每个标题前的序号与《动手学深度学习》一书中的相对应

3.线性神经网络

3.1.线性回归

我们希望根据房屋的面积（平方英尺）和房龄（年）来估算房屋价格（美元）。

线性模型

- 线性假设是指目标（房屋价格）可以表示为特征（面积和房龄）的加权和，w为权重，b为偏移量，如下面的式子：

$$\text{price} = w_{\text{area}} \cdot \text{area} + w_{\text{age}} \cdot \text{age} + b.$$

- 用矩阵和向量简洁表达

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b.$$

损失函数

损失函数（loss function）能够量化目标的实际值与预测值之间的差距。

- 平方误差函数

$$l^{(i)}(\mathbf{w}, b) = \frac{1}{2}\left(\hat{y}^{(i)} - y^{(i)}\right)^2.$$

- 为了度量模型在整个数据集上的质量，我们需计算在训练集n个样本上的损失均值（均方误差损失）。

$$L(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n} l^{(i)}(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)}\right)^2.$$

- 在训练模型时，我们希望寻找一组参数（w，b）， 这组参数能最小化在所有训练样本上的总损失。

$$\mathbf{w}^*, b^* = \operatorname*{argmin}_{\mathbf{w},b} L(\mathbf{w}, b).$$

随机梯度下降

通过不断地在损失函数递减的方向上更新参数来降低误差。

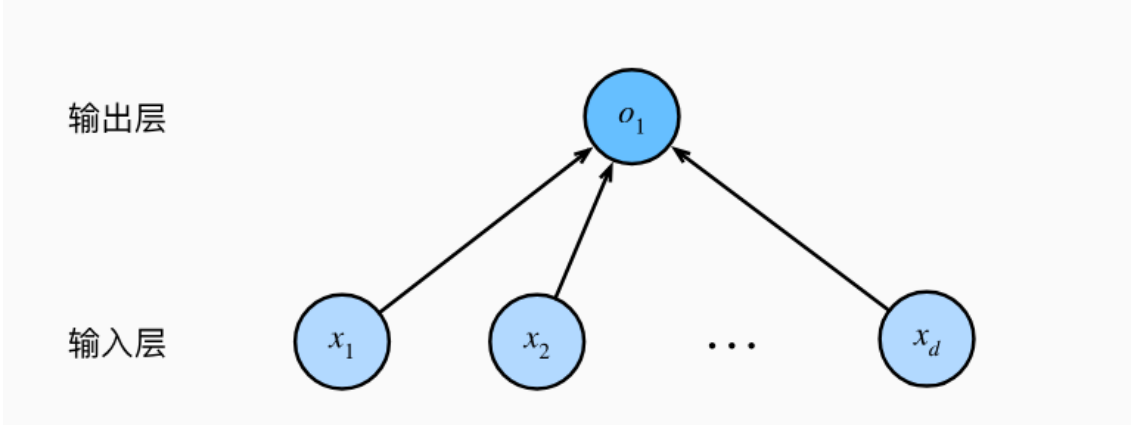
- 每次迭代中，我们随机抽样一个小批量，计算小批量的平均损失关于模型参数的梯度，将梯度乘以一个预先确定的正数（学习率），并从当前参数的值中减掉。

$$(\mathbf{w}, b) \leftarrow (\mathbf{w}, b) - \frac{\eta}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \partial_{(\mathbf{w},b)} l^{(i)}(\mathbf{w}, b).$$

- 调参， 批量大小和学习率的值通常是手动预先指定，称为超参数，调参即根据在验证数据集上的训练迭代结果调整超参数。

神经网络图

- 单层神经网络，特征维数，即输入层的输入数为d，输出数为1，层数为1（不考虑输入层）。



- 对于线性回归，每个输入都与每个输出相连， 我们将这种变换称为全连接层（fully-connected layer）或称为稠密层

3.4.softmax回归

回归可以用于预测多少的问题，我们也对分类问题感兴趣，判断图片属于{狗，猫，鸡}中的哪一个。softmax回归是一个线性模型。

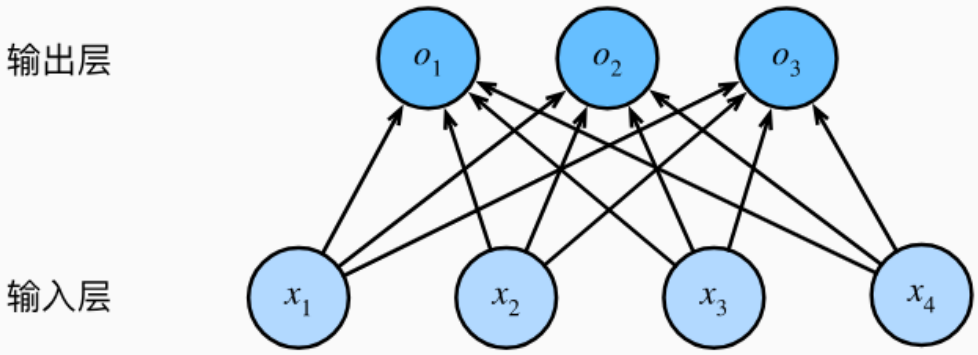
分类问题

- 独热编码，独热编码是一个向量，它的分量和类别一样多。例如{狗，猫，鸡}有三个类别，则标签y将是一个三维向量，（1，0，0）对应于“猫”，（0，1，0）对应于“鸡”，(0，0，1)对应于“狗”，即y={（1，0，0），（0，1，0），（0，0，1）}。

网络架构

- 为了估计所有可能类别的条件概率，我们需要一个有多个输出的模型，每个类别对应一个输出，每个输出对应一个仿射函数。softmax回归的输出层也是全连接层。

$$
\begin{aligned}
o_1 &= x_1 w_{11} + x_2 w_{12} + x_3 w_{13} + x_4 w_{14} + b_1, \\
o_2 &= x_1 w_{21} + x_2 w_{22} + x_3 w_{23} + x_4 w_{24} + b_2, \\
o_3 &= x_1 w_{31} + x_2 w_{32} + x_3 w_{33} + x_4 w_{34} + b_3.
\end{aligned}
$$



softmax运算

- softmax函数能够将未规范化的预测变换为非负数并且总和为1。首先对每个未规范化的预测求幂，这样可以确保输出非负。 为了确保最终输出的概率值总和为1，我们再让每个求幂后的结果除以它们的总和。

$$
\hat{\mathbf{y}} = \mathrm{softmax}(\mathbf{o}) \quad \text{其中} \quad \hat{y}_j = \frac{\exp(o_j)}{\sum_k \exp(o_k)}
$$

小批量样本矢量化

- 为了提高计算效率并且充分利用GPU，我们通常会对小批量样本的数据执行矢量计算。 假设我们读取了一个批量的样本**X**， 其中特征维度（输入数量）为d，批量大小为n。 此外，假设我们在输出中有q个类别。那么小批量样本的特征为**X**（n行d列）， 权重为**W**（d行q列）， 偏置为**b**（1行q列）。

$$\mathbf{O} = \mathbf{X}\mathbf{W} + \mathbf{b},$$
$$\hat{\mathbf{Y}} = \mathrm{softmax}(\mathbf{O}).$$

