第 3 章 深度学习网络

深度学习网络是一种基于人工神经网络的机器学习方法家族，其核心思想是通过多层次的表征学习来实现数据的抽象和复杂模式的发现。深度学习包括监督学习、半监督学习和无监督学习等多种方法，其中监督学习是较常用的方法之一。在监督学习中，模型从已标记的数据中学习，通过优化模型参数以最小化预测误差来进行训练。深度学习架构包括深度神经网络、深度信念网络、深度强化学习、递归神经网络、卷积神经网络和转换器等。这些架构已广泛应用于计算机视觉、语音识别、自然语言处理、机器翻译、生物信息学、药物设计、医学图像分析、气候科学、材料检验和棋盘游戏程序等领域。在这些应用程序中，深度学习模型已经表现出与人类专家相当甚至超过人类专家的表现。与传统的线性感知器相比，深度学习模型具有更强的表达能力，能够学习非线性决策边界。深度神经网络特别是多层感知器通过在不同层之间使用逻辑函数来实现非线性性质，从而能够学习更复杂的模式。使用无限宽度隐含层的深度置信网络也能提高模型的表达能力，并且理论结果表明，深度置信网络可以任意精确地逼近任意连续函数。因此，深度学习模型具有极强的表达能力，可以应用于各种任务。

深度学习算法的一个重要特点就是它可以通过学习输入数据中的未知结构来提供更好的表示。这种表示通常是通过多层神经网络来实现的，其中每个层次都使用较低层次的特征来表示更高层次的抽象特征。这种层次结构可以帮助深度学习模型对数据进行更深入的理解和建模，从而提高模型的准确性和泛化能力。深度学习策略旨在学习特征层次结构，其中较高层次结构的特征是由较低层次特征的组合创建的。深度学习是一种现代的机器学习方法，它通过使用大量的层来学习数据的复杂表达。这种方法允许网络学习到数据的深层特征，这些特征可能在其他方法中难以捕捉。在深度学习中，网络结构可以是异构的，即允许使用不同类型的层。例如，深度学习网络可以包括全连接层、卷积层、循环层、局部响应归一化层等。这样做能够提高网络的效率和可训练性。其中，卷积层在图像识别中很常用，它利用图像局部空间结构特性进行特征提取。循环层在序列问题上常用，如自然语言处理，能够更好捕获序列数据的长期相关性。在可理解性方面，深度学习网络结构有时与生物学知识相关，但经常是基于统计学习理论和数据驱动的方法来设计和优化网络结构，而不是依赖于生物学知识。因此可以说，在深度学习中，网络结构设计是一个结构化的部分，需要考虑模型的效率、可训练性和可理解性，根据具体任务进行调整。

深度学习架构和算法已经在计算机视觉和模式识别等领域取得了惊人的进步。顺着这个趋势，最近的自然语言处理目前越来越专注于最近的深度学习策略领域。自然语言处理受益于最近深度学习架构的复兴，因为它们有可能在不需要工程特征的情况下达到高精度。一方面，深度学习算法需要比传统机器学习算法更多的训练数据，即至少数百万个标记示

例；另一方面，支持向量机（support vector machines，SVM）、朴素贝叶斯（naive bayes，NB）、决策树（decision tree），逻辑回归（logistic regression）等传统机器学习算法已经在许多应用领域中取得了非常好的结果，但是它们也存在着一些瓶颈。其中一个重要的瓶颈就是在处理高维稀疏数据时，这些算法的性能会受到严重的影响，特别是当数据维度过高时传统机器学习算法的效率和准确性会明显降低。另外，这些传统算法也有很大的局限性只能应用于特定类型的问题，如支持向量机、贝叶斯、决策树主要应用于分类问题。这些算法在解决大数据问题上可能会有一定局限性，即添加更多训练数据并不能提高其准确性，相比之下深度学习分类器会随着人们提供的数据越多而变得更好。目前，深度学习架构包括许多种不同类型的网络结构，如图 3-1 所示。深度学习架构分为有监督和无监督学习，包括卷积神经网络、循环神经网络（recurrent neural network，RNN）、长短期记忆/门控循环单元（gated recurrent unit/long short-term memory，GRU/LSTM）、自组织映射（self-organizing map，SOM）、自动编码器（autoencoder，AE）和受限玻尔兹曼机（restricted boltzmann machine，RBM），还包括深度信念网络和深度堆叠网络（deep stacked network，DSN）等[①]。这些架构可以单独使用或结合使用以解决各种不同类型的问题。

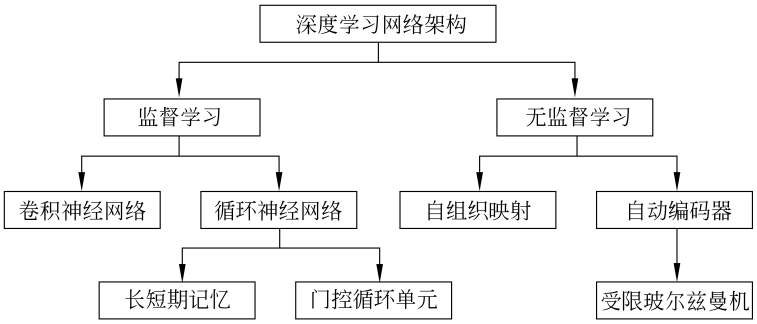

图 3-1　深度学习网络架构

3.1　深度的定义

大多数现代深度学习模型都基于人工神经网络，特别是卷积神经网络，尽管它们也可以包括命题公式[②]或在深度生成模型中按层组织的潜在变量，如深度信念网络和深度玻尔兹曼中的结点机器[③]。在深度学习中，每个级别都学习将其输入数据转换为稍微更抽象和复合的表示形式。在图像识别应用程序中，原始输入可能是像素矩阵；第一表示层可以抽象像

① Deep learning architectures, By Samaya Madhavan, M. Tim Jones, Updated January 24, 2021 | Published September 7, 2017.

② 命题公式是数理逻辑中的基本概念之一，用于描述命题之间的关系和逻辑结构。在深度学习中，命题公式可以用来表示复杂的逻辑规则或约束条件，从而帮助模型更好地处理输入数据。例如，命题公式可以用来表示某个输入数据是否属于某个类别，或者表示某些特征之间的约束关系等。

③ 结点机器是另一种常见的组件，它在深度学习中用于表示和学习数据的潜在结构或特征。结点机器通常由许多结点或神经元组成，每个结点或神经元都有一些权重或参数，用于计算结点或神经元的输出。不同类型的结点机器有不同的结构和学习算法。例如，深度信念网络和深度玻尔兹曼机就是两种常见的结点机器。这些结点机器通常用于无监督学习或半监督学习，用于学习数据中的特征和结构，从而提高模型的性能。

素和编码边缘;第二层可以组合和编码边的排列;第三层可以编码鼻子和眼睛;第四层可以识别图像中包含人脸。重要的是深度学习过程可以自己学习将哪些特征最佳地放置在哪个级别。这并不能消除手动调整的需要。例如,不同数量的层和层大小可以提供不同程度的抽象。

深度学习中的"深度"一词指的是数据经过多少层变换,更准确地说,是指深度学习系统具有相当大的置信分配路径(credit assignment path,CAP)。置信分配路径指的是对于一个错误的输出训练网络所需更新的权重和偏差参数所需遍历的层数。置信分配路径是从输入到输出的转换链,描述了输入和输出之间潜在的因果关系。对于非循环的前馈神经网络,置信分配路径的深度是网络的深度,是隐含层的数量加 1(因为输出层也被参数化);对于循环的循环神经网络,其中一个信号可能多次通过一个层传播,置信分配路径深度可能是无限的。没有普遍认可的深度阈值来区分浅层学习和深度学习,但大多数研究人员认为深度学习涉及高于 2 的置信分配路径[①]。深度为 2 的置信分配路径已被证明是一种通用逼近器,因为它可以模拟任何函数,除此之外更多的层不会增加网络的函数逼近能力。深度模型(CAP>2)能够提取比浅层模型更好的特征,因此额外的层有助于有效地学习特征。在实际应用中,深度学习网络的层数可能在几十层甚至更多,然而确切的层数并不重要,重要的是网络是否有能力学习和表示足够复杂的特征。

基于卷积神经网络的深度学习架构可以用贪心逐层训练的方法来构建,而在这个过程中逐层训练可以帮助我们理清不同层的抽象,并选择出哪些特征可以提高网络的性能。对于监督学习任务深度学习方法消除了特征工程,将数据转换为类似于主成分的紧凑中间表示,并推导出去除表示冗余的分层结构。深度学习算法可以应用于无监督学习任务。这是一个重要的好处,因为未标记的数据比标记的数据更丰富。以无监督方式训练的深层结构的例子是深度信念网络。

3.2 卷积神经网络

卷积神经网络是一种深度学习模型,主要应用于图像识别、目标检测等视觉任务。卷积神经网络通过多个卷积层和池化层对输入的二维结构进行特征提取和降维,然后将特征输入全连接层进行分类或回归。卷积层是卷积神经网络中重要的组成部分之一,它通过一系列的卷积操作对输入数据进行特征提取。卷积操作可以理解为将一个滤波器(也称为卷积核)应用于输入数据的过程,滤波器通过滑动窗口的方式对输入数据进行卷积操作,从而得到特征图(feature map)。每个卷积层包含多个滤波器,每个滤波器可以提取出不同的特征,这些特征在不同位置可能存在,通过卷积操作可以提取出来。池化层是卷积神经网络中用于降低特征图尺寸的操作,它通过对特征图进行降采样,可以减少计算量和参数数量,并增强特征的鲁棒性。全连接层是卷积神经网络中的一种传统神经网络层,用于将特征图转换为分类或回归结果。在卷积层中,每个滤波器都是一组权重,通过卷积操作应用于整个输入数据的不同位置,因此滤波器中的权重是共享的。这种共享权重的方式可以大大减少参

① Shigeki, Sugiyama (12 April 2019). Human Behavior and Another Kind in Consciousness: Emerging Research and Opportunities: Emerging Research and Opportunities. IGI Global. ISBN 978-1-5225-8218-2.

