第六章　卷积神经网络

当处理图像时，全连接的前馈神经网络会存在以下两个问题：

1. 图像不能太大。比如，输入图像大小为$100 \times 100 \times 3$（即图像高度为100，宽度为100，3个颜色通道RGB）。在全连接前馈神经网络中，第一个隐藏层的每个神经元到输入层都有$100 * 100 * 3 = 30,000$个相互独立的连接，每个连接都对应一个权重参数。随着隐藏层神经元数量的增多，参数的规模也会极具增加。这会导致整个神经网络的训练效率会非常低，也很容易出现过拟合。
2. 难以处理图像不变性。自然图像中的物体都具有局部不变性特征，比如在尺度缩放、平移、旋转等操作不影响人们对它的正确识别。而全连接的前馈神经网络很难提取这些特征，一般需要进行数据增强来提高性能。

卷积神经网络（Convolutional Neural Networks，CNN）是受生物学上**感受野**（Receptive Field）的机制而提出的一种前馈神经网络。

感受野主要是指听觉、视觉等神经系统中一些神经元的特性，即神经元只接受其所支配的刺激区域内的信号。在视觉神经系统中，视觉皮层中的神经细胞的输出依赖于视网膜上的光感受器。视网膜上的光感受器受刺激兴奋时，将神经冲动信号传到视觉皮层，但不是所有视觉皮层中的神经元都会接受这些信号。一个神经元的感受野是指视网膜上的特定区域，只有这个区域内的刺激才能够激活该神经元。David Hubel和Torsten Wiesel在1959年发现，在猫的初级视觉皮层中存在两种细胞：简单细胞和复杂细胞，这两种细胞承担不同层次的视觉感知功能 [Hubel and Wiesel, 1959, 1962]。简单细胞的感受野是狭长型的，每个简单细胞只对感受野中特定角度（orientation）的光带敏感，而复杂细胞对于感受野中以特定方向（direction）移动的某种角度（orientation）的光带敏感。

David Hubel和Torsten Wiesel在此方面的贡献，与1981年获得诺贝尔生理学或医学奖。

受此启发，1980年，Kunihiko Fukishima(福岛邦彦）提出了一种带卷积和子采样操作的多层神经网络：新知机（Neocognitron）[Fukushima, 1980]。但当时还没有反向传播算法，新知机采用了无监督学习的方式来训练。Yann LeCun在1989年将反向传播算法引入了卷积神经网络 [LeCun et al., 1989]，并在手写体数字识别上取得了很大的成功 [LeCun et al., 1998]。

目前的卷积神经网络一般采用交替使用卷积层和最大值池化层，然后在顶端使用多层全连接的前馈神经网络。训练过程使用反向传播算法。卷积神经网络有三个结构上的特性：局部连接，权重共享以及次采样。这些特性使得卷积神经网络具有一定程度上的平移、缩放和扭曲不变性。和前馈神经网络相比，卷积神经网络的参数更少。在图像识别任务上，基于卷积神经网络模型的准确率也远远超出了一般的神经网络模型。

6.1 卷积

卷积，也叫摺积，是分析数学中一种重要的运算。我们这里只考虑离散序列的情况。

6.1.1 一维场合

一维卷积经常用在信号处理中。给定一个输入信号序列$x_t, t = 1, \cdots, n$，和滤波器 $f_t, t = 1, \cdots, m$，一般情况下滤波器的长度m远小于信号序列长度n。

卷积的输出为：

$$y_t = \sum_{k=1}^{m} f_k \cdot x_{t-k+1}. \tag{6.1}$$

当滤波器$f_t = 1/n$时，卷积相当于信号序列的移动平均。

卷积的结果按输出长度不同可以分为三类：

- **窄卷积**：输出长度$n - m + 1$，不补零。
- **宽卷积**：输出长度$n+m-1$，对于不在$[1, n]$范围之外的x_t用零补齐（zero-padding）。（Padding=m-1）
- **等长卷积**：输出长度n，对于不在$[1, n]$范围之外的x_t用零补齐（zero-padding）。(Padding=(m-1)/2)