交叉熵损失函数

- 对于样本i，我们构造向量$\mathbf{y}^{(i)}$（q维向量，显然q为输入维数），使其第$y^{(i)}$（跟前面的y区分，1 <= $y^{(i)}$ <= q）个元素为1，其余为0。这样我们的训练目标可以设为使预测概率分布$\hat{y}^{(i)}$尽可能接近真实的标签概率分布$\mathbf{y}^{(i)}$。其中带下标的$y_j^{(i)}$是向量$y^{(i)}$中非0即1的元素，需要注意将它与样本i类别的离散数值，即不带下标的$y^{(i)}$区分。

$$H\left(\boldsymbol{y}^{(i)}, \hat{\boldsymbol{y}}^{(i)}\right) = -\sum_{j=1}^{q} y_j^{(i)} \log \hat{y}_j^{(i)},$$

- 假设训练数据集的样本数为n，交叉熵损失函数定义为

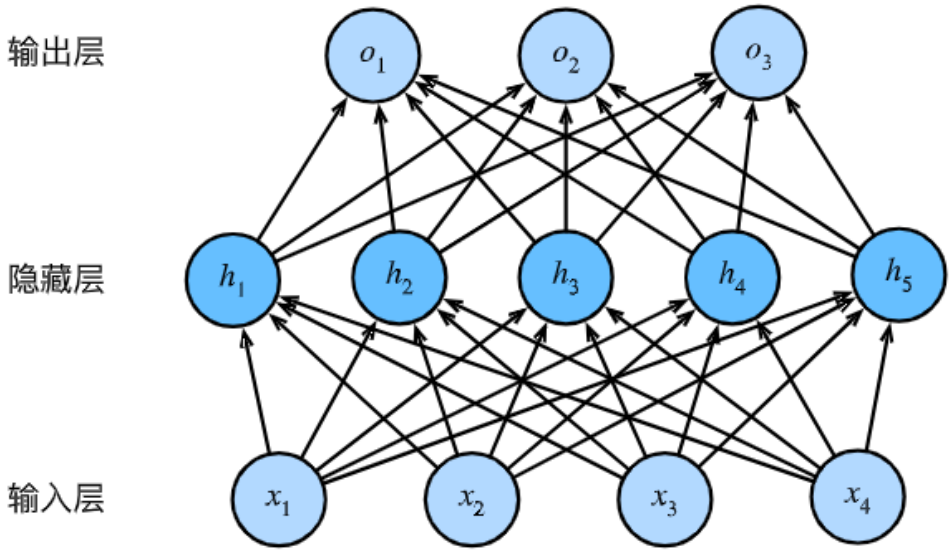
$$\ell(\boldsymbol{\Theta}) = \frac{1}{n} \sum_{i=1}^{n} H\left(\boldsymbol{y}^{(i)}, \hat{\boldsymbol{y}}^{(i)}\right),$$

4.多层感知机

4.1.多层感知机

隐藏层

- 多层感知机在单层神经网络的基础上引入了一到多个隐藏层（hidden layer）。隐藏层位于输入层和输出层之间。



带有隐藏层的多层感知机。它含有一个隐藏层，该层中有5个隐藏单元

- 给定一个小批量样本**X** （n行d列），其批量大小为n，输入个数为d。假设多层感知机只有一个隐藏层，其中隐藏单元个数为h。记隐藏层的输出（也称为隐藏层变量或隐藏变量）为**H**（n行h列）。因为隐藏层和输出层均是全连接层，可以设隐藏层的权重参数和偏差参数分别为$\mathbf{W}_h$（d行h列） 和 $\mathbf{b}_h$ (1行h列），输出层的权重和偏差参数分别为$\mathbf{W}_o$ （h行q列）和$\mathbf{b}_o$（1行q列），输出**O**(n行q列)的计算。

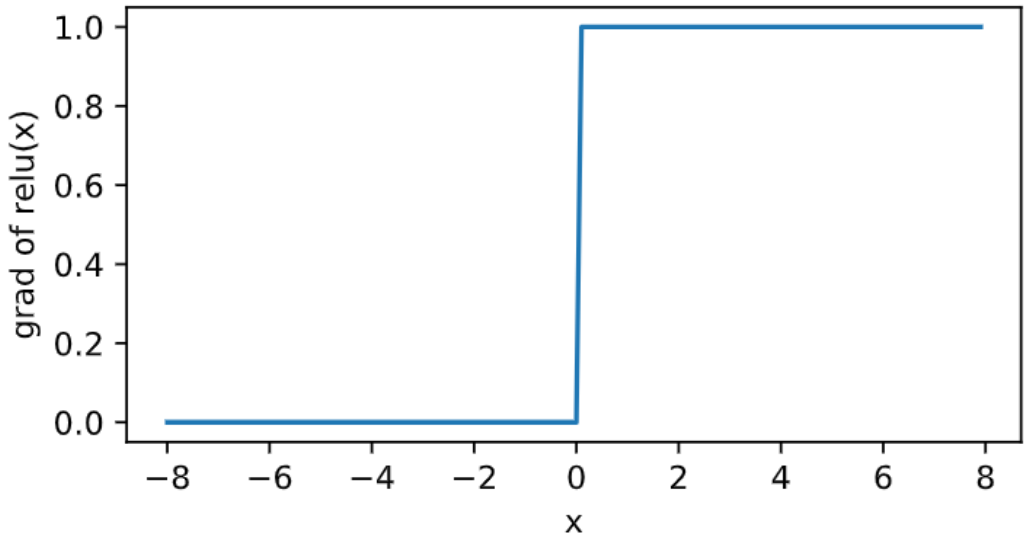
$$\boldsymbol{H} = \boldsymbol{X}\boldsymbol{W}_h + b_h,$$
$$\boldsymbol{O} = \boldsymbol{H}\boldsymbol{W}_o + b_o,$$

激活函数

引入非线性变换，对隐藏变量使用按元素运算的非线性函数进行变换，然后再作为下一个全连接层的输入。这个非线性函数被称为激活函数（activation function）

