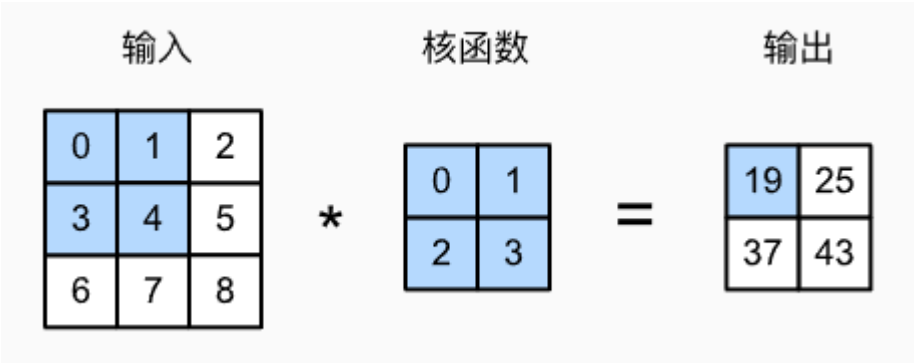
CNN卷积神经网络

卷积层

卷积运算

数学定义两个函数的卷积：$(f*g)(x)=\int f(z)g(x-z)dz$
当为离散对象时，积分变为求和：
$(f*g)(i)=\sum_a f(a)g(i-a)$
当为离散二维对象：$(f*g)(i,j)=\sum_a\sum_b f(a,b)g(i-a,i-b)$