在这里除了特别声明，我们一般说的卷积默认为**窄卷积**。

6.1.2 两维场合

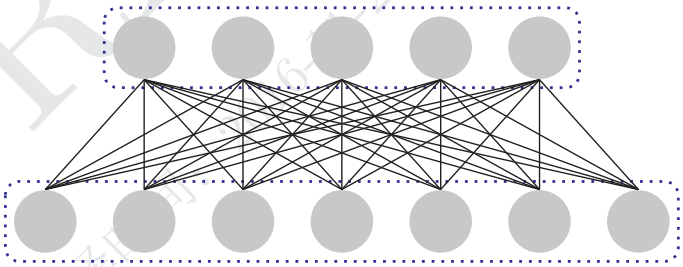
两维卷积经常用在图像处理中。给定一个图像 $x_{ij}, 1 \leq i \leq M, 1 \leq j \leq N$，和滤波器 $f_{ij}, 1 \leq i \leq m, 1 \leq j \leq n$，一般 $m << M, n << N$。

卷积的输出为：

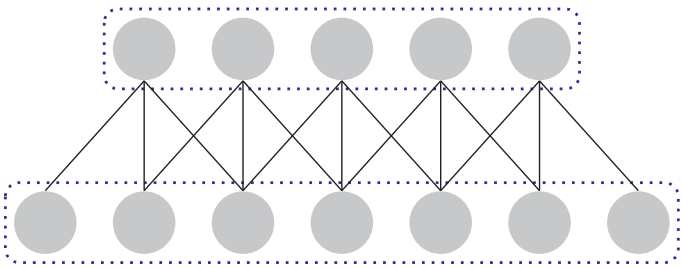
$$y_{ij} = \sum_{u=1}^{m} \sum_{v=1}^{n} f_{uv} \cdot x_{i-u+1,j-v+1}. \tag{6.2}$$

在图像处理中，常用的均值滤波（mean filter）就是当前位置的像素值设为滤波器窗口中所有像素的平均值，也就是 $f_{uv} = \frac{1}{mn}$。

6.2 卷积层：用卷积来代替全连接



(a) 全连接层



(b) 卷积层

图 6.1: 全连接层和卷积层

在全连接前馈神经网络中，如果第l层有n^l个神经元，第$l-1$层有$n^{(l-1)}$个神经元，连接边有$n^{(l)} \times n^{(l-1)}$个，也就是权重矩阵有$n^{(l)} \times n^{(l-1)}$个参数。当m和n都很大时，权重矩阵的参数非常多，训练的效率会非常低。

如果采用卷积来代替全连接，第l层的每一个神经元都只和第$l-1$层的一个局部窗口内的神经元相连，构成一个局部连接网络。第l层的第i个神经元的输入定义为：

$$a_i^{(l)} = f(\sum_{j=1}^{m} w_j^{(l)} \cdot a_{i-j+m}^{(l-1)} + b^{(l)}), \tag{6.3}$$

$$= f(\mathbf{w}^{(l)} \cdot \mathbf{a}_{(i+m-1):i}^{(l-1)} + b_i), \tag{6.4}$$

其中，$\mathbf{w}^{(l)} \in \mathbb{R}^m$为$m$维的滤波器，$\mathbf{a}_{(i+m-1):i}^{(l)} = [a_{(i+m-1)}^{(l)}, \cdots, a_i^{(l)}]^T$。这里，$a^{(l)}$的下标从1开始，我们这里的卷积公式和原始的公式中$\mathbf{a}$的下标有所不同。

上述公式也可以写为：

$$\mathbf{a}^{(l)} = f(\mathbf{w}^{(l)} \otimes \mathbf{a}^{(l-1)} + b^{(l)}), \tag{6.5}$$

$\otimes$表示卷积运算。

从公式6.5可以看出，$\mathbf{w}^{(l)}$对于所有的神经元都是相同的。这也是卷积层的另外一个特性是**权值共享**。这样，在卷积层里，我们只需要$m+1$个参数。另外，第$l+1$层的神经元个数不是任意选择的，而是满足$n^{(l+1)} = n^{(l)} - m + 1$。