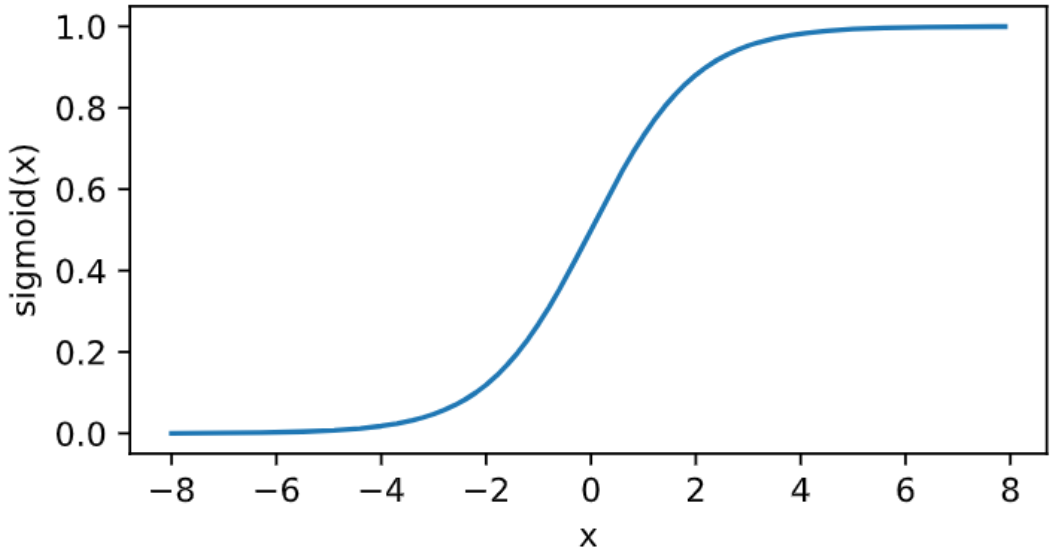
- **ReLU函数**

$$\text{ReLU}(x) = \max(x, 0).$$



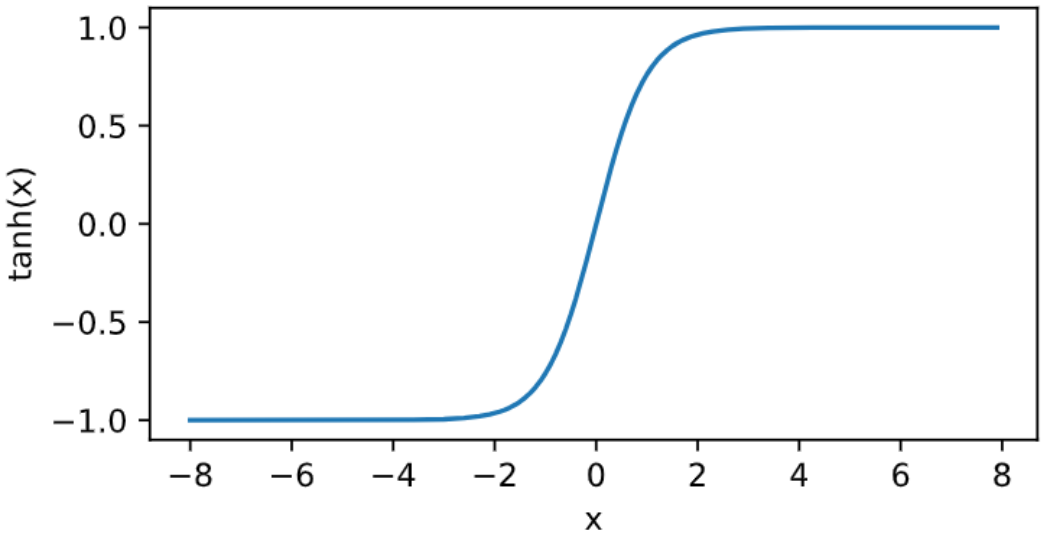
- **sigmoid函数**

$$\text{sigmoid}(x) = \frac{1}{1 + \exp(-x)}.$$



- **tanh函数**

$$\tanh(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}.$$



4.4.模型选择、欠拟合和过拟合

训练误差和泛化误差

- 训练误差指模型在训练数据集上表现出的误差，泛化误差指模型在任意一个测试数据样本上表现出的误差的期望，并常常通过测试数据集上的误差来近似。计算可以用平方损失函数和交叉熵损失函数。

训练集、验证集和测试集

从严格意义上讲，除非明确说明，否则本书中实验所使用的测试集应为验证集，实验报告的测试结果（如测试准确率）应为验证结果（如验证准确率）。

- **训练集**：训练集是机器学习模型用于训练和学习的数据集。通常情况下，训练集是原始数据集的一部分，用于训练模型的参数。模型通过训练集来学习数据的特征，并产生一个模型，以便在之后的预测中使用。

- **验证集**：验证集是用于评估模型性能的数据集。它通常是从原始数据集中划分出来的，用于在训练过程中调整模型的参数和超参数，以提高模型的性能。验证集的作用是帮助开发人员调整模型，避免模型过拟合或欠拟合。

- **测试集**：测试集是用于评估模型最终性能的数据集。它通常是从原始数据集中划分出来的，与训练集和验证集互不重叠。测试集的作用是评估模型在未见过的数据上的性能，并判断模型是否足够准确和鲁棒。

K折交叉验证

- 在K折交叉验证中，我们把原始训练数据集分割成K个不重合的子数据集，然后我们做K次模型训练和验证。每一次，我们使用一个子数据集验证模型，并使用其他K-1个子数据集来训练模型。在这K次训练和验证中，每次用来验证模型的子数据集都不同。最后，我们对这K次训练误差和验证误差分别求平均。

欠拟合和过拟合

- 欠拟合：模型无法得到较低的训练误差。

- 过拟合：模型的训练误差远小于它在测试数据集上的误差。

模型复杂度

- 给定训练数据集，如果模型的复杂度过低，很容易出现欠拟合；如果模型复杂度过高，很容易出现过拟合。

训练数据集大小

- 一般来说，如果训练数据集中样本数过少，过拟合更容易发生。此外，泛化误差不会随训练数据集里样本数量增加而增大。因此，在计算资源允许的范围内，我们通常希望训练数据集大一些。

4.5.权重衰减

应对过拟合的常见方法

方法

- 权重衰减等价于L_2范数正则化，正则化通过为模型损失函数添加惩罚项。L_2范数惩罚项指的是模型权重参数每个元素的平方和与一个正的常数的乘积。其中w_1,w_2是权重参数，b是偏差参数，样本i的输入为$x_1^{(i)}x_2^{(i)}$，标签为$y^{(i)}$，样本数为n。将权重参数用向量$\boldsymbol{w} = [w_1; w_2]$表示，则

$$\ell(w_1, w_2, b) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\left(x_1^{(i)} w_1 + x_2^{(i)} w_2 + b - y^{(i)}\right)^2$$

$$\ell(w_1, w_2, b) + \frac{\lambda}{2n}\|\boldsymbol{w}\|^2,$$

- 权重w_1和w_2迭代方式为(L_2范数正则化令权重w_1和w_2先自乘小于1的数，再减去不含惩罚项的梯度。)

$$w_1 \leftarrow \left(1-\frac{\eta\lambda}{|\mathcal{B}|}\right) w_1 - \frac{\eta}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} x_1^{(i)}\left(x_1^{(i)}w_1 + x_2^{(i)}w_2 + b - y^{(i)}\right),$$

$$w_2 \leftarrow \left(1-\frac{\eta\lambda}{|\mathcal{B}|}\right) w_2 - \frac{\eta}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} x_2^{(i)}\left(x_1^{(i)}w_1 + x_2^{(i)}w_2 + b - y^{(i)}\right).$$

4.6.暂退法（丢弃法）

本节中提到的丢弃法特指倒置丢弃法（inverted　dropout）。

方法

- 以多层感知机中的隐藏层为例。ϕ为激活函数，x_1…… x_4是输入，隐藏单元i的权重参数为w_{1i}…… w_{4i}，偏差参数为b_i。

$$h_i = \phi\left(x_1 w_{1i} + x_2 w_{2i} + x_3 w_{3i} + x_4 w_{4i} + b_i\right),$$