互相关运算

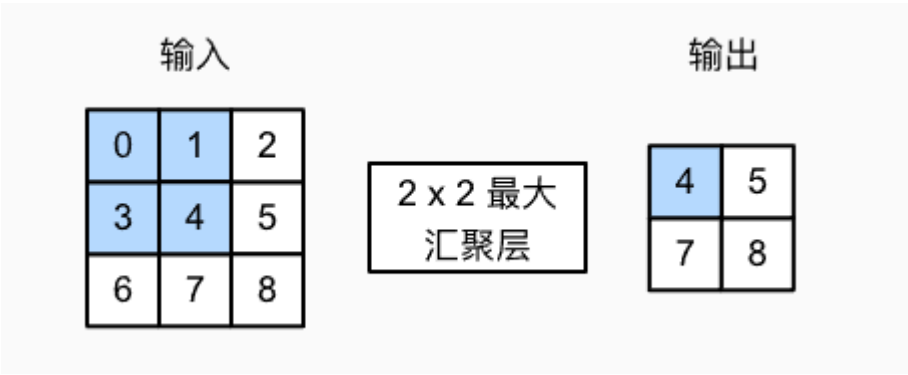


(卷积层运行示意图)
卷积窗口从输入张量的左上角开始，从左到右、从上到下滑动。
例如：19=0x0+1x1+3x2+4x3

卷积层的含义

实际上，卷积层内进行的并不是卷积运算，而是互相关运算。但是卷积核是从数据中学习不断更新的，所以无论是严格卷积还是互相关运算，输出是一样的。

汇聚层（池化层）

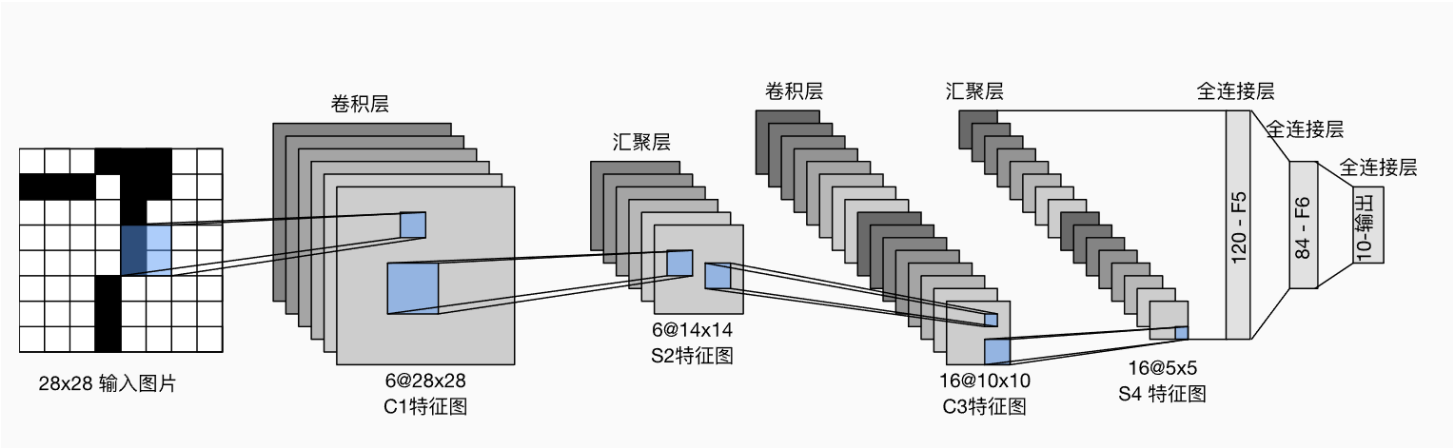


汇聚层不包含参数，从输入张量的左上角开始，从左往右、从右往左地滑动。计算窗口的最大值或平均值（取决于使用最大汇聚层还是平均汇聚层）。
汇聚层在精简数据，降低卷积层对位置的敏感性，同时降低对空间降采样表示的敏感性。

全连接层

全连接层起到“分类器”的作用，整合前面层中学习到的所有信息。但是其参数较多，需要更大的内存和计算资源。

卷积神经网络结构



此为LeNet卷积神经网络，属于早期卷积神经网络。可以看出基本构架为卷积层和汇聚层交替进行，全连接层作为结尾整合信息。

优化器SGD

欠拟合和过拟合

训练误差和验证误差都很严重，但它们之间仅有一点差距。这可能意味着模型过于简单（即表达能力不足），无法捕获试图学习的模式。这种现象被称为**欠拟合**。
将模型在训练数据上拟合的比在潜在分布中更接近的现象称为**过拟合**。训练误差明显低于验证误差时表明严重的过拟合。我们通常更关心验证误差，而不是训练误差和验证误差之间的差距。而过拟合的原因之一是算法模型过于复杂，过分考虑了当前样本结构。

正则化

正则化方法包括L1正则化和L2正则化，它们通过对损失函数加上一个约束（也可叫惩罚项），来减小解的范围，从而使学习后的参数估计更趋近于零。

权重衰减

权重衰减（weight decay）是最广泛使用的正则化的技术之一， 它通常也被称为L2正则化。
例如：
线性回归中的损失函数：L(**w**,b)=$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\left(\mathbf{w}^T\mathbf{x}^{(i)}+b-y^{(i)}\right)^2$
$\mathbf{x}^{(i)}$是样本i的特征，$\mathbf{y}^{(i)}$是样本i的标签，$(\mathbf{w},b)$是权重和偏置参数，为了惩罚权重向量的大小，需要以某种方式添加$\|\mathbf{w}\|^2$，通过正则化常数λ来平衡加入新的损失（常数2是为了方便求导）。
加入惩罚项后权重更新式：$\mathbf{w}\leftarrow(1-\eta\lambda)\mathbf{w}-\frac{\eta}{|\beta|}\sum_{i\in\beta}\left(\mathbf{w}^T\mathbf{x}^{(i)}+b-y^{(i)}\right)$
仅考虑惩罚项，在训练的每一步**w**的系数都是小于1的，也就是在衰减权重。并且λ是一个常数，可以据此来调节对**w**的约束力。

SGD相关参数

```
torch.optim.SGD(params, lr=<required parameter>, momentum=0, dampening=0, weight_decay=0, neste
```

1. params（必须参数）：这是一个包含了需要优化的参数（张量）的迭代器，例如模型的参数model.parameters()。

2. lr（必须参数）为学习率，它是一个正数，控制每次参数更新的步长。较小的学习率会导致收敛较慢，较大的学习率可能导致震荡或无法收敛。在sgd中，可以理解为：$p^{'} = p * momentum - lr * dp$其中p就是模型中的参数比如：权重(w), 偏置(b)。$p^{'}$为p的另一种形式，即用来替换上一次的p.
3. momentum（默认值为 0）: 动量（momentum）是一个用于加速 SGD 收敛的参数。它引入了上一步梯度的指数加权平均。通常设置在 0 到 1 之间。当 momentum 大于 0 时，算法在更新时会考虑之前的梯度，有助于加速收敛。
4. weight_decay（默认值为 0）: 权重衰减，也称为 L2 正则化项。它用于控制参数的幅度，以防止过拟合，也就是权重更新式中的λ。通常设置为一个小的正数，若weight_decay很大，则复杂的模型损失函数的值也就大。
5. *dampening（默认值为 0）: 阻尼项，用于减缓动量的速度。在某些情况下，为了防止动量项引起的震荡，可以设置一个小的 dampening值。nesterov：采用Nesterov加速梯度法。默认值为False。*