数数量，并且可以保证卷积层具有平移不变性，即输入数据在平移时，卷积层的输出不变。

3.2.1　什么是卷积计算

在卷积神经网络中，卷积核是一个矩阵用来执行卷积运算。卷积运算是卷积神经网络中的基本运算，主要用于提取图像中的特征。卷积运算的基本流程是将卷积核与输入图像进行卷积，即将卷积核在输入图像上滑动，并在每一个位置上进行卷积运算。卷积运算的具体过程是将卷积核与图像的对应部分进行乘法运算，再将所有乘积求和得到一个新的值。卷积计算是一种数学运算，它可以在图像处理、信号处理、自然语言处理等领域中广泛应用。卷积计算通过将一个小矩阵（称为卷积核或滤波器）在输入数据（如图像）上滑动，并与输入数据在每个位置上的值进行乘法和加和运算，来得到输出数据。这种运算能够提取出图像中的特征，并且在深度学习中广泛使用。例如，给定一个 5×5 的图像和一个 3×3 的卷积核，如果要在图像上计算卷积，那么可以这样做。

（1）将卷积核放在图像的左上角，并将核中的每个元素与图像中对应位置的元素相乘。

（2）将这些乘积求和得到一个新的像素值，记为卷积结果。

（3）将卷积核右移一个像素，重复上述过程，直到卷积核移动到图像的右下角。

（4）每次移动卷积核都会得到一个新的卷积结果，这些结果组成了一个新的图像。

可以用粗体显示输入图像的哪些像素用于卷积的每个步骤，在每个步骤中将所选像素的值（粗体）与卷积核的对应值相乘，并将结果求和为单个输出，如图 3-2 所示。在下面的示例中，计算中心像素的卷积，卷积核移动到第 5 步，如图 3-3 所示。首先计算卷积核的每个像素与图像相应像素的乘积，然后求和得

$$2\times1+4\times3+6\times2+24\times2+5\times9+13\times3+1\times7+6\times1+8\times6=219$$

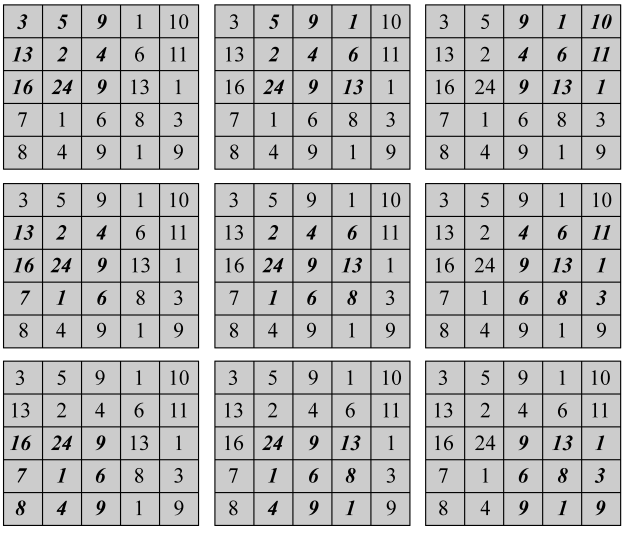

图 3-2　与卷积核的对应值相乘

因此，输出激活图的中心像素等于 219。将卷积核移动到图像的其他位置并进行同样的计算，最终得到输出图像。这只是一个简单的例子。在实际应用中，卷积核的大小和权重值通常是可以调整的，并且会有多层卷积层叠加在一起来获取更复杂的特征。在卷积过程

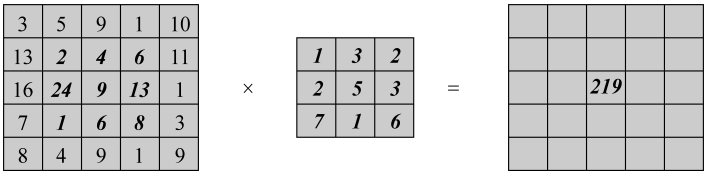

图 3-3　卷积核移动到第 5 步

中，卷积核从左到右、从上到下滑动，直到它穿过整个输入图像。步长(stride)是卷积操作中每次移动的距离，决定了输出特征图中相邻特征的距离，因此也影响了空间信息的丢失程度。例如，如果步长为 1，则相邻的特征将具有最大的重叠，从而最大程度地保留空间信息；如果步长为 2，则相邻特征的重叠将减少，并且输出特征图的大小将减半。将 S 定义为卷积核的步长，当 $S=2$ 时可以看到如图 3-4 所示。

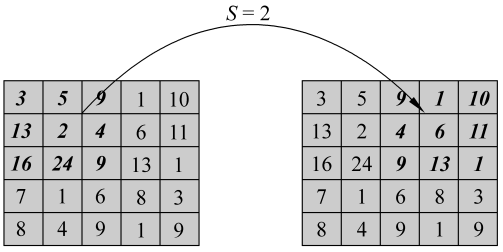

图 3-4　当 $S=2$ 时

在卷积层中，观察到位于角落和边缘的像素比中间的像素使用得少得多。可以看到像素(0,0)只使用了一次，而中心像素(3,3)使用了 9 次，通常位于中间的像素比位于边缘和角落的像素使用得更多。图像的边缘信息往往不能被充分利用，因为卷积核的尺寸比图像的尺寸小，导致图像边缘的信息在卷积过程中丢失。同样，因为图像的中间部分的信息会得到更多的关注，所以图像的中心信息也不能被充分利用。这个问题的一个简单而强大的解决方案是填充(padding)。填充是在进行卷积运算前在输入数据的边界处添加特定数量的补充数据的技术，可以使边缘和中心数据提取得更平滑，因为填充可以保留图像的边缘信息，并且避免了在卷积运算中破坏边缘数据的情况。填充还可以控制卷积运算的输出大小，因此可以在一定程度上控制特征提取的效果。一般来说，填充的数量为 0 或 1，但也可以选择更大的数量以调整卷积层的输出。如果在大小为 $W\times H$ 的输入图像中应用填充 P，则输出图像的尺寸为 $(W+2P)\times(H+2P)$。下面可以看到使用填充 $P=2$ 前后的示例图像，维度从 5×5 增加到 9×9，如图 3-5 所示。

通过在卷积层中使用填充，增加了角落和边缘像素对学习过程的贡献。现在继续本教程的主要目标，即介绍计算卷积层输出大小的公式。有以下输入。

✧ 图像尺寸 $W_{\text{in}}\times H_{\text{in}}$。

✧ 卷积核尺寸 $K\times K$。

✧ 步长 S 和填充 P。

输出激活图将具有以下维度。

$$W_{\text{out}}=\frac{W_{\text{in}}-K+2P}{S}+1,\quad H_{\text{out}}=\frac{H_{\text{in}}-K+2P}{S}+1$$

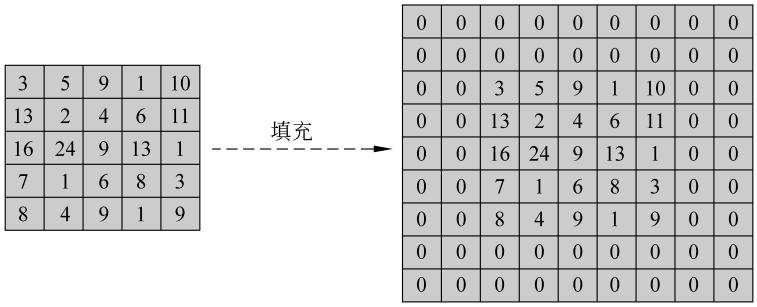

图 3-5　填充 $P=2$

如果输出维度不是整数，则意味着没有正确设置步长，有两个特殊情况。

✧ 当没有填充时，输出尺寸为$\left(\frac{W_{\text{in}}-K}{S}+1,\frac{H_{\text{in}}-K}{S}+1\right)$。

✧ 如果想在卷积层之后保持输入的大小不变，在 $W_{\text{out}}=W_{\text{in}}$ 和 $H_{\text{out}}=H_{\text{in}}$ 处应用相同的填充。如果 $S=1$，设置 $P=\frac{K-1}{2}$。