- 设丢弃概率为p（丢弃概率是丢弃法的超参数），那么有p的概率h_i会被清零，有1-p的概率h_i会除以1-p做拉伸。设随机变量ξ_i为0和1的概率分别为p和1-p。使用丢弃法时我们计算新的隐藏单元h'_i

$$h_i' = \frac{\xi_i}{1-p} h_i.$$

- 例如

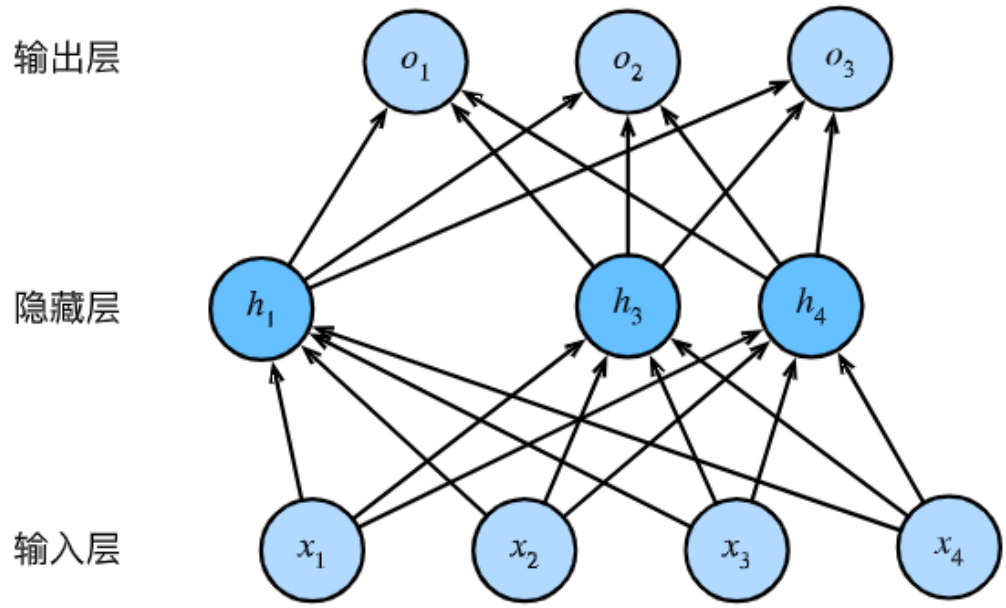


图 3.5: 隐藏层使用了丢弃法的多层感知机

4.8.数值稳定性和模型初始化

深度模型有关数值稳定性的典型问题是衰减（vanishing）和爆炸（explosion）。

衰减和爆炸

- 当神经网络的层数较多时，模型的数值稳定性容易变差。给定输入**X**，多层感知机的第l层的输出$\mathbf{H}^{(l)} = \mathbf{X}\mathbf{W}^{(1)}\mathbf{W}^{(2)} \ldots\ldots \mathbf{W}^{(l)}$。此时，如果层数l较大，$\mathbf{H}^{(l)}$的计算可能会出现衰减或爆炸。举个例子，假设输入和所有层的权重参数都是标量，如权重参数为0.2和5，多层感知机的第30层输出为输入**X**分别与$0.2^{30} \approx 1\times10^{-21}$（衰减）和$5^{30}\approx 9\times10^{20}$（爆炸）的乘积。

随机初始化模型参数

- 如果将每个隐藏单元的参数都初始化为相等的值，那么在正向传播时每个隐藏单元将根据相同的输入计算出相同的值，并传递至输出层。在反向传播中，每个隐藏单元的参数梯度值相等。因此，这些参数在使用基于梯度的优化算法迭代后值依然相等。之后的迭代也是如此。在这种情况下，无论隐藏单元有多少，隐藏层本质上只有1个隐藏单元在发挥作用。因此，正如在前面的实验中所做的那样，我们通常将神经网络的模型参数，特别是权重参数，进行随机初始化。

- **MXNet的默认随机初始化**：使用默认的随机初始化方法：权重参数每个元素随机采样于-0.07到0.07之间的均匀分布，偏差参数全部清零。

- **Xavier随机初始化**：假设某全连接层的输入个数为a，输出个数为b，Xavier随机初始化将使该层中权重参数的每个元素都随机采样于均匀分布。它的设计主要考虑到，模型参数初始化后，每层输出的方差不该受该层输入个数影响，且每层梯度的方差也不该受该层输出个数影响。

$$U\left(-\sqrt{\frac{6}{a+b}}, \sqrt{\frac{6}{a+b}}\right).$$

5.深度学习计算

5.1.层和块

块（block）可以描述单个层、由多个层组成的组件或整个模型本身。

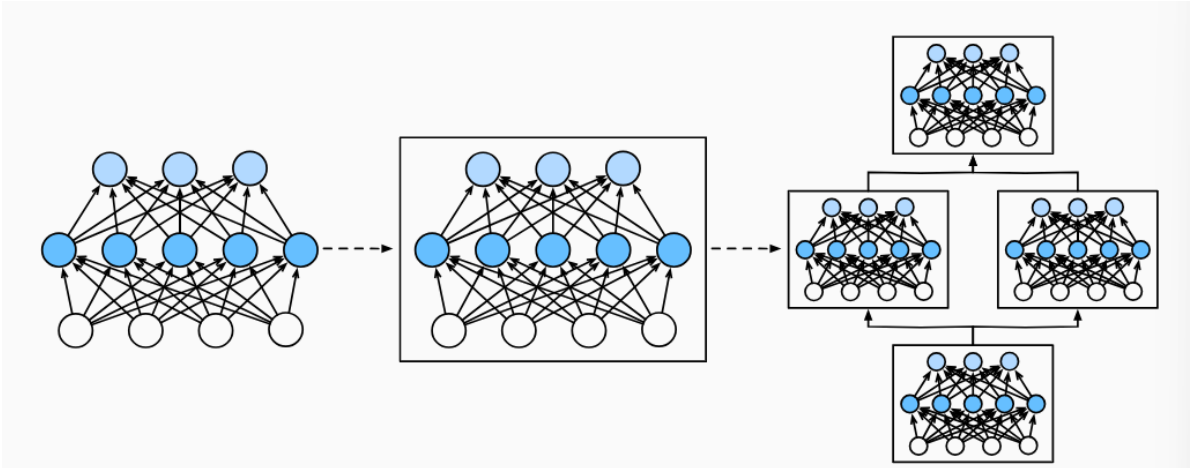


图5.1.1 多个层被组合成块，形成更大的模型

6.卷积神经网络

6.1.从全连接层到卷积

卷积神经网络（CNN）是机器学习利用自然图像中一些已知结构的创造性方法。

不变性

- **平移不变性（translation invariance）**：不管检测对象出现在图像中的哪个位置，神经网络的前面几层应该对相同的图像区域具有相似的反应，即为“平移不变性”。

- **局部性（locality）**：神经网络的前面几层应该只探索输入图像中的局部区域，而不过度在意图像中相隔较远区域的关系，这就是“局部性”原则。最终，可以聚合这些局部特征，以在整个图像级别进行预测。

- 下图是一个**卷积层**，**V**被称为卷积核或者滤波器，亦或简单地称之为该卷积层的权重，通常该权重是可学习的参数。

$$[\mathbf{H}]_{i,j} = u + \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} [\mathbf{V}]_{a,b} [\mathbf{X}]_{i+a,j+b}.$$

卷积

- 在数学中，两个函数之间的“卷积”被定义为以下所示（把一个函数“翻转”并移位**x**时，测量**f**和**g**之间的重叠）

$$(f * g)(\mathbf{x}) = \int f(\mathbf{z}) g(\mathbf{x} - \mathbf{z}) d\mathbf{z}.$$

- 对于离散对象

$$(f * g)(i) = \sum_a f(a) g(i - a).$$

- 对于二维张量的离散对象

$$(f * g)(i, j) = \sum_a \sum_b f(a, b) g(i - a, j - b).$$

6.2.图像卷积

互相关运算

- **互相关运算**（严格来说，卷积层是个错误的叫法，因为它所表达的运算其实是互相关运算）

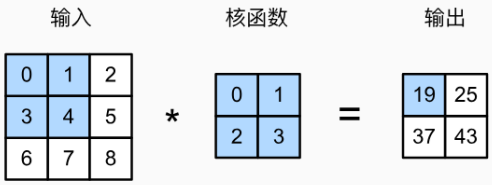


图6.2.1 二维互相关运算。阴影部分是第一个输出元素，以及用于计算输出的输入张量元素和核张量元素：$0 \times 0 + 1 \times 1 + 3 \times 2 + 4 \times 3 = 19$.