前向传播与反向传播

前向传播指的是：从输入层到输出层计算和储存神经网络中每层的结果。
反向传播指的是：从输出层到输入层计算和存储参数梯度的任何中间变量（偏导数）。
反向传播主要是为了利用其给出的梯度来更新模型参数。

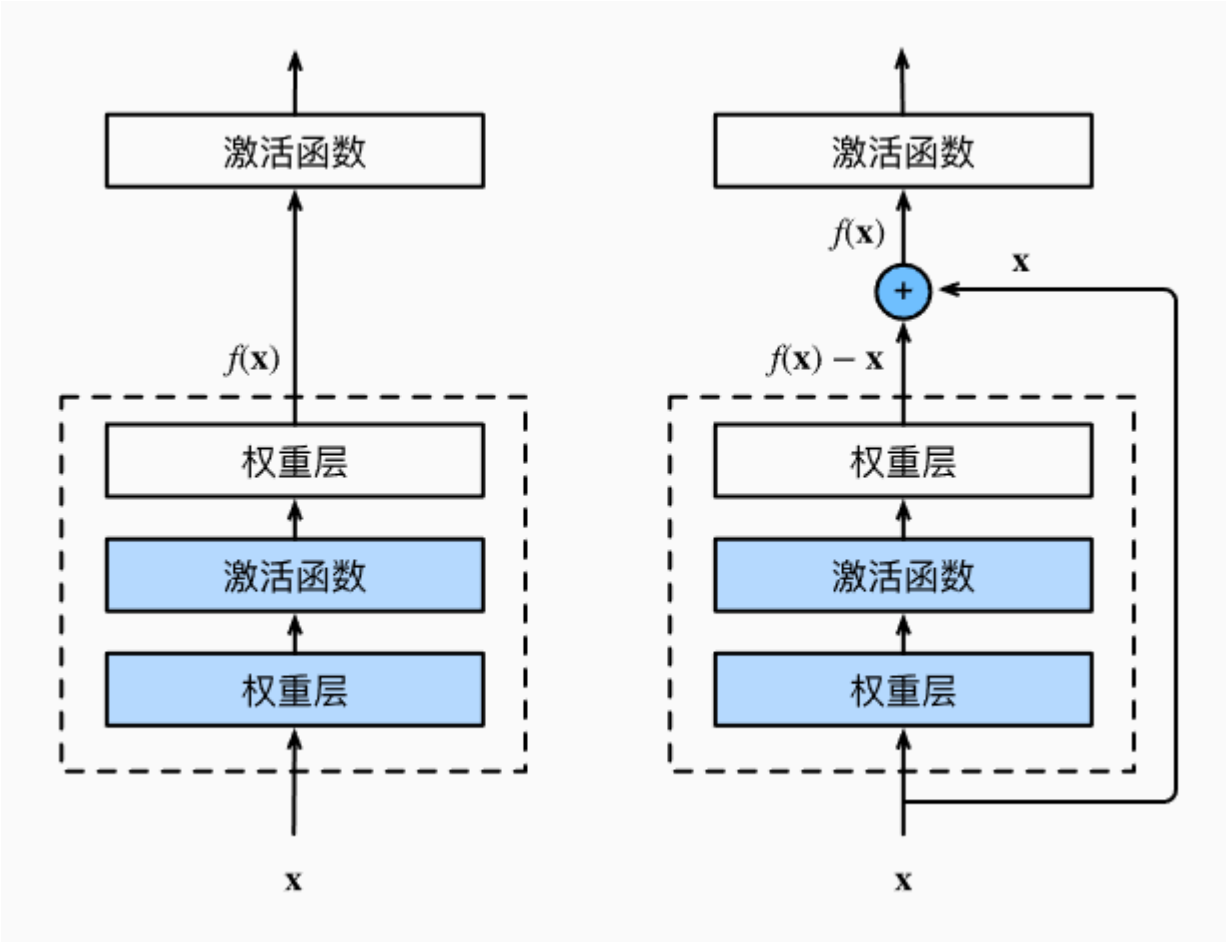
ResNet残差网络（ResNet-18）

激活函数

激活函数负责将神经元的输入映射到输出端，将非线性特性引入到网络中，使得神经网络能够逼近任意非线性函数。
常用的ReLU函数：其解析式为out = max(0, x)。当输入x<0时，输出为0；当x>0时，输出为x。