最后，将展示一个计算卷积层输出大小的示例。假设有一个 125×49 的输入图像和一个 5×5 的卷积核，$P=2$ 和 $S=2$。那么输出维度如下：

$$W_{\text{out}}=\frac{125-5+2\times2}{2}=\frac{124}{2}=62,\quad H_{\text{out}}=\frac{49-5+2\times2}{2}=\frac{48}{2}=24$$

因此，输出激活图的维度为(62,24)。

卷积计算和逻辑回归是不同的概念，但它们可以结合在一起使用。实际上，卷积计算和线性方程有着密切的联系，卷积核对应线性方程中的系数矩阵，卷积核与图像的卷积操作等价于矩阵乘法。卷积计算的每个步骤都可以看作线性方程的矩阵乘法，因此卷积计算可以看作一种在图像处理中应用线性代数的方法。逻辑回归是一种分类算法，它将输入特征映射到类别标签。在卷积神经网络中，逻辑回归常常用作最终的分类层，通过处理从卷积层提取的特征来预测图像的类别，因此卷积计算和逻辑回归可以协同工作，将从图像中提取的特征与分类任务相结合，以实现高效的图像分类。

3.2.2　感受野与卷积层

在本章节中，将探索的卷积神经网络是由 Yann LeCun 等于 1998 年首次发明的。LeCun 和他的团队实施的想法较早，并且建立在 David H.Hubel 和 Torsten Weisel 开创性论文中提出的想法之上，该论文为他们赢得了 1981 年的诺贝尔生理学和医学奖。他们探索了动物的视觉皮层，发现了大脑一小块但定义明确的区域活动与视野小区域活动之间的联系。在某些情况下，甚至可以精确定位负责部分视野的确切神经元。这导致他们发现了**感受野**(receptive field)，它表示了神经网络中一个神经元或特征图上的输出受到输入数据中多大区域的影响。在卷积神经网络中，每个卷积层的输出都由一组卷积核和输入数据中特定区域的像素共同决定。这些卷积核在输入数据上滑动并执行卷积操作，以生成输出特征图。在这个过程中，每个神经元或特征图上的像素都具有一个感受野，它指的是输入数据中对应的区域，能够影响该像素的输出值。在卷积层中，神经元的感受野大小由卷积核的大小

和卷积层的深度共同决定，如图 3-6 所示。卷积层的深度增加会导致更大的感受野，因为每个神经元接受输入的区域将包括前一层中更多的神经元。这种增加的感受野大小有助于网络更好地理解输入数据的全局特征。在图像处理中，增加卷积核的大小和层数可以扩大神经元的感受野，使其能够接收更广阔的视觉信息，并提高网络的感知能力。特别是在处理高分辨率图像时，较大的卷积核和更深的网络可以帮助捕捉更丰富和复杂的特征。但需要注意的是，增加卷积核的大小和层数可能会增加计算负担和内存需求，因此需要在设计网络时进行平衡。此外，对于某些任务和数据集，较小的感受野可能更为适合，因此需要根据具体情况进行选择。

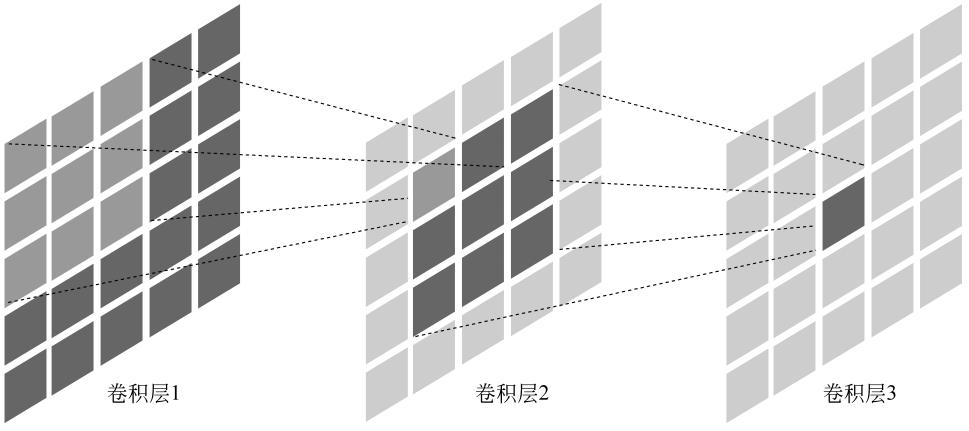

图 3-6　卷积神经网络中感受野的示意图

在卷积神经网络中，每个卷积核（或称卷积滤波器）都有自己的感受野，这些感受野可以用来捕捉图像中不同尺度的特征。在多层卷积网络中，每一层的感受野都不同，前面层的感受野较小，后面层的感受野较大。这样可以捕捉图像中不同级别的特征。卷积核和感受野之间存在一个直接关系，卷积核的大小就是感受野的大小，当卷积核的大小变大时感受野也会变大；反之，当卷积核的大小变小时感受野也会变小。感受野决定了卷积核能够感知到的图像区域的大小，因此影响了卷积核能够提取的特征的范围。例如，如果卷积核的感受野很小，它只能捕捉到图像的局部特征，如边缘和纹理；而如果卷积核的感受野很大，它能够捕捉到图像的全局特征，如整体形状和颜色。卷积核的感受野大小可以通过调整卷积核的大小来调节，通过使用不同感受野大小的卷积核可以捕捉不同级别的特征，进而提升网络的性能。另外，每个卷积核的感受野大小由卷积核的大小和卷积核在输入上移动的范围共同决定。卷积核的大小决定了卷积核能够感知到的图像区域的大小，较大的卷积核能够感知到更大的图像区域，因此感受野也会变大。卷积核在输入上移动的范围也是影响感受野大小的重要因素。如果卷积核在输入上移动的范围较大，那么卷积核能够滑动到图像的更多区域，感受野也会变大；而如果卷积核在输入上移动的范围较小，那么卷积核能够滑动到图像的区域就会变小，感受野也会变小。通过调整卷积核的大小和在输入上移动的范围，可以调整每个卷积核的感受野大小，使得网络能够捕捉到不同级别的特征。在图像处理中，感受野越大，表示该单元或层能够“看到”的输入信息就越多，因此感受野的大小对于网络的性能有很大影响。感受野大小是卷积神经网络中非常重要的概念。以下是影响感受野大小的因素。

✧ 卷积核大小：每个卷积核的大小决定了它可以感受到输入张量的区域大小。较大的

卷积核可以捕捉更大范围的特征，因此会导致更大的感受野大小。

- ✧ 卷积层数：卷积神经网络中的每一层都会扩大上一层的感受野。因此，较深的网络结构可以捕捉到更全局的信息，因此会导致更大的感受野大小。
- ✧ 步幅和填充：卷积神经网络中的步幅和填充参数会影响每一层的输出张量大小，进而影响下一层的感受野。较大的步幅和填充会缩小输出张量的大小，因此会导致较小的感受野大小。

总之，感受野大小是由多种因素共同决定的，这些因素包括卷积核大小、卷积层数、步幅和填充等。理解这些因素对于设计和优化卷积神经网络结构非常重要。

卷积运算的结果是一个新的图像，这个图像称为卷积后的图像（或称特征图）。卷积后的图像中的每一个像素点都是卷积核与输入图像对应位置上的值的结果。卷积后的图像能够捕捉到输入图像中的特征，这些特征可以用来识别图像中的对象。在卷积神经网络中，卷积运算可以用来提取图像中的特征，这些特征可以用来识别图像中的对象。通过使用不同的卷积核和不同的卷积运算，可以捕捉到不同级别的特征，进而提升网络的性能。卷积核在图像上滑动，与图像的每一部分相乘并相加得到新的图像。卷积核的大小一般是奇数，如 3×3、5×5、7×7 等。卷积核的权重和偏置决定了它对图像的提取特征的能力。在卷积神经网络中，一般会有多个卷积核，每个卷积核会提取不同的特征。

可以将卷积神经网络定义为具有一个或多个卷积层的神经网络，这是一个快速而简单的定义。有些架构虽然使用卷积层但不会被称为卷积神经网络，因为卷积神经网络可能会包含其他类型的层，如池化层、全连接层等。但是，卷积层是卷积神经网络的核心部分，使用卷积核来提取图像中的特征，所以现在必须具体描述什么是卷积层。卷积层接收图像和小型逻辑回归。例如，输入大小为 4（通常输入大小为 4 或 9 是常见的大小，有时也可能是 16）并将逻辑回归应用于整个图像。在这种情况下，卷积核会在图像上滑动，并对图像的每个区域应用逻辑回归。这种方法可以提取图像中的特征，并将其用于分类或其他目的。这些大小决定了卷积核能够滑动到图像上的区域大小，较小的输入可以提取更细粒度的特征，而较大的输入可以提取更粗粒度的特征。这意味着第一个输入由扁平向量的分量 1～4 组成，第二个输入是分量 2～5，第三个是分量 3～6，以此类推，可以在如图 3-7 所示的底部看到该过程的概述。这个过程创建了一个小于整体输入向量的输出向量，因为从分量 1 开始并取 4 个分量并产生一个输出，最终结果是：如果使用逻辑回归沿着展平的输入向量移动，当移动到第 10 维输入变量时将产生一个 7 维输出向量。在卷积神经网络中，如果卷积层应用于一维数据，如时间序列或声音信号，则称为一维卷积层或时间卷积层。这种类型的卷积层主要用于语音识别、音频处理和时间序列预测等任务中。一维卷积层的工作方式与二维卷积层相似，但在这种情况下卷积核将沿着一维数据滑动并与其进行卷积运算。实际上，一维卷积层不一定只用于时间序列数据，可用于任何一维数据，如文本序列或数字序列，这是因为：对于任何二维或多维数据都可以将其展平为一维数据，并使用一维卷积层处理。在这种情况下，卷积核会沿着展平后的一维数据滑动，并与其进行卷积运算。术语“时间”强调这种类型的卷积层是专门设计用于处理时间相关的数据。这与常用于图像处理应用的经典二维卷积层形成对比。