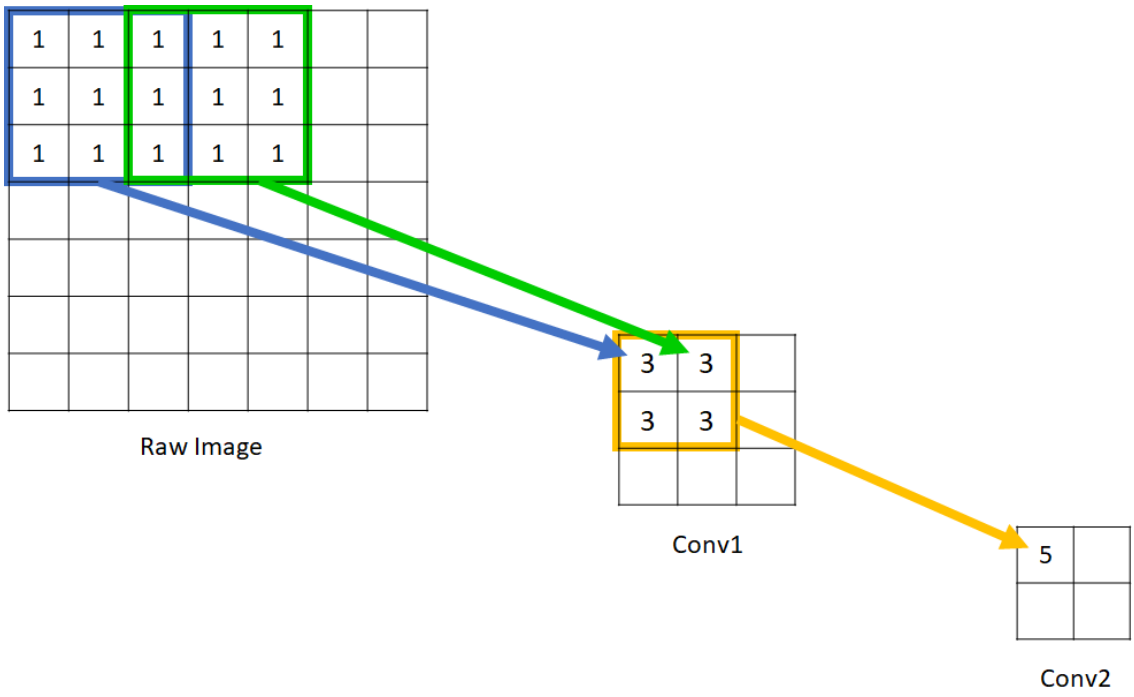
卷积层

- 卷积层对输入和卷积核权重进行互相关运算，并在添加标量偏置之后产生输出。 所以，卷积层中的两个被训练的参数是卷积核权重和标量偏置。 就像我们之前随机初始化全连接层一样，在训练基于卷积层的模型时，我们也随机初始化卷积核权重。

特征映射和感受野

- 卷积层有时被称为特征映射（feature map），因为它可以被视为一个输入映射到下一层的空间维度的转换器。

- **感受野**，例如，Conv1的感受野是3，Conv2的感受野是5，输入图像的每个单元的感受野被定义为1



6.3.填充和步幅

填充

- 在应用多层卷积时，我们常常丢失边缘像素。解决这个问题的简单方法即为填充（padding）：在输入图像的边界填充元素（通常填充元素是0）。

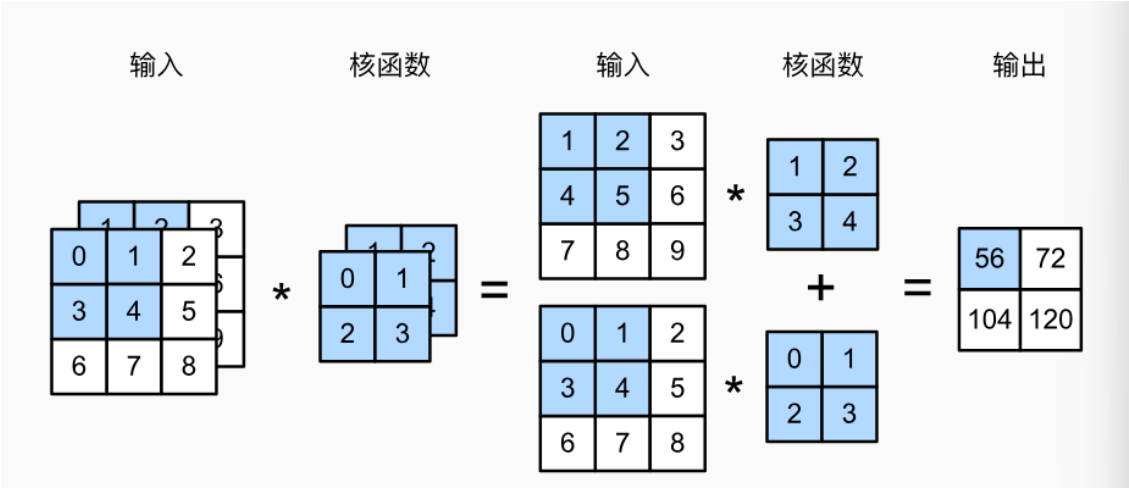
步幅

- 为了高效计算或是缩减采样次数，卷积窗口可以跳过中间位置，每次滑动多个元素。每次滑动元素的数量称为步幅。

6.4.多输入多输出通道

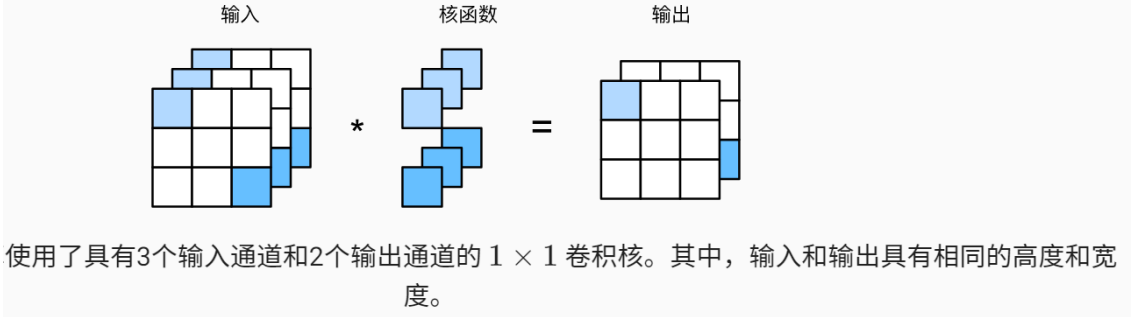
多输入通道

- 演示一个具有两个输入通道的二维互相关运算的示例。阴影部分是第一个输出元素以及用于计算这个输出的输入和核张量元素：(1×1+2×2+4×3+5×4)+(0×0+1×1+3×2+4×3)=56。



多输出通道

- 两个输出通道



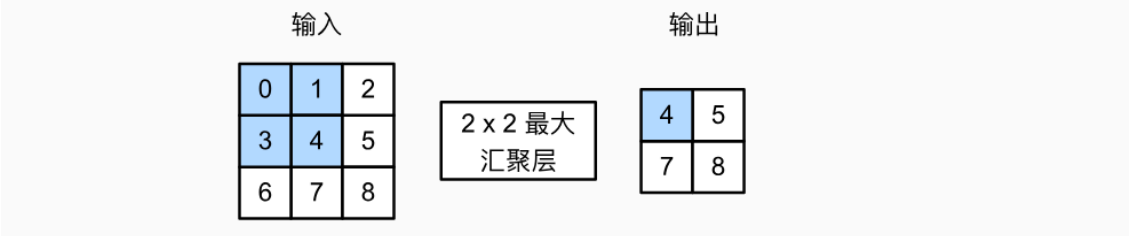
使用了具有3个输入通道和2个输出通道的 1×1 卷积核。其中，输入和输出具有相同的高度和宽度。

