Media. 2003.