在卷积神经网络的概念中，感受野是一个重要的方面，但并不是构建卷积神经网络所需的组成部分。根据上面的例子，构建功能性卷积神经网络需要三个关键组件。第一个就是

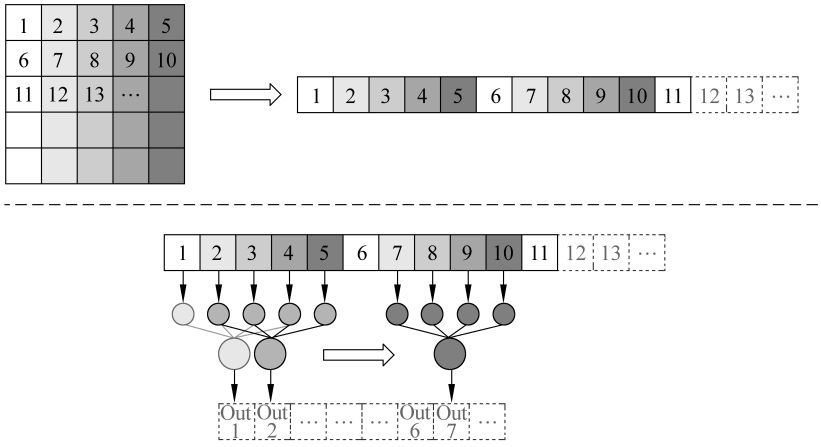

图 3-7　使用逻辑回归构建一维卷积层

卷积层，即通过使用卷积核和感受野的概念来提取图像的特征，这些特征可以用来识别图像中的对象。已经了解了卷积层，那么另外两个层次是什么呢？其中之一是展平层，将图像（二维数组）展平为向量。在传统的神经网络中，输入数据通常是一个向量，但是在卷积神经网络中输入数据是一个图像，也是一个二维数组，因此需要将图像展平为向量才能将它作为输入数据输入到网络中。尽管现代的计算机能够很容易地处理多维数组，但在底层它们通常会被扁平化为一个一维向量。这样做的原因是：计算机中的内存和处理器都是一维的，所以将多维数组扁平化为一维向量后，能够更容易地访问和处理数据；另外，扁平化数组还可以使得计算机更容易地利用现有的硬件资源，提高计算效率。在处理多维数组时，扁平化操作可以自动完成，避免了人工编写复杂的索引计算代码，同时读者能够更好地理解某些技术细节。可以在图 3-7 的顶部看到使用 3×3 的卷积核扁平化图像。我们已经介绍了卷积层和展平层，卷积神经网络的第三个主要层次称为主力层，是获取图像矢量并将其提供给一个负责处理的主力神经元。主力层在卷积神经网络中通常被称为全连接层或密集层，其作用是将卷积层和展平层的输出向量连接在一起，并将其输入到一个多层感知机中进行分类或回归任务。主力层中的每个神经元都与前一层中的每个神经元相连接，从而将所有的输入特征结合起来，并计算输出的预测结果。在主力层中，可以使用各种激活函数，如 Sigmoid、ReLU 和 Tanh 等，来增加网络的非线性能力，并更好地捕捉输入数据中的非线性特征。使用逻辑回归作为主力层是一种常见的方法，其中 Sigmoid 函数被用作激活函数。但是，卷积神经网络中的主力层通常包含多个全连接层，并使用更复杂的激活函数和正则化技术，如 Dropout 和批量归一化等。总体来说，卷积神经网络中的主力层是一个关键组件，它可以提取输入图像中的高级特征，并将其转换为可用于分类或回归的输出结果。总之，在卷积神经网络中，这三个主要层次分别负责特征提取、特征转换和分类等任务。具体来说，卷积层是负责特征提取的层，其输入是图像或者其他类型的数据，输出是图像中不同位置、不同尺度、不同方向的特征图。卷积层通过卷积操作将输入数据与一组卷积核进行卷积计算，提取出图像中的局部特征。展平层是将卷积层输出的特征图转换为一维向量的层，其目的是将特征图中的二维空间信息转换为一维向量，以便输入到全连接层中进行分类或回归等任务。全连接层是负责分类、回归等任务的层，其输入是展平层输出的一维向量，输出是

具体的分类或回归结果。全连接层通过一组权重矩阵和偏置向量对展平层的输出进行线性组合，并通过激活函数进行非线性变换，最终输出分类或回归结果。

在卷积神经网络中，每个卷积层的每个神经元都对应于输入图像的一个局部感受野，它能够看到输入图像中的一部分区域，并从中提取特征。局部感受野的大小通常由卷积核的大小和步长决定。卷积核的大小决定了每个神经元能够看到的局部区域的大小，而步长决定了卷积核在输入图像上滑动的步幅。通过调整卷积核的大小和步长，可以控制每个神经元的感受野大小，从而提取不同尺度和粒度的特征。局部感受野可以捕捉图像的局部特征，如边缘、纹理、角点等。通过多个卷积层的堆叠，可以逐渐扩大神经元的感受野，提取更高层次、更抽象的特征，如物体的形状、轮廓、纹理、颜色等。这些特征可以用于识别图像中的对象、区分不同类别的图像等任务。因此，在卷积神经网络中，局部感受野是通过卷积操作在图像上使用小型卷积核的过程来实现的。这些卷积核对输入图像的每个小块进行卷积计算，从而捕捉局部特征，逐渐提取出更高层次、更抽象的特征，实现对图像的有效分析和处理。局部感受野定义了卷积神经网络中每个神经元所连接的输入图像中的区域。在卷积神经网络中，卷积层中的每个神经元都使用局部感受野与输入图像中的一部分进行卷积操作，以捕捉图像中的空间特征。而逻辑回归是一种线性分类器，它不会直接应用于局部感受野的输出。在卷积神经网络中，卷积层通常会紧接着池化层，然后才是全连接层和最终的逻辑回归分类器。局部感受野和逻辑回归在卷积神经网络中都是用于实现图像分类和对象识别的重要组件。

如果步长为1，卷积核从左边开始向右走，完成后向下一排，可以在图3-8的顶部看到此过程的步骤。如果使用3×3局部感受野扫描17×8的图像，作为本地感受野的输出将得到一个15×6的数组（如图3-8所示的底部），这样就完成了一个二维卷积层来说，很明显将获得更少的输出。在卷积神经网络中，二维卷积层通常用于处理二维图像数据，如灰度图像或彩色图像。每个像素都可以被视为二维坐标系中的一个点，其值表示该点的亮度或颜色信息。这些图像可以被视为二维矩阵，其中每个元素表示一个像素的值，而每个通道则对应于一个颜色通道或特定类型的信息通道。对于二维卷积层来说，卷积核的形状也是二维的，它在图像上滑动以提取特征。与一维卷积层不同，二维卷积层需要考虑图像数据的空间结构，因此需要卷积核的二维形状和滑动窗口的二维移动。这些层通常称为二维卷积层或平面卷积层。如果需要处理三维数据，如视频数据，那么卷积核的形状将变为三维。类似地，如果要处理四维或更高维度的数据，则需要使用超空间卷积层。最后，逻辑回归模型的输入可以是任何维度的数据，而选择使用4、9或16这些数字的滤波器大小只是因为它们方便构建一个平方形的滤波器，并且可以覆盖多个像素。卷积神经网络具有多个层。想象一个由三个卷积层和一个全连接层组成的卷积神经网络，假设它将处理大小为10×10的图像，并且所有三层都有一个3×3的局部感受野，它的任务是确定一张图片中是否有汽车。看看网络是如何工作的：第一层采用10×10的图像，产生大小为8×8的输出（它具有随机初始化的权重和偏差）；然后将其提供给第二个卷积层，有自己的局部感受野和随机初始化的权重和偏差，决定将它也设为3×3，这会产生大小为6×6的输出，并将其提供给第三层；第三个卷积层产生一个4×4的图像；然后将其展平为16维向量，并将其发送到一个全连接层，该层具有一个输出神经元并使用逻辑函数作为其非线性。