6.5.汇聚层

降低卷积层对位置的敏感性，同时降低对空间降采样表示的敏感性。

最大汇聚层和平均汇聚层

- 与卷积层类似，汇聚层运算符由一个固定形状的窗口组成，该窗口根据其步幅大小在输入的所有区域上滑动，为固定形状窗口（有时称为汇聚窗口）遍历的每个位置计算一个输出。，汇聚层不包含参数。举例



汇聚窗口形状为 2×2 的最大汇聚层。着色部分是第一个输出元素，以及用于计算这个输出的输入元素: $\max(0, 1, 3, 4) = 4$.

- 默认情况下，深度学习框架中的步幅与汇聚窗口的大小相同。 因此，如果我们使用形状为`(3, 3)`的汇聚窗口，那么默认情况下，我们得到的步幅形状为`(3, 3)`。

多个通道

- 在处理多通道输入数据时，汇聚层在每个输入通道上单独运算，而不是像卷积层一样在通道上对输入进行汇总。 这意味着汇聚层的输出通道数与输入通道数相同。

6.6.卷积神经网络

LeNet

- 总体来看，LeNet（LeNet-5）由两个部分组成：
  - 卷积编码器：由两个卷积层组成;
  - 全连接层密集块：由三个全连接层组成。

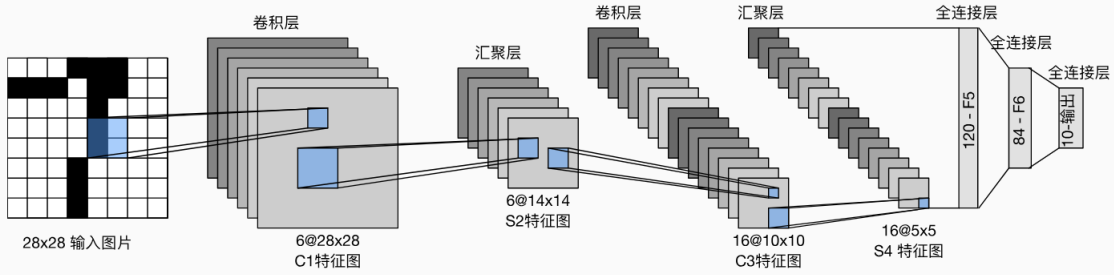


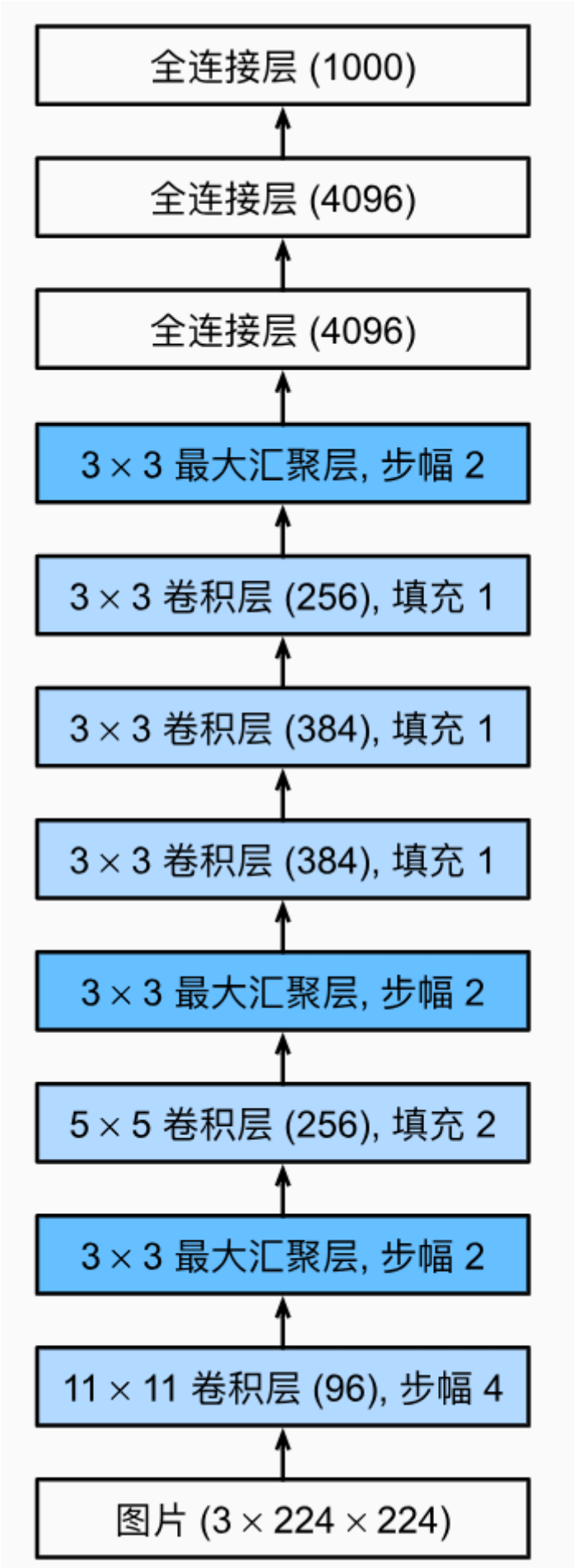
图6.6.1 LeNet中的数据流。输入是手写数字，输出为10种可能结果的概率。