上面是一维的卷积层，下面我们来看下两维的情况。在图像处理中，图像是以二维矩阵的形式输入到神经网络中，因此我们需要二维卷积。假设$x^{(l)} \in \mathbb{R}^{(w_l \times h_l)}$和$x^{(l-1)} \in \mathbb{R}^{(w_{l-1} \times h_{l-1})}$分别是第$l$层和第$l-1$层的神经元活性。$X^{(l)}$的每一个元素为：

$$X_{s,t}^{(l)} = f\left(\sum_{i=1}^{u}\sum_{j=1}^{v} W_{i,j}^{(l)} \cdot X_{s-i+u,t-j+v}^{(l-1)} + b^{(l)}\right), \tag{6.6}$$

其中，$W^{(l)}\mathbb{R}^{u \times v}$为两维的滤波器，$B$为偏置矩阵。第$l-1$层的神经元个数为$(w_l \times h_l)$，并且$w_l = w_{l-1} - u + 1$，$h_l = h_{l-1} - v + 1$。

也可以写为：

$$X^{(l)} = f\left(W^{(l)} \otimes X^{(l-1)} + b^{(l)}\right), \tag{6.7}$$

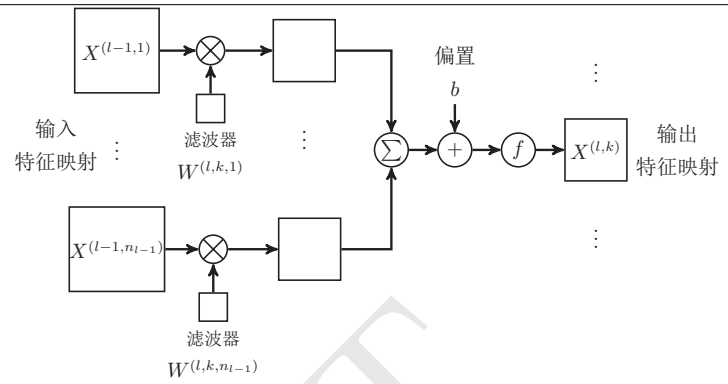


图 6.2: 两维卷积层的映射关系

为了增强卷积层的表示能力，我们可以使用K组不同的滤波器来得到K组输出。每一组输出都共享一个滤波器。如果我们把滤波器看成一个特征提取器，每一组输出都可以看成是输入图像经过一个特征抽取后得到的特征。因此，在卷积神经网络中每一组输出也叫作一组**特征映射**（Feature Map）。

不失一般性，我们假设第$l-1$层的特征映射组数为n_{l-1}，每组特征映射的大小为$m_{l-1}=w_{l-1}\times h_{l-1}$。第$l-1$层的总神经元数：$n_{l-1}\times m_{l-1}$。第$l$层的特征映射组数为$n_l$。如果假设第$l$层的每一组特征映射$X^{(l,k)}$的输入为第$l-1$层的所有组特征映射。

第l层的第k组特征映射$X^{(l,k)}$为：

$$X^{(l,k)}=f\left(\sum_{p=1}^{n_{l-1}}\left(W^{(l,k,p)}\otimes X^{(l-1,p)}\right)+b^{(l,k)}\right), \tag{6.8}$$

其中，$W^{(l,k,p)}$表示第$l-1$层的第p组特征向量到第l层的第k组特征映射所需的滤波器。

第l层的每一组特征映射都需要n_{l-1}个滤波器以及一个偏置b。假设每个滤波器的大小为$u\times v$，那么共需要$n_l\times n_{l-1}\times(u\times v)+n_l$。

这样，我们在第$l+1$层就得到n_l组特征映射，每一组特征映射的大小为$m_l=w_{l-1}-u+1\times h_{l-1}-v+1$，总的神经元个数为$n_l\times m_l$。图6.2给出了公式6.8的可视化映射关系。