训练卷积层涉及训练局部感受野，也就是全连接层中的权重和偏置，其具有单一偏置和

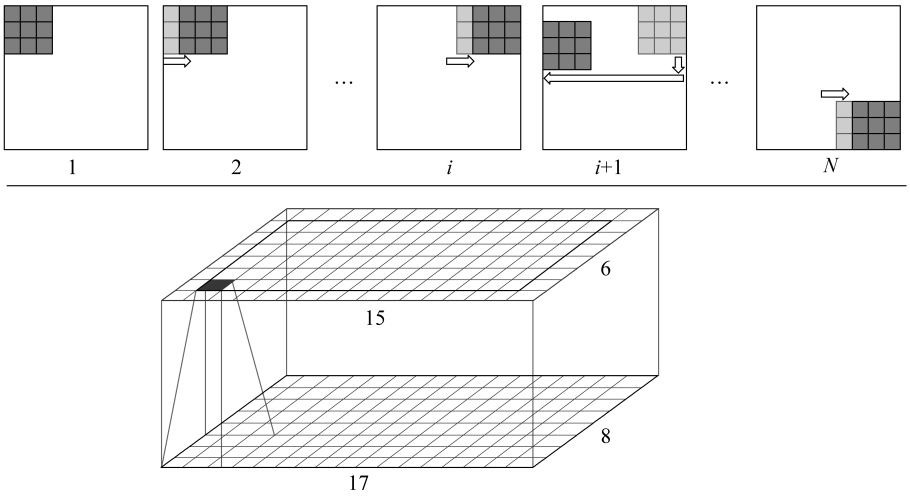

图 3-8　二维卷积层

少量权重(等于局部感受野中的单元数)。这使其类似于一个小型的逻辑回归模型,这也是卷积神经网络可以快速训练的原因之一,因为它们只需要学习少量的参数。主要区别在于,局部感受野中可以使用任何激活函数,而逻辑回归中应该使用逻辑函数。最常用的激活函数是整流线性单元或 ReLU,它的作用是将负输入设置为 0,而将正输入保持不变。

3.2.3　特征图和池化层

在处理具有多个通道的图像时,可以在每个通道上使用相同大小但不同权重和偏差初始化的多个局部感受野。这种方法称为多通道卷积或深度卷积。例如,如果想要在每个通道上使用 5 个局部感受野,可以创建 5 个大小相同但不同权重和偏差初始化的卷积核,每个卷积核在每个通道上都有一个副本。对于具有 3 个颜色通道的图像,一种常见的处理方法是将每个通道的图像分开处理,然后将结果合并。当输入图像通过这些卷积核时,每个通道会产生 5 个特征图。回想一下,卷积层使用如 3×3 的局部感受野(9 个权重,1 个偏差)扫描 10×10 的图像,并构建一个新的 8×8 图像作为输出。这意味着当卷积核扫描 10×10 三通道图像时,它们将构建 15 个 8×8 输出图像。这些图像称为特征图。然后,可以将这些特征图合并为一个输出张量。多通道卷积可以增加模型的表示能力,使其能够更好地捕捉图像中的结构和模式。然而,使用更多的卷积核会增加模型的参数数量和计算成本,因此需要在模型的性能和计算效率之间进行权衡。需要注意的是,在使用多通道卷积时,每个通道上的权重和偏差初始化应该是不同的,以允许模型学习不同通道之间的不同特征。

特征图是卷积神经网络中非常重要的概念,是卷积层的输出结果。在每个卷积层中,卷积核在输入图像上滑动并计算每个位置上的卷积结果,这些结果被组织成一个特征图。每个特征图都可以看作一个高维数组,其中每个元素代表了卷积核在输入图像的一个位置上计算得到的结果。通过在特征图上执行池化操作,可以减少特征图的维度,有助于简化问题和减少计算负担。在深度卷积神经网络中,特征图是学习到的关键特征之一,这些特征对于分类和识别任务非常重要。在训练过程中,卷积层的权重和偏差会随着数据的输入而不断地更新,以优化模型的准确性和泛化性能。通过多个卷积层的叠加,网络可以学习到越来越

抽象和高级的特征，从而实现更加准确的图像分类和识别。对于狗的图像分类任务，可以通过将网络的最后一个或多个卷积层的特征图输入到全连接层，以生成最终的分类结果。在这个过程中，可以选择仅使用具有最高准确性的特征图。例如那些能够识别狗图片上的眼睛和鼻子的特征图。这样可以提高网络的准确性并加速训练过程。

这里的主要思想之一是将 10×10 的 3 通道图像变成 8×8 的 15 通道图像，输入图像被转换为更小但更深的对象，使图像更小意味着将信息包装在更紧凑(但更深)的表示中，这将发生在每个卷积层中。在追求紧凑性的过程中，可能会在卷积层之后或之前添加一个新层，这个新层称为最大池化层。最大池化层将池大小作为超参数，通常为 2×2，然后按以下方式处理其输入图像：将图像划分为 2×2 区域(如网格)，并从每 4 个像素中取池化具有最大值的像素。因此，输出图像的大小为输入图像的宽度和高度都除以 2。将这些像素组合成一个新图像，其顺序与原始图像相同。例如，如果输入图像的大小为 10×10，则经过 2×2 的最大池化层处理后，输出图像的大小将变为 5×5。原始图像中的每个 2×2 区域被映射到输出图像中的一个像素。此外，最大池化层不会增加通道数，因此可以在不增加计算量的情况下缩小图像。

最大池化背后的想法是：图片中的重要信息很少包含在相邻像素中，所以这里采用了“四选一”将 4 个像素缩减为一个；另外，重要信息通常包含在较暗的像素中，所以使用最大值选择最暗的一个像素代表整个区域。在卷积神经网络中，最大池化层通常应用于卷积层的输出特征图上，而不是原始输入图像。因为在卷积层中，特征图已经被卷积核处理过，从而提取了更抽象的特征，而最大池化层可以帮助减少特征图的大小和参数数量，提高计算效率，并且有时还能够提高模型的泛化能力。可以尝试修改示例中的代码(下一节)，然后打印出来自卷积层的特征图，可以将最大池化视为降低屏幕分辨率的过程。一般来说，如果在 1200×1600 的图像上认出一只狗，可能也会在 600×800 的颗粒感图像上认出它。可能会注意到，假设图像中重要信息通常包含在较暗的像素中并不总是适用于所有图像。这是一个非常强的预设条件假设，仅在某些特定情况下有效，因此在使用该算法时应根据实际情况进行评估和调整。除了最大池化，平均池化和 L^2 范数池化等不同的池化方法也可以应用于卷积神经网络。平均池化采用每个 2×2 图像块的平均值，而 L^2 范数池化则采用每个 2×2 图像块像素的 L^2 范数的平均值。这些池化方法也可以实现类似于最大池化的降维效果，但可能对图像的细节信息保留更多或更少，具体取决于不同的应用场景和任务。

当图像通过网络时，经过若干层后得到带有很多通道的小图像，然后可以将其展平为一个向量，并在最后使用一个简单的逻辑回归来提取与分类问题相关的部分。逻辑回归(这次使用逻辑函数)将挑选出用于分类的部分并预测一个结果，该结果将与目标进行比较，然后误差将被反向传播，形成了一个完整的卷积神经网络。一个简单但功能齐全的四层卷积神经网络如图 3-9 所示。

在卷积神经网络中，下采样是一种常用的技术，它可以通过减少特征图的空间分辨率来降低特征图的维度，同时保持关键信息的相对位置和关系不变。下采样通常使用池化层或跨步卷积来实现。在池化层中，最常用的下采样技术是最大池化和平均池化。这些操作将输入特征图划分为不重叠的区域，然后在每个区域上应用池化操作。例如，在最大池化中，取每个区域内的最大值作为输出，而在平均池化中，取每个区域内的平均值作为输出。这样，输出特征图的空间分辨率会缩小，但特征图的通道数不变。另一种常用的下采样技术是

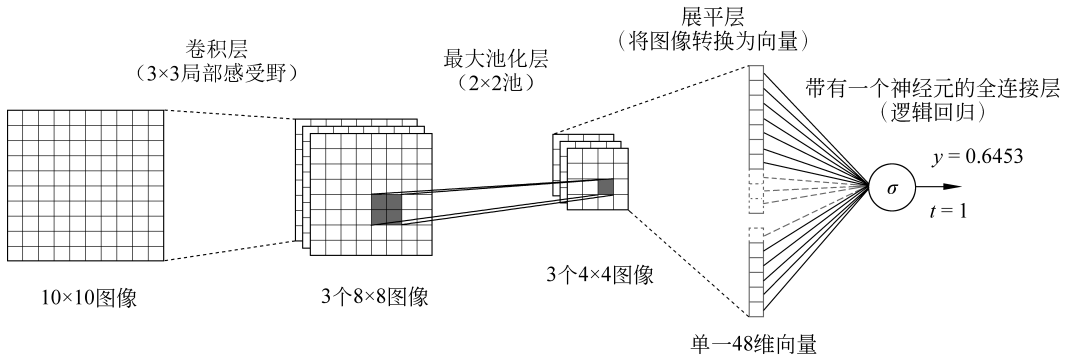

图 3-9　具有一个神经元全连接层的四层卷积神经网络

跨步卷积，它通过使用大于 1 的步幅来移动卷积核，从而减小输出特征图的空间分辨率。跨步卷积与池化的不同之处在于，它可以通过训练来学习卷积核中的参数，以从输入特征图中提取有用的特征，并且可以在网络中的任何地方使用。总之，下采样是一种常用的技术，用于减小特征图的维度和大小，同时保持关键信息的相对位置和关系不变。在卷积神经网络中，下采样通常使用池化层或跨步卷积来实现。