7.现代卷积神经网络

7.1.深度卷积神经网络

AlexNet

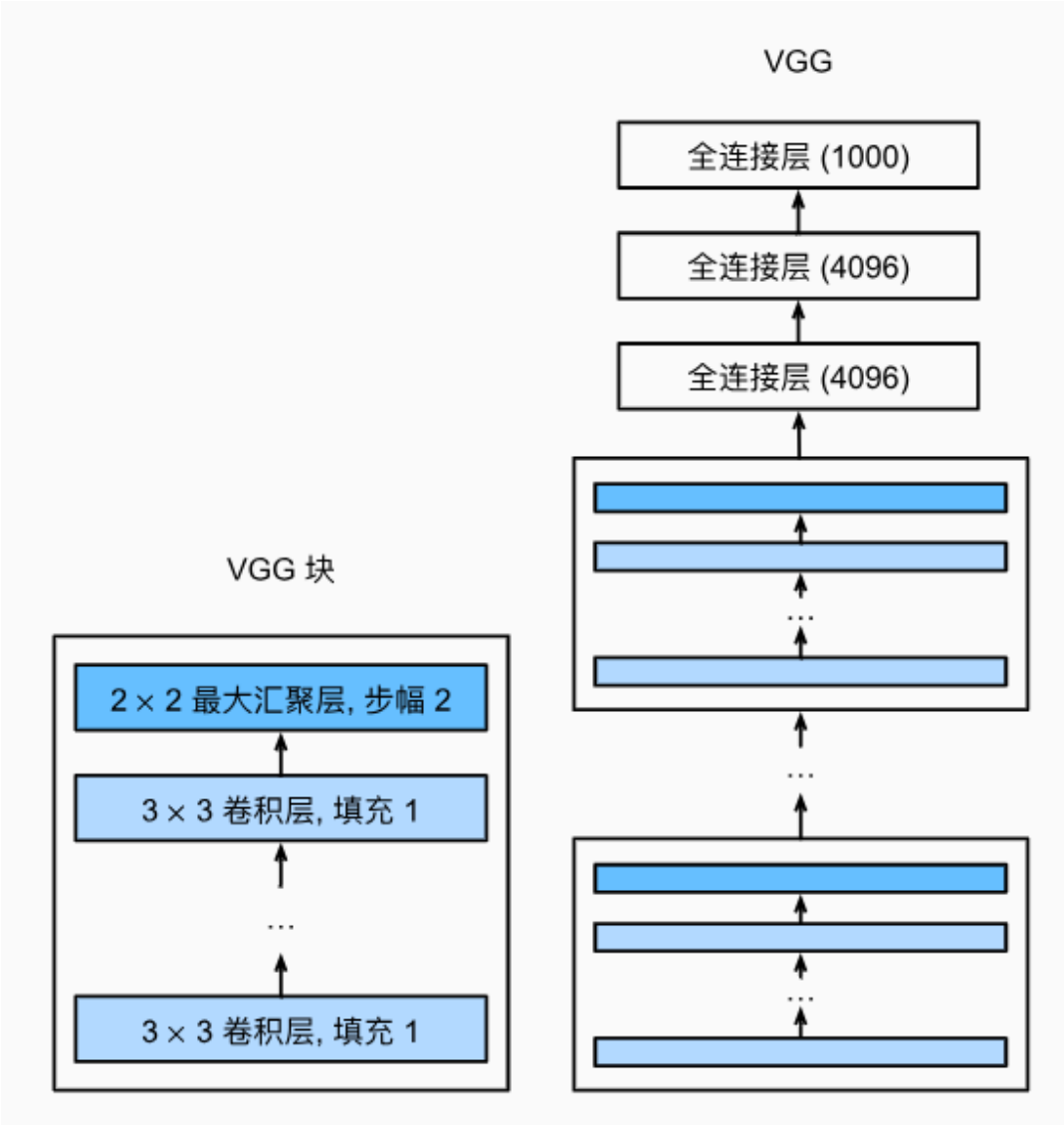
- AlexNet由八层组成：五个卷积层、两个全连接隐藏层和一个全连接输出层。



7.2.使用块的网络（VGG）

VGG块

- 经典卷积神经网络的基本组成部分是下面的这个序列：
    1. 带填充以保持分辨率的卷积层；
    2. 非线性激活函数，如ReLU；
    3. 汇聚层，如最大汇聚层。



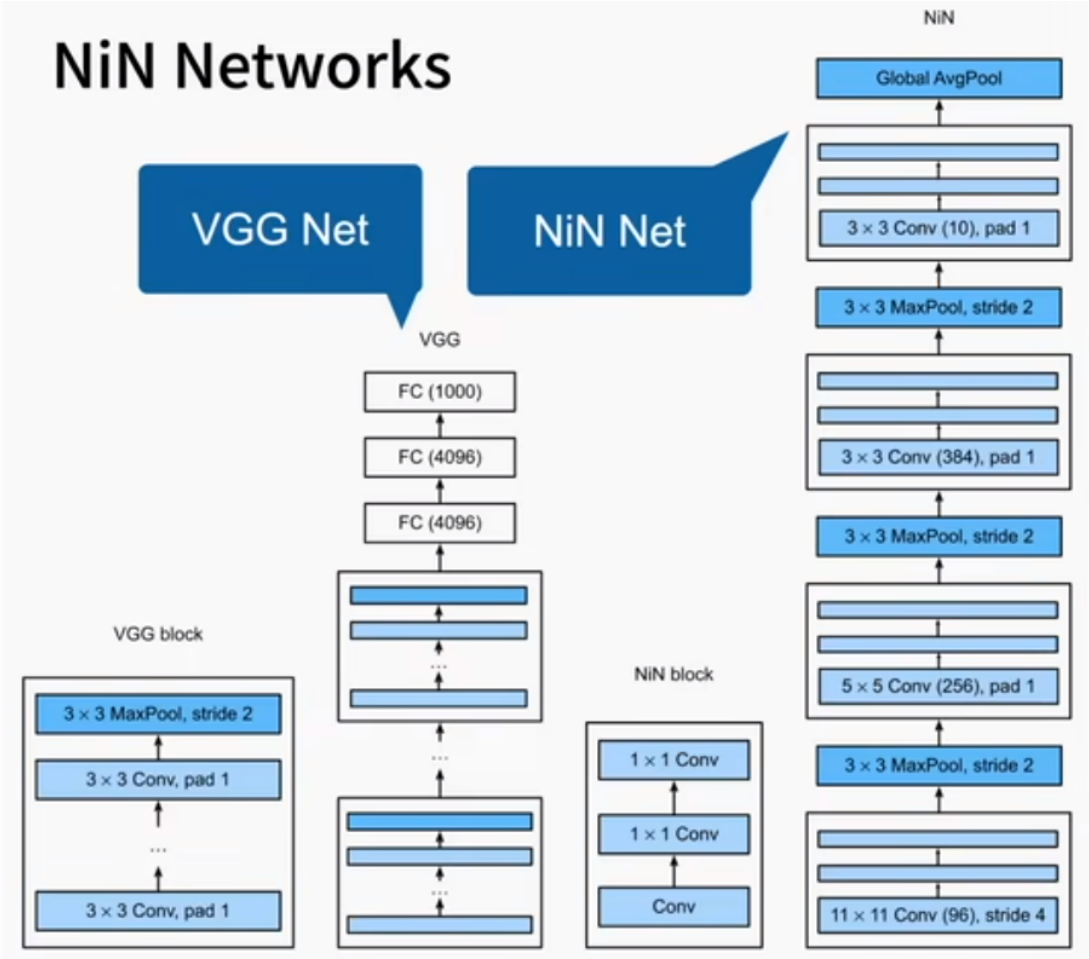
- LeNet (1995)
  - 2 卷积 + 池化层
  - 2 全连接层
- AlexNet
  - 更大更深
  - ReLu, Dropout, 数据增强
- VGG
  - 更大更深的 AlexNet（重复的 VGG 块）

7.3.NiN

NiN块

NiN架构

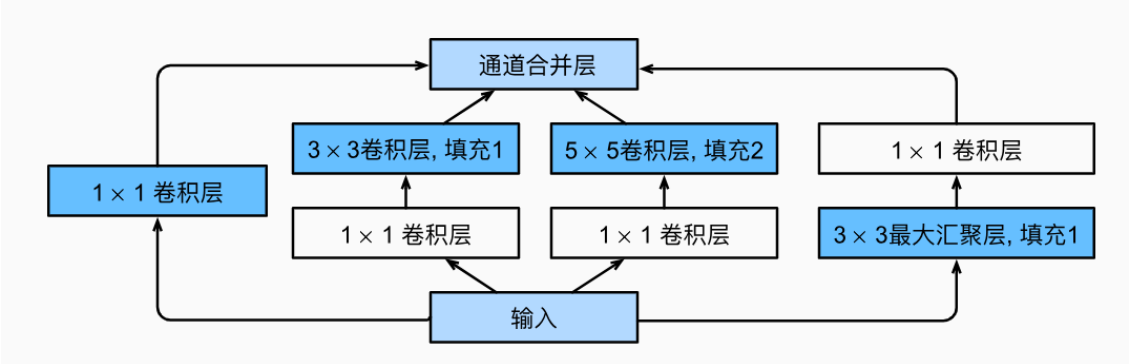
- 无全连接层
- 交替使用NiN块和步幅为2的最大池化层
  - 逐步减小高宽和增大通道数
- 最后使用全局平均池化层得到输出
  - 其输入通道数是类别数