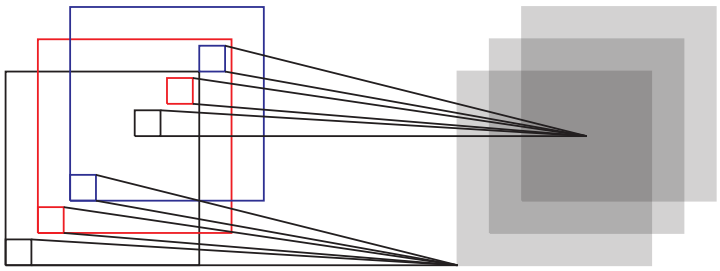


图 6.3: 两维卷积层

连接表 公式6.8中，第$l-1$层的所有特征映射都经过滤波器得到一个第l层的一组特征映射$X^{(l,k)}$。也就是说，第l层的每一组特征映射都依赖于第l层的所有特征映射，相当于不同层的特征映射之间是全连接的关系。实际上，这种全连接关系不是必须的。我们可以让第l层的每一组特征映射都依赖于前一层的少数几组特征映射。这样，我们定义一个**连接表**T来描述不同层的特征映射之间的连接关系。如果第l层的第k组特征映射依赖于前一层的第p组特征映射，则$T_{p,k}=1$，否则为0。

$$X^{(l,k)}=f\left(\sum_{\substack{p,\\ T_{p,k}=1}}\left(W^{(l,k,p)}\otimes X^{(l-1,p)}\right)+b^{(l,k)}\right) \tag{6.9}$$

这样，假如连接表T的非零个数为K，每个滤波器的大小为$u\times v$，那么共需要$K\times(u\times v)+n_l$参数。

卷积层的作用是提取一个局部区域的特征，每一个滤波器相当于一个特征提取器。图6.3给出了两维卷积层示例。

6.3 子采样层

卷积层虽然可以显著减少连接的个数，但是每一个特征映射的神经元个数并没有显著减少。这样，如果后面接一个分类器，分类器的输入维数依然很高，很容易出现过拟合。为了解决这个问题，在卷积神经网络一般会在卷积层之后

再加上一个池化（Pooling）操作，也就是子采样（Subsampling），构成一个子采样层。子采样层可以来大大降低特征的维数，避免过拟合。

对于卷积层得到的一个特征映射 $X^{(l)}$，我们可以将 $X^{(l)}$ 划分为很多区域 $R_k, k = 1, \cdots, K$，这些区域可以重叠，也可以不重叠。一个子采样函数 $\mathbf{down}(\cdots)$ 定义为：

$$X_k^{(l+1)} = f(Z_k^{(l+1)}), \tag{6.10}$$

$$= f\left(w^{(l+1)} \cdot \mathbf{down}(R_k) + b^{(l+1)}\right), \tag{6.11}$$

其中，$w^{(l+1)}$ 和 $b^{(l+1)}$ 分别是可训练的权重和偏置参数。

$$X^{(l+1)} = f(Z^{(l+1)}), \tag{6.12}$$

$$= f\left(w^{(l+1)} \cdot \mathbf{down}(X^l) + b^{(l+1)}\right), \tag{6.13}$$

$\mathbf{down}(X^l)$ 是指子采样后的特征映射。

子采样函数 $\mathbf{down}(\cdot)$ 一般是取区域内所有神经元的最大值（Maximum Pooling）或平均值（Average Pooling）。

$$pool_{max}(R_k) = \max_{i \in R_k} a_i \tag{6.14}$$

$$pool_{avg}(R_k) = \frac{1}{|R_k|} \sum_{i \in R_k} a_i. \tag{6.15}$$

子采样的作用还在于可以使得下一层的神经元对一些小的形态改变保持不变性，并拥有更大的感受野。

6.4 卷积神经网络示例：LeNet-5

下面我们来看一个具体的深层卷积神经网络：LeNet-5[LeCun et al., 1998]。LeNet-5 虽然提出时间比较早，但是是一个非常成功的神经网络模型。基于 LeNet-5 的手写数字识别系统在 90 年代被美国很多银行使用，用来识别支票上面的手写数字。LeNet-5 的网络结构如图6.4所示。