卷积神经网络相对于全连接神经网络在训练过程中更容易的原因主要有以下几点。

- ✧ 参数共享：在卷积层中，通过使用卷积核对输入进行卷积操作，参数权重被共享，即同一个卷积核在图像的不同位置使用相同的权重。这导致了卷积神经网络中的参数数量较少，从而减少了训练过程中需要优化的参数数量，降低了训练复杂度。
- ✧ 局部感知机制：卷积层通过局部感知机制，只关注输入图像的局部区域，而不是整个图像。这使得 CNN 能够更好地捕捉图像中的局部特征，如边缘、纹理等，从而提高了模型对图像特征的表示能力。
- ✧ 并行计算：在卷积神经网络中，每个卷积核都可以独立地进行卷积操作，因此特征图之间的计算是并行的。这可以有效地利用并行计算的优势，加速了网络的训练过程，尤其在处理大规模图像数据时更加显著。
- ✧ 减少过拟合：由于卷积层和池化层的存在，卷积神经网络可以减少模型的参数量，从而减少了过拟合的风险。池化层可以降低特征图的尺寸，减少模型对输入数据的过度拟合。此外，通过使用 Dropout 等正则化技术，可以进一步减少过拟合的风险。
- ✧ 梯度传播效果更好：在全连接神经网络中，误差的反向传播是顺序的，每层的误差都依赖于前一层的误差，导致梯度可能在传播过程中逐渐减小或消失，从而影响训练效果。而在卷积神经网络中，卷积层和池化层的存在可以减小特征图的尺寸，降低了梯度的传播路径，使得梯度能够更好地传播，从而加速了训练过程。

综上所述，卷积神经网络由于其参数共享、局部感知机制、并行计算、减少过拟合，以及梯度传播效果更好等特点，使得其在图像处理任务中相对容易训练，并且在许多计算机视觉领域中取得了显著的成功。

3.2.4　一个卷积网络

现在用 Python 展示一个完整的卷积神经网络。使用 Keras 库能够从组件构建神经网

络，而不必过多担心数据的复杂度。这里的所有代码都应该放在一个 Python 文件中，然后在终端或命令提示符下执行。文件中代码的第一部分实现 Keras 和 Numpy 库的导入。

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D, MaxPooling2D
from keras.utils import np_utils
from keras.datasets import mnist
(train_samples, train_labels), (test_samples, test_labels)=mnist.load_data()
```

可能会注意到正在从 Keras 存储库中导入 MNIST。这段代码的最后一行在 4 个不同的变量中加载了训练样本(train_samples)、训练标签(train_labels)、测试样本(test_samples)和测试标签(test_labels)。这个 Python 文件中的大部分代码实际上将用于格式化(或预处理)MNIST 数据，以满足将其输入卷积神经网络所必须满足的要求，代码的下一部分处理 MNIST 图像。

```
train_samples=train_samples.reshape(train_samples.shape[0], 28, 28, 1)
test_samples=test_samples.reshape(test_samples.shape[0], 28, 28, 1)
train_samples=train_samples.astype('float32')
test_samples=test_samples.astype('float32')
train_samples=train_samples/255
test_samples=test_samples/255
```

首先注意到代码实际上是重复的，所有操作都是在训练集和测试集上执行的，将只是讨论训练集，测试集以相同的方式运行。此代码块的第一行重塑了包含 MNIST 的数组，这个重塑的结果是一个(60000,28,28,1)维数组。第一个维度简单来说就是样本的个数；第二个和第三个代表[28×28]维的图像；最后一个维度是通道，图像可能是 RGB，但 MNIST 是灰度的，所以这似乎是多余的，但这包括了重塑数组的全部意义。这背后的原因是：随着通过卷积层进行处理，在这个处理方向特征图将被提取，因此需要准备张量以能够接收它。第三行将数组中的条目声明为 float32 类型，这仅仅意味着它们将被视为十进制数。Python 会自动执行，但是 Numpy 需要类型声明会大大加快计算速度，所以必须放入这一行。第五行将数组条目从 0～255 的范围标准化为 0～1 的范围(被解释为像素中灰度的百分比)。我们现在处理样本，必须使用 one-hot 预处理标签(0～9 的数字)，使用以下代码执行此操作。

```
c_train_labels=np_utils.to_categorical(train_labels, 10)
c_test_labels=np_utils.to_categorical(test_labels, 10)
```

这样就完成了数据的预处理，可以继续构建实际的卷积神经网络，以下代码指定层。

```
convnet =Sequential()
convnet.add(Convolution2D(32, 4, 4, activation='relu', input_shape=(28,28,1)))
convnet.add(MaxPooling2D(pool_size=(2,2)))
convnet.add(Convolution2D(32, 3, 3, activation='relu'))
convnet.add(MaxPooling2D(pool_size=(2,2)))
convnet.add(Dropout(0.3))
convnet.add(Flatten())
convnet.add(Dense(10, activation='Softmax'))
```

该代码块的第一行创建了一个新的空白模型，此处其余行按照卷积神经网络的规范填充了。第二行添加了第一层，在这种情况下它是一个卷积层，产生 32 个特征图，以 ReLU 作为激活函数，并具有[4×4]的局部感受野。input_shape 是 Keras 中模型层中常用的参数，表示模型期望接收的输入数据的形状。在使用 Keras 构建神经网络时，每一层都需要指定输入数据的形状，以便 Keras 能够自动计算权重矩阵的形状，对于第一层，input_shape 参数必须指定，因为这是模型的输入层；对于其他层，input_shape 参数通常是隐含的，因为 Keras 可以从前一层自动推导出它的输入形状。“input_shape”参数的形式是一个元组，其中包含输入数据的样本数(通常是 None)和每个样本的形状。例如，在图像分类任务中，一个常见的 input_shape 参数可以是(None,28,28,1)，表示输入图像是[28×28]像素的灰度图像。如果传递给 Keras 的张量具有不同的形状，可能会导致代码崩溃，因此在指定 input_shape=(28,28,1)之后，不必担心输入的张量为(65600,28,28,1)而不是(60000,28,28,1)，但是如果给它一个(60000,29,29,1)甚至(60000,28,28)的数据集，代码就会崩溃。

第三行定义了一个最大池化层，池的大小为[2×2]，定义了第二层；下一行定义了第三层，这是一个局部感受野为[3×3]的卷积层，这里不必指定输入尺寸，Keras 会从上一层隐含推出。之后，定义另一个最大池化层，池大小也是[2×2]，定义了第四层；在此之后有一个退学层，这不是一个真正的层，而只是对前一层和下一层之间全连接数目的修改，0.3 的退学率表示只包括所有连接的 3%；下一行将张量展平，这里描述的是将固定大小的矩阵转换为向量的过程。张量可以通过将它们重塑为一维数组来展平，此过程涉及通过沿单个维度连接张量的所有元素，将多维张量转换为一维向量，展平张量的过程通常在最后的池化层之后执行，然后再将数据传递到全连接层。

然后，将扁平向量输入到最后一层(此块中的最后一行代码)，这是一个标准的全连接前馈层(在 Keras 中称为 Dense)，接收与扁平向量中的分量一样多的输入，并输出 10 个值(10 个输出神经元)，每个神经元代表一个数字，并输出相应的概率，具体表示哪个数字实际上仅由对标签进行 one-hot 编码时的顺序定义。最后一层使用 Softmax 激活函数用于多于两个以上分类的情况。Softmax 激活函数和逻辑激活函数(Sigmoid)都属于概率分布的函数，但它们有很大的不同。逻辑函数只能用于二分类问题，它的输出范围在(0,1)，表示每一类的概率；而 Softmax 函数则可以用于多分类问题，它的输出是一个概率向量，每个元素的总和为 1。后面的章节中将继续描述它，现在只须将其视为用于多分类的逻辑函数，每个分类对应每个 0～9 的标签。

到现在为止，已经指定了一个模型，必须编译它。编译模型意味着 Keras 现在可以推断和填充未指定的所有必要细节，例如第二个卷积层的输入大小或扁平化向量的维度。下一行代码的作用是编译模型。

```
convnet.compile(loss='mean_squared_error', optimizer='sgd', metrics=
                ['accuracy'])
```

在这里可以看到已经将训练方法指定为 sgd，是指随机梯度下降，以均方误差 MSE 作为误差函数，还要求 Keras 在训练时计算准确度。下一行代码训练编译后的模型。

```
convnet.fit(train_samples, c_train_labels, batch_size=32, nb_epoch=20,
            verbose=1)
```

这行代码使用 train_samples 作为训练样本，使用 c_train_labels 作为训练标签来训练模型，还使用 32 的批量大小并训练 20 个 epoch。verbose 标志设置为 1，这意味着它将打印出训练的详细信息。现在继续代码的最后一部分，打印出训练模型的准确度，并根据学习到的模型对一组新数据进行预测。