- 

7.4.含并行连结的网（GoogLeNet）

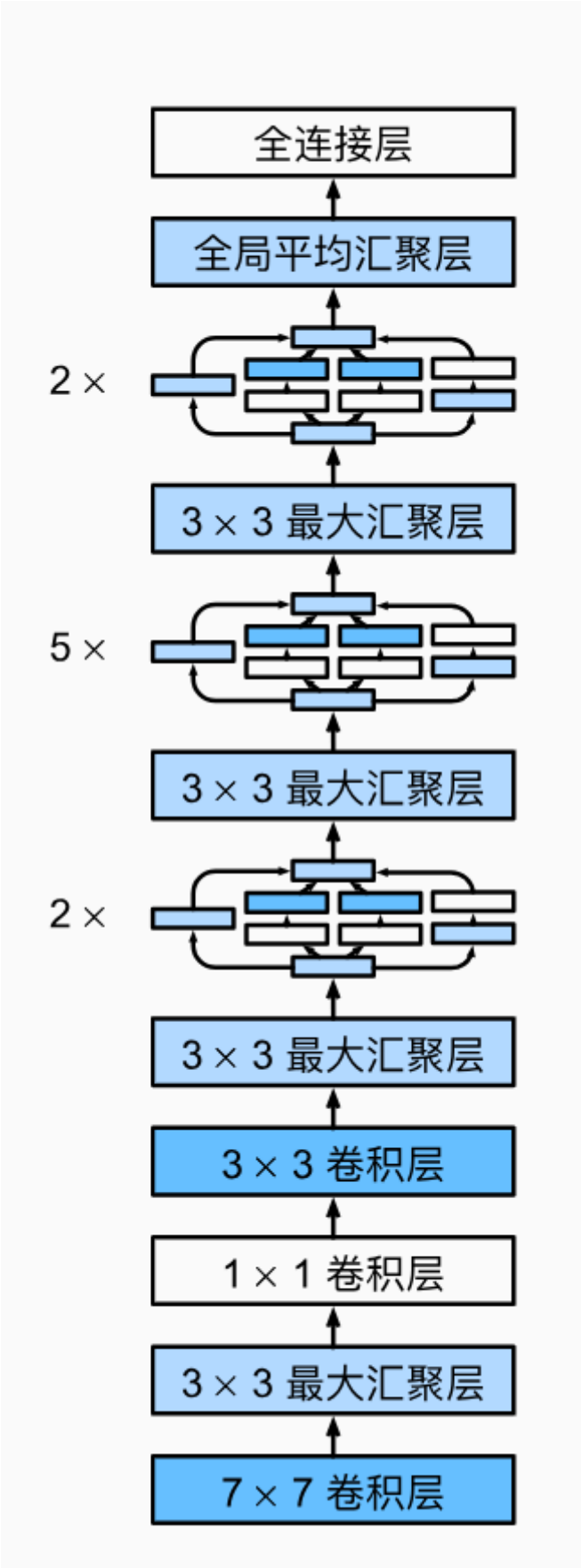
Inception块

- Inception块由四条并行路径组成



GoogLeNet模型

- GoogLeNet一共使用9个Inception块和全局平均汇聚层的堆叠来生成其估计值



7.6.残差网络（ResNet）

残差块

- 残差块里首先有2个有相同输出通道数的3×3卷积层。 每个卷积层后接一个批量规范化层和ReLU激活函数。 然后我们通过跨层数据通路，跳过这2个卷积运算，将输入直接加在最后的ReLU激活函数前。

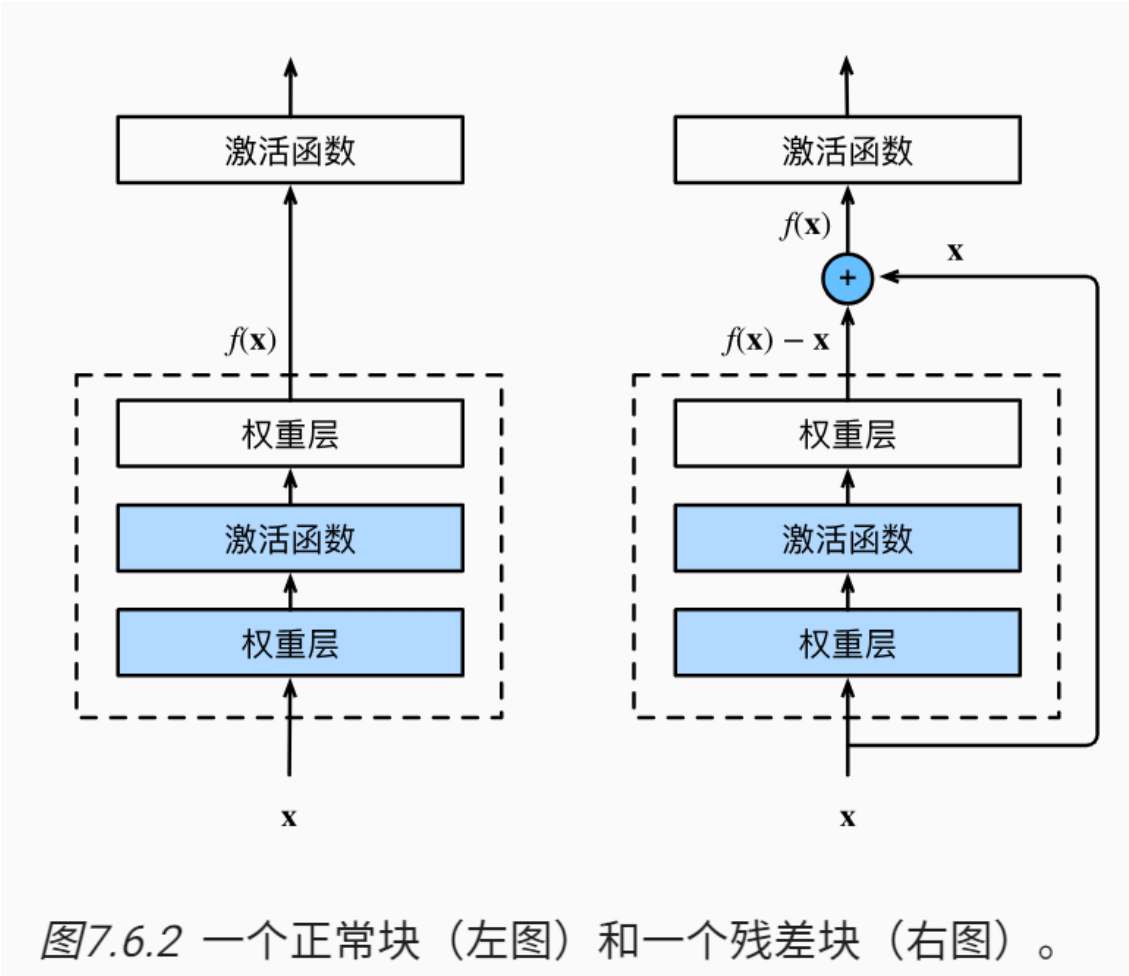


图7.6.2 一个正常块（左图）和一个残差块（右图）。