不计输入层，LeNet-5 共有 7 层，每一层的结构为：

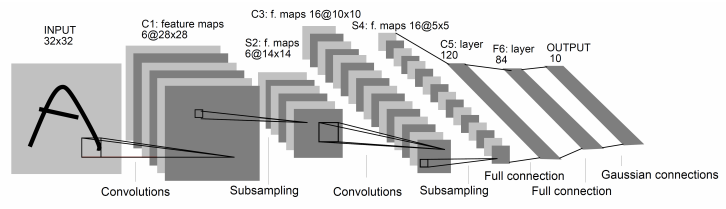


图 6.4: LeNet-5 网络结构。图片来源：[LeCun et al., 1998]

1. 输入层：输入图像大小为 $32 \times 32 = 1024$。
2. C1 层：这一层是卷积层。滤波器的大小是 $5 \times 5 = 25$，共有 6 个滤波器。得到 6 组大小为 $28 \times 28 = 784$ 的特征映射。因此，C1 层的神经元个数为 $6 \times 784 = 4,704$。可训练参数个数为 $6 \times 25 + 6 = 156$。连接数为 $156 \times 784 = 122,304$（包括偏置在内，下同）。
3. S2 层：这一层为子采样层。由 C1 层每组特征映射中的 2×2 邻域点次采样为 1 个点，也就是 4 个数的平均。这一层的神经元个数为 $14 \times 14 = 196$。可训练参数个数为 $6 \times (1 + 1) = 12$。连接数为 $6 \times 196 \times (4 + 1) = 5,880$（包括偏置的连接）
4. C3 层：这一层是卷积层。由于 S2 层也有多组特征映射，需要一个连接表来定义不同层特征映射之间的依赖关系。LeNet-5 的连接表如图6.5所示。这样的连接机制的基本假设是：C3 层的最开始的 6 个特征映射依赖于 S2 层的特征映射的每 3 个连续子集。接下来的 6 个特征映射依赖于 S2 层的特征映射的每 4 个连续子集。再接下来的 3 个特征映射依赖于 S2 层的特征映射的每 4 个不连续子集。最后一个特征映射依赖于 S2 层的所有特征映射。这样共有 60 个滤波器，大小是 $5 \times 5 = 25$。得到 16 组大小为 $10 \times 10 = 100$ 的特征映射。C3 层的神经元个数为 $16 \times 100 = 1,600$。可训练参数个数为 $(60 \times 25 + 16 = 1,516$。连接数为 $1,516 \times 100 = 151,600$。
5. S4 层：这一层是一个子采样层，由 2×2 邻域点次采样为 1 个点，得到 16 组 5×5 大小的特征映射。可训练参数个数为 $16 \times 2 = 32$。连接数为 $16 \times (4 + 1) = 2000$。
6. C5 层：是一个卷积层，得到 120 组大小为 1×1 的特征映射。每个特征映射与 S4 层的全部特征映射相连。有 $120 \times 16 = 1,920$ 个滤波器，大小是 $5 \times 5 = 25$。C5 层的神经元个数为 120，可训练参数个数为 $1,920 \times 25 + 120 = 48,120$。连接数为 $120 \times (16 \times 25 + 1) = 48,120$。

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

图 6.5: LeNet-5 中 C3 层的连接表。图片来源：[LeCun et al., 1998]

7. F6 层：是一个全连接层，有 84 个神经元，可训练参数个数为 $84\times(120+1)=10,164$。连接数和可训练参数个数相同，为 $10,164$。
8. 输出层：输出层由 10 个欧氏径向基函数（Radial Basis Function，RBF）函数组成。这里不再详述。