```
metrics=convnet.evaluate(test_samples, c_test_labels, verbose=1)
print()
print("%s: %.2f%%" %(convnet.metrics_names[1], metrics[1] * 100))
predictions =convnet.predict(test_samples)
```

最后一行很重要，这里放了测试数据集 test_samples，要使用该模型进行预测，应使用与 test_samples 具有相同维度的新样本，但第一个维度除外（该维度包含单独的训练样本）。这是为了确保新样本具有与 test_samples 相同的结构，并且可以由模型处理。predictions 和 c_test_labels 数组的第一维应该相同，表示测试集中的样本数，它们的剩余维度应该匹配，代表每个样本的预测标签的形状。可以在末尾添加一行 print(predictions)以查看实际预测，或者 print(predictions.shape)查看存储在预测中的数组的维数。这 29 行代码形成了一个功能齐全的卷积神经网络。

3.2.5　用于文本分类

卷积神经网络是一种用于图像分类和识别的深度学习技术。它利用卷积层来提取图像的特征，再通过多层神经网络来分类。在自然语言处理中，卷积神经网络也可以用于文本分类和语义分析等任务。在自然语言处理中，卷积操作通常是在词向量空间上进行的，而不是在图像空间上。例如，一个卷积核可以用于检测一个特定的词汇或语法结构，通常使用词嵌入来将文本数据转换为数值矩阵，然后再使用卷积神经网络进行分类。这种方法能够有效地处理变长的文本序列，并且具有很好的分类性能。总之，卷积神经网络在自然语言处理中也是一种有效的技术，可以通过卷积操作来提取文本的特征，从而进行文本分类和语义分析等任务。下面介绍三个卷积神经网络用于文本处理的模型。

Kim 于 2014 年发布了一篇名为 *Convolutional Neural Networks for Sentence Classification* 的论文，该论文介绍了一种基于卷积神经网络的文本分类模型。这篇论文对自然语言处理领域的发展产生了重大影响，并启发了许多后续研究的方向。在这篇论文中，Kim 提出了一种基于卷积神经网络的模型，可以对给定的文本进行分类。例如，将文本分类为正面或负面情感。这种模型在处理自然语言处理任务时表现出色，并且与传统的基于词袋模型和 n-gram 特征的方法相比，具有更好的性能。Kim 使用了 Collobert 等在 2011 年发表的论文中提出的具有单个卷积层的简单卷积神经网络架构作为模型的基础架构，如图 3-10 所示。这个模型被称为 CNN-rand，它的输入是单词的随机初始化向量。在 CNN-rand 模型中，模型首先使用一个固定大小的卷积核来提取 n-gram 特征，其次对提取的特征进行池化操作，最后通过一个全连接层进行分类。具体而言，模型首先将输入的单词转换为随机初始化的向量表示，然后将这些向量堆叠成矩阵的形式作为卷积层的输入。卷积层使用一个固定大小的卷积核来提取特征，然后通过一个池化操作来减少特征的数量。最后，通过一个全连接层来将提取的特征映射到具体的分类结果。尽管 CNN-rand 模型比传统的基于 n-gram 特征的文本分类方法有一定的性能提升，但它的随机初始化向量表示并没有考虑单词的语义信

息，这在一定程度上限制了模型的性能。因此，后续的研究提出了许多使用预训练的词向量来替代随机初始化向量的方法，如使用 Word2Vec、GloVe 等预训练的词向量。

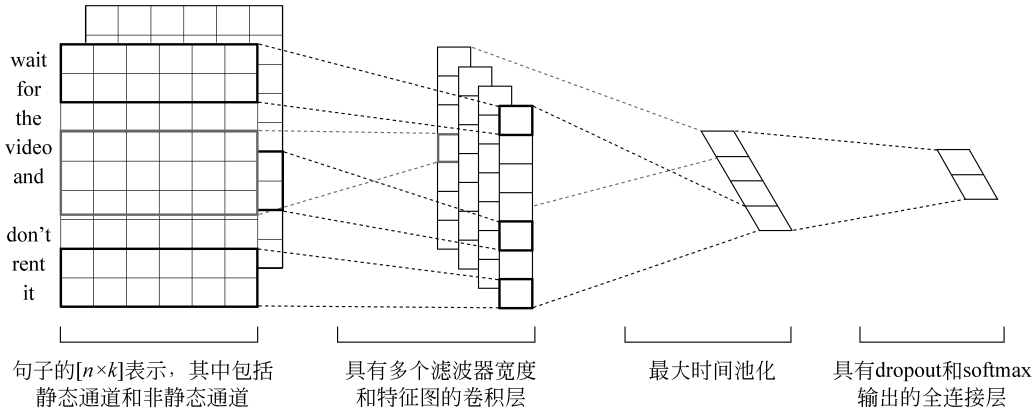

图 3-10　用于文本分类的卷积神经网络模型架构

第二个模型是由 Zhang、Zhao 和 LeCun 在 2015 年发表的、名为《字符级卷积神经网络用于文本分类》(*Character-level Convolutional Networks for Text Classification*)的论文，如图 3-11 所示。相较于之前的模型有两个主要的区别。

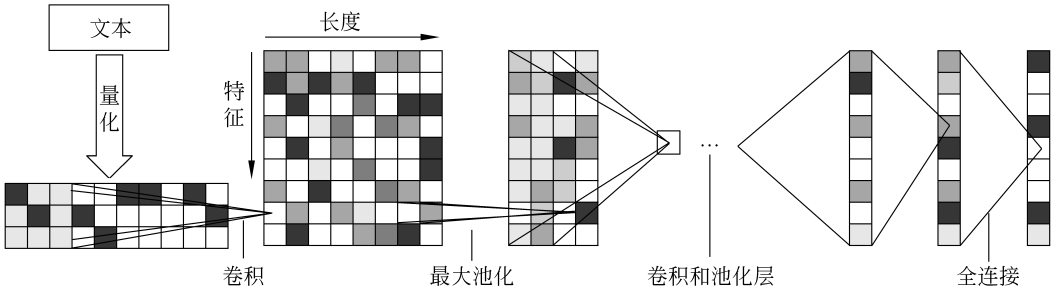

图 3-11　字符级卷积神经网络模型架构

- ✧ 输入的方式：Kim 使用了固定长度的词向量作为输入，而该模型使用字符级别的卷积操作。它把每个单词都看成是由字符组成的，因此将每个单词切分为字符序列，将字符转换成向量，并将字符序列作为输入。
- ✧ 模型结构：该模型使用了多个不同大小的卷积核，以提取不同大小的 n-gram 特征，并将特征通过池化层进行压缩。此外，该模型还使用了多个卷积池化层的组合，以增强模型的表征能力。

第三个模型 VDCNN(very deep convolutional networks for text classification)是一种用于文本分类任务的卷积神经网络模型，由 Conneau 等于 2016 年提出。相较于传统的卷积神经网络模型，VDCNN 主要有以下特点。

- ✧ 深度：VDCNN 非常深，它的深度可以根据任务自行设置。深度网络能够学习更加高层次的抽象特征，从而提高模型的分类性能。
- ✧ 残差连接：VDCNN 引入了深度残差连接结构，使得网络能够更好地学习到输入数据中的重要特征。

✧ 多种卷积核：VDCNN 使用了多种不同大小的卷积核，以提取不同长度的 n-gram 特征，从而能够更好地处理长文本。

✧ 金字塔池化：VDCNN 还使用了金字塔池化结构，通过分层的方式将输入数据进行压缩，从而提取出更加具有区分性的特征。

VDCNN 在多个文本分类任务上取得了很好的性能，特别是在长文本分类任务中表现优异。

3.3　循环神经网络

循环神经网络是一种递归神经网络，被广泛用于处理序列数据，如文本、语音和时间序列数据。与前馈神经网络不同，循环神经网络具有自循环的特性，使其能够捕获先前时刻的信息并将其传递到当前时刻，从而对序列进行建模。循环神经网络的结点通常称为循环单元，每个单元都包含一个内部状态(也称为记忆单元)，用于存储与序列相关的信息。每个时刻的输入和前一时刻的状态被输入到循环单元中，生成当前时刻的输出和更新后的状态。通过链式连接，序列中所有时刻的循环单元构成了一个递归结构，从而实现了序列的建模。

对循环神经网络的研究始于 20 世纪 80—90 年代，早期的循环神经网络模型主要用于语音识别、手写字符识别等任务。但由于梯度消失和梯度爆炸问题的存在，传统的循环神经网络在处理长序列时往往无法捕捉到序列中的长期依赖关系，限制了其应用范围。到了 21 世纪初期，随着深度学习的兴起，研究者们开始探索如何改进循环神经网络，使其能够更好地处理长序列。2003 年，Bengio 等提出了基于时间反向传播的训练算法，可以有效缓解梯度消失问题。之后，一些改进的循环神经网络模型也相继出现，如长短期记忆网络、门控循环单元等，这些模型都通过引入门控机制解决了梯度消失问题，使得循环神经网络成为深度学习算法中的重要组成部分。

3.3.1　不等长序列

前馈神经网络可以处理向量，卷积神经网络可以处理矩阵(它们被转换为向量)，那么如何处理长度不同的序列呢？如果讨论的是大小不同的图像，那么可以简单地重新缩放它们以匹配。例如，如果有一个[800×600]的图像和一个[1600×1200]的图像，很明显可以简单地调整其中一张图像的大小，有两个选择。第一种选择是使更大的画面变小，可以通过两种方式做到这一点：要么取 4 个像素的平均值，要么将它们最大池化；第二种选择，可以类似地通过插值[①]像素使图像更大。如果图像不能很好地缩放。例如，一个是[800×600]，另一个是[800×555]，可以简单地将图像向一个方向扩展。所做的变形不会影响图像处理，因为图像将保留大部分形状。但是，如果构建一个分类器来区分椭圆和圆形，然后调整图像大小，这会使圆形看起来像椭圆。请注意，当处理固定大小的矩阵时，可以将它们编码为具有固定长度的向量，每个向量元素对应于矩阵中的一个像素值。然而，当矩阵的大小不同时，