6.5 梯度计算

在全连接前馈神经网络中，目标函数关于第 l 层的神经元 $\mathbf{z}^{(l)}$ 的梯度为

$$\delta^{(l)} \triangleq \frac{\partial J(W,\mathbf{b};\mathbf{x},y)}{\partial \mathbf{z}^{(l)}} \tag{6.16}$$

$$= f_l'(\mathbf{z}^{(l)}) \odot (W^{(l+1)})^T \delta^{(l+1)} \tag{6.17}$$

在卷积神经网络中，每一个卷积层后都接着一个子采样层，然后不断重复。所以我们需要分别来看下卷积层和子采样层的梯度。

6.5.1 卷积层的梯度

我们假定卷积层为 l 层，子采样层为 $l+1$ 层。因为子采样层是下采样操作，$l+1$ 层的一个神经元的误差项 δ 对应于卷积层（上一层）的相应特征映射的一个区域。l 层的第 k 个特征映射中的每个神经元都有一条边和 $l+1$ 层的第 k 个特征映射中的一个神经元相连。根据链式法则，第 l 层的一个特征映射的误差项 $\delta^{(l,k)}$，只需要将 $l+1$ 层对应特征映射的误差项 $\delta^{(l+1,k)}$ 进行上采样操作（和第 l 层的大小一样），再和 l 层特征映射的激活值偏导数逐元素相乘，再乘上权重 $w^{(l+1,k)}$，就得到了 $\delta^{(l,k)}$。

第l层的第k个特征映射的误差项$\delta^{(l,k)}$的具体推导过程如下：

$$\delta^{(l,k)} \triangleq \frac{\partial J(W,\mathbf{b};X,y)}{\partial Z^{(l,k)}} \tag{6.18}$$

$$= \frac{\partial X^{(l,k)}}{\partial Z^{(l,k)}} \cdot \frac{\partial Z^{(l+1,k)}}{\partial X^{(l,k)}} \cdot \frac{\partial J(W,\mathbf{b};X,y)}{\partial Z^{(l+1,k)}} \tag{6.19}$$

$$= f_l'(Z^{(l)}) \odot \left(\mathbf{up}\left(w^{(l+1,k)}\delta^{(l+1,k)}\right)\right) \tag{6.20}$$

$$= w^{(l+1,k)}\left(f_l'(Z^{(l)}) \odot \mathbf{up}(\delta^{(l+1,k)})\right), \tag{6.21}$$

其中，$\mathbf{up}$为上采用函数（Upsampling）。

在得到第l层的第k个特征映射的误差项$\delta^{(l,k)}$，目标函数关于第l层的第k个特征映射神经元滤波器$W_{i,j}^{(l,k,p)}$的梯度

$$\frac{\partial J(W,\mathbf{b};X,y)}{\partial W_{i,j}^{(l,k,p)}} = \sum_{s=1}^{w_l}\sum_{t=1}^{h_l}\left(X_{s-i+u,t-j+v}^{(l-1,p)} \cdot (\delta^{(l,k)})_{s,t}\right) \tag{6.22}$$

$$= \sum_{s=1}^{w_l}\sum_{t=1}^{h_l}\left(X_{(u-i)-s,(v-j)-t}^{(l-1,p)} \cdot \left(\mathbf{rot180}(\delta^{(l,k)})\right)_{s,t}\right). \tag{6.23}$$

公式6.23也刚好是卷积形式，因此目标函数关于第l层的第k个特征映射神经元滤波器$W^{(l,k,p)}$的梯度可以写为：

$$\frac{\partial J(W,\mathbf{b};X,y)}{\partial W^{(l,k,p)}} = \mathbf{rot180}\left(X^{(l-1,p)} \otimes \mathbf{rot180}(\delta^{(l,k)})\right). \tag{6.24}$$

目标函数关于第l层的第k个特征映射的偏置$b^{(l)}$的梯度可以写为：

$$\frac{\partial J(W,\mathbf{b};X,y)}{\partial b^{(l,k)}} = \sum_{i,j}(\delta^{(l,k)})_{i,j}. \tag{6.25}$$

6.5.2 子采样层的梯度

我们假定子采样层为l层，$l+1$层为卷积层。因为子采样层是下采样操作，$l+1$层的一个神经元的误差项δ对应于卷积层（上一层）的相应特征映射的一个区域。

$$Z^{(l+1,k)} = \sum_{\substack{p,\\ T_{p,k}=1}}\left(W^{(l+1,k,p)} \otimes X^{(l,p)}\right) + b^{(l+1,k)} \tag{6.26}$$