① 插值是一种计算机图形学和数字图像处理中广泛使用的技术，它涉及从已知像素值来估计新像素值的过程。当要将图像放大时，插值技术将使用已有像素之间的关系来估算新像素。常用的插值算法有双线性插值、双三次插值和 Lanczos 插值等。

无法直接将它们编码为固定长度的向量，并保持其矩阵属性。这是因为每个矩阵的行和列的长度可能不同，导致向量长度不同，难以进行有效的比较和分析。例如，如果所有图像都是[20×20]，那么可以将它们转换为大小为 400 的向量。这意味着图像第三行中的第 2 个像素是 400 维向量的 43 分量；如果有两个图像，一个[20×20]和一个[30×30]，那么维向量的第43 个分量(假设可以在这里以某种方式拟合一个维度)将是第 1 张图像第 3 行第 2 个像素和第 2 张图像第 2 行第 13 个像素的组合。但是，真正的问题是如何在神经网络中适应不同维度(400 和 900)的向量。到目前为止，我们所看到的一切都需要固定维度的向量。

在学习长度不等序列的问题中，音频处理是一个很好的例子。由于音频片段的长度不同，因此需要一种方法来处理不同长度的音频剪辑。当处理音频时，一种常见的方法是将音频片段划分为固定大小的时间段，然后将这些时间段编码为固定长度的特征向量。这种方法可以有效地处理不同长度的音频，但可能会损失一些信息。理论上，可以只取最长的，然后让所有其他的长度与那个长度相同，但就所需空间而言这是一种浪费。此外，在处理音频时，静音是一个很重要的问题。静音确实是语言的一部分，并且可以用于传达意义。例如，在演讲或音乐中，静默可能被用来传达情感或强调。因此，在训练集中，包含一些静默的内容标记为 1 的声音片段是完全合理的。然而，如果在音频片段的开头或结尾添加了长时间的静音剪辑，这可能会改变音频的含义，因此在这种情况下标签 1 可能不再适用。

所以问题是我们能做什么？答案是需要一种不同于之前所见的神经网络架构。目前为止，看到的每个神经网络都具有推动信息向前的连接，这就是为什么称它们为前馈神经网络的原因。事实证明，通过将输出作为输入反馈到层的连接来处理长度不等的序列，具有这种反馈回路的网络称为循环神经网络。在传统的神经网络中，每个输入和输出之间都是独立的，没有联系。但是，在处理序列数据(如语音、文本)时，输入和输出之间的联系是非常重要的。循环神经网络通过反馈机制来实现这种联系，即将当前时间步的输出作为下一个时间步的输入。然而，这种反馈机制也带来了梯度消失的问题。梯度消失是指在反向传播时，梯度(误差信号)随着时间步的增加而逐渐减小，最终消失。这是因为反向传播涉及多次乘法操作，如果乘数小于 1，则会导致梯度逐渐减小，甚至消失。使用共享权重的循环神经网络结构可以部分避免梯度消失的问题。因为在这种结构中，相同的权重被用于每个时间步的计算，这意味着梯度可以沿着同样的路径反向传播，从而更容易传递到较早的时间步骤。因此，这种结构可以处理更长的序列数据。

在循环神经网络出现之前，人们尝试使用感知器来处理序列数据。感知器是一种最简单的神经网络模型，它只有一层神经元，并且每个神经元只接收来自输入的信号，没有反馈连接。然而，感知器的局限性很快就被发现了。它只能处理线性可分的问题，对于非线性问题无法处理。因此，人们开始尝试制作多层感知器来解决这个问题。多层感知器可以通过增加神经元的数量和层数来学习更复杂的特征和模式。然而，当时人们并不知道如何训练多层感知器，因为反向传播算法还没有被广泛接受。在这个时期，人们开始探索一些理论上看起来比较自然的想法。例如，添加单层、添加多层和添加反馈循环，来尝试解决多层感知器的训练问题。其中，添加反馈循环的想法最终演化成了循环神经网络。但是，在反向传播算法被广泛接受之前，这些想法都没有被成功地应用到实际问题。直到 1986 年，反向传播算法被重新发现并广泛应用于神经网络的训练，这使得多层感知器的训练问题得到了有效解决。从此，神经网络开始在各领域得到广泛应用，包括自然语言处理、语音识别、图像识别

等。循环神经网络也得到了更广泛的应用，并成为处理序列数据的重要工具。

Hopfield网络是由物理学家John Hopfield于1982年提出的一种递归神经网络模型。它最初被设计用于模拟神经元之间的相互作用，以及用于解决优化问题。Hopfield网络采用一种称为“能量函数”的形式来表示神经元之间的相互作用，该能量函数与所要求解的优化问题的目标函数是等价的。Hopfield网络与今天认为的循环神经网络有所不同。Hopfield网络是一种无向图模型，神经元之间的连接权重是对称的，且不存在时间序列上的先后关系。在Hopfield网络中，神经元的状态是通过同步更新的方式进行的，而不是递归更新。此外，Hopfield网络也没有梯度下降的训练算法，它的权重是通过计算能量函数的梯度来得到的。尽管Hopfield网络不是一个循环神经网络，但它的思想对于后来的神经网络发展有着重要的启示作用。Hopfield网络表明，神经元之间的相互作用可以通过能量函数的形式来描述，并且可以通过优化能量函数来解决问题。此外，Hopfield网络的概念也促进了神经网络领域对于递归和循环神经网络的研究。1997年，Hochreiter和Schmidhuber发明的长短期记忆网络在递归神经网络发展历程中的重要性和广泛应用。长短期记忆网络是一种特殊的递归神经网络，它能够解决传统循环神经网络中存在的梯度消失和梯度爆炸等问题，从而更好地处理长序列数据。长短期记忆网络中的关键组成部分是“门控单元”，它能够控制信息的流动和保留，使得长短期记忆网络能够记住需要的信息，并遗忘不需要的信息。LSTM的发明标志着递归神经网络领域的一个重要里程碑，它在自然语言处理、语音识别、图像处理等多个领域都取得了重要的进展和应用。例如，在机器翻译领域，长短期记忆网络已经成为先进的机器翻译模型之一，其表现优于传统的n-gram模型和基于规则的机器翻译模型。在语音识别领域，长短期记忆网络也被广泛应用于声学建模和语音合成等任务，取得了显著的性能提升。除此之外，长短期记忆网络还被用于视频分析、图像标注、自然语言推理等多个领域。本节将引入必要的概念来详细解释长短期记忆网络。

3.3.2 循环连接的构成

现在看看循环神经网络是如何工作的。还记得梯度消失的问题吗？梯度消失问题指的是在神经网络中，当反向传播算法计算梯度时，由于梯度的连乘效应，导致梯度在网络较深的层中逐渐减小，并在最终层中变得非常小或者趋于0，这使得该层的权重几乎不会被更新。这会导致该层无法学习到有用的特征，从而影响整个网络的性能。卷积神经网络通过使用共享权重的卷积层和池化层，减少了神经网络中的参数数量，从而在处理图像等空间结构数据时表现出色。卷积神经网络的卷积层使用一个固定大小的卷积核对输入数据进行卷积运算，从而提取出输入数据中的特征。卷积核的权重被共享，这意味着它们在整个图像上进行相同的卷积运算，从而减少了网络中的参数数量。此外，卷积神经网络还使用池化层来减少数据的空间大小，进一步降低了参数数量。然而，卷积神经网络确实具有特定的架构，更适合于处理具有空间结构的数据，如图像、音频和视频等。对于一些其他类型的数据，如自然语言等序列数据，卷积神经网络可能不太适用。在这些情况下，循环神经网络和其改进模型可能更适合处理序列数据。

循环神经网络的工作原理不是向简单的前馈神经网络添加新层，而是通过在隐含层上添加循环连接。如图3-12(a)所示，这里显示了一个简单前馈神经网络；如图3-12(b)所示，这里显示了如何将循环连接添加到图3-12(a)中的简单前馈神经网络。在一个简单的前馈