第l层的第k个特征映射的误差项$\delta^{(l,k)}$的具体推导过程如下：

$$\delta^{(l,k)} \triangleq \frac{\partial J(W,\mathbf{b};X,y)}{\partial Z^{(l,k)}} \tag{6.27}$$

$$= \frac{\partial X^{(l,k)}}{\partial Z^{(l,k)}} \cdot \frac{\partial Z^{(l+1,k)}}{\partial X^{(l,k)}} \cdot \frac{\partial J(W,\mathbf{b};X,y)}{\partial Z^{(l+1,k)}} \tag{6.28}$$

$$= f_l'(Z^{(l)}) \odot \left(\sum_{\substack{p,\\ T_{p,k}=1}} \left(\delta^{(l+1,p)} \tilde{\otimes} \mathbf{rot180}(W^{(l,k,p)}) \right) \right). \tag{6.29}$$

其中，$\tilde{\otimes}$为宽卷积。

公式6.23也刚好是卷积形式，因此目标函数关于第l层的第k个特征映射神经元滤波器$W^{(l,k,p)}$的梯度可以写为：

$$\frac{\partial J(W,\mathbf{b};X,y)}{\partial w^{(l,k)}} = \sum_{i,j} \left(\mathbf{down}(X^{(l-1,k)}) \cdot \delta^{(l,k)} \right)_{i,j}. \tag{6.30}$$

目标函数关于第l层的第k个特征映射的偏置$b^{(l)}$的梯度可以写为：

$$\frac{\partial J(W,\mathbf{b};X,y)}{\partial b^{(l,k)}} = \sum_{i,j} (\delta^{(l,k)})_{i,j}. \tag{6.31}$$

6.6 总结和深入阅读

参考文献

Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

James A Anderson and Edward Rosenfeld. *Talking nets: An oral history of neural networks*. MiT Press, 2000.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166, 1994.

Yoshua Bengio, Jean-Sébastien Senécal, et al. Quick training of probabilistic neural nets by importance sampling. In *AISTATS Conference*, 2003.

Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. Advances in optimizing recurrent networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8624–8628. IEEE, 2013.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, 2010.

C.M. Bishop. *Pattern recognition and machine learning*. Springer New York., 2006.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

P.F. Brown, P.V. Desouza, R.L. Mercer, V.J.D. Pietra, and J.C. Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4): 467–479, 1992.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, 2002.

Hal Daumé III. A course in machine learning. http://ciml.info/. [Online].

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley, New York, 2nd edition, 2001. ISBN 0471056693.

Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. *Advances in Neural Information Processing Systems*, pages 472–478, 2001.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Yoav Freund and Robert E Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296, 1999.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256, 2010.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

Ian Goodfellow, Aaron Courville, and Yoshua Bengio. Deep learning. Book in preparation for MIT Press, 2015. URL http://goodfeli.github.io/dlbook/.

Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron C Courville, and Yoshua Bengio. Maxout networks. In *ICML*, volume 28, pages 1319–1327, 2013.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79 (8):2554–2558, 1982.

David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574–591, 1959.

David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

M.I. Jordan. *Learning in Graphical Models*. Kluwer Academic Publishers, 1998.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, volume 30, 2013.

Ryan McDonald, Keith Hall, and Gideon Mann. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464. Association for Computational Linguistics, 2010.

Marvin Minsky and Papert Seymour. Perceptrons. 1969.

Marvin L Minsky and Seymour A Papert. *Perceptrons - Expanded Edition: An Introduction to Computational Geometry*. MIT press Boston, MA:, 1987.

T.M. Mitchell. *Machine learning*. Burr Ridge, IL: McGraw Hill, 1997.

Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*, 2012.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

Albert BJ Novikoff. On convergence proofs for perceptrons. Technical report, DTIC Document, 1963.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3, 1988.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. 1974.

Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

DE Rumelhart GE Hinton RJ Williams and GE Hinton. Learning representations by back-propagating errors. *Nature*, pages 323–533, 1986.

Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.