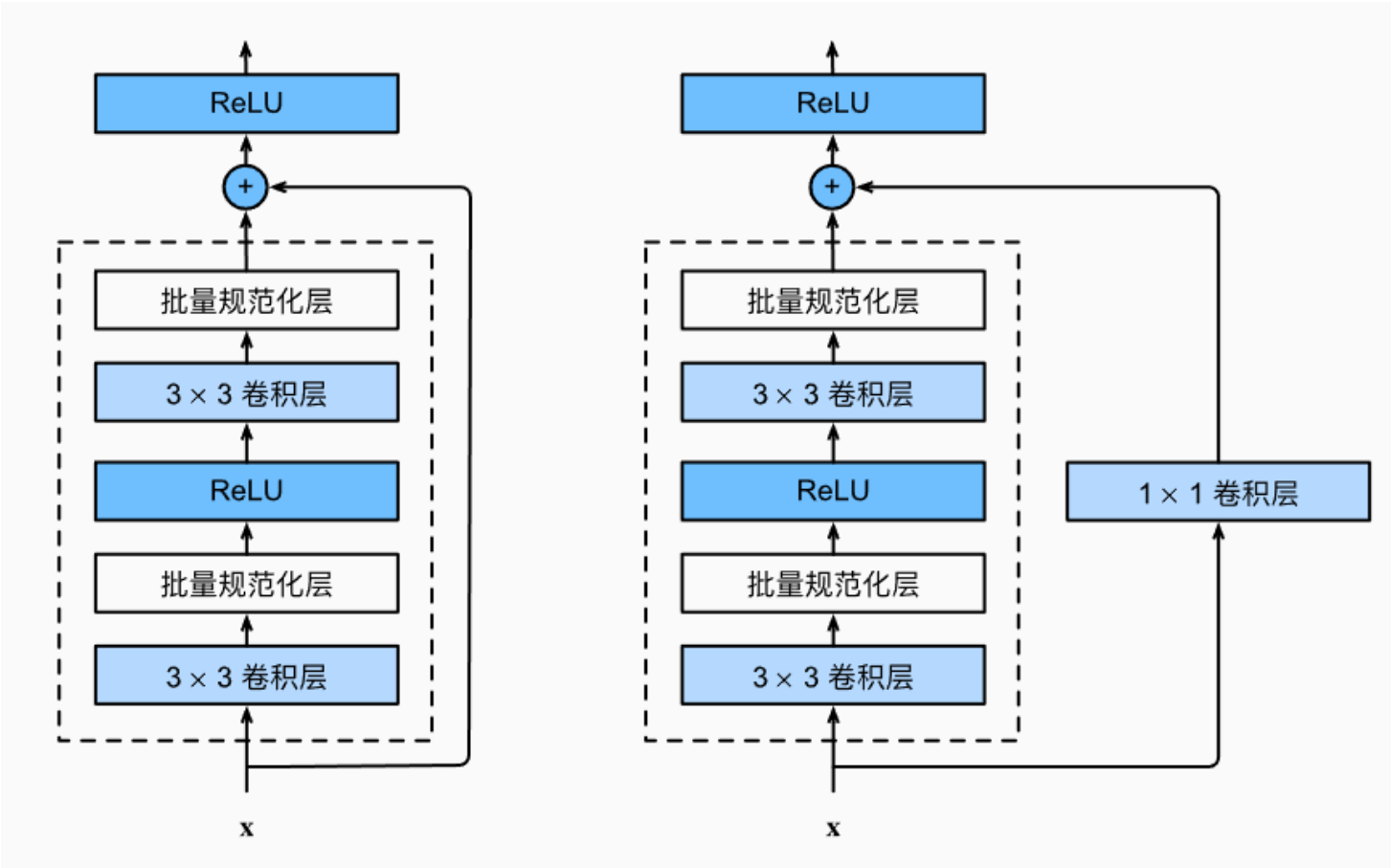
残差块

正常块与残差块



左侧是一个正常块，它需要模型去学习由x到f(x)完整的映射关系；右侧是一个残差块，同样是得到f(x)，但它不需要学习完整的映射关系，只需要学习输入到输出的微小变化，进而提高了优化效率。除此之外，残差块引入了恒等映射，也就是f(x)=(f(x)-x)+x，使梯度在传播过程中保持较大的值，避免了网络层数的增加导致梯度在反向传播过程中逐渐消失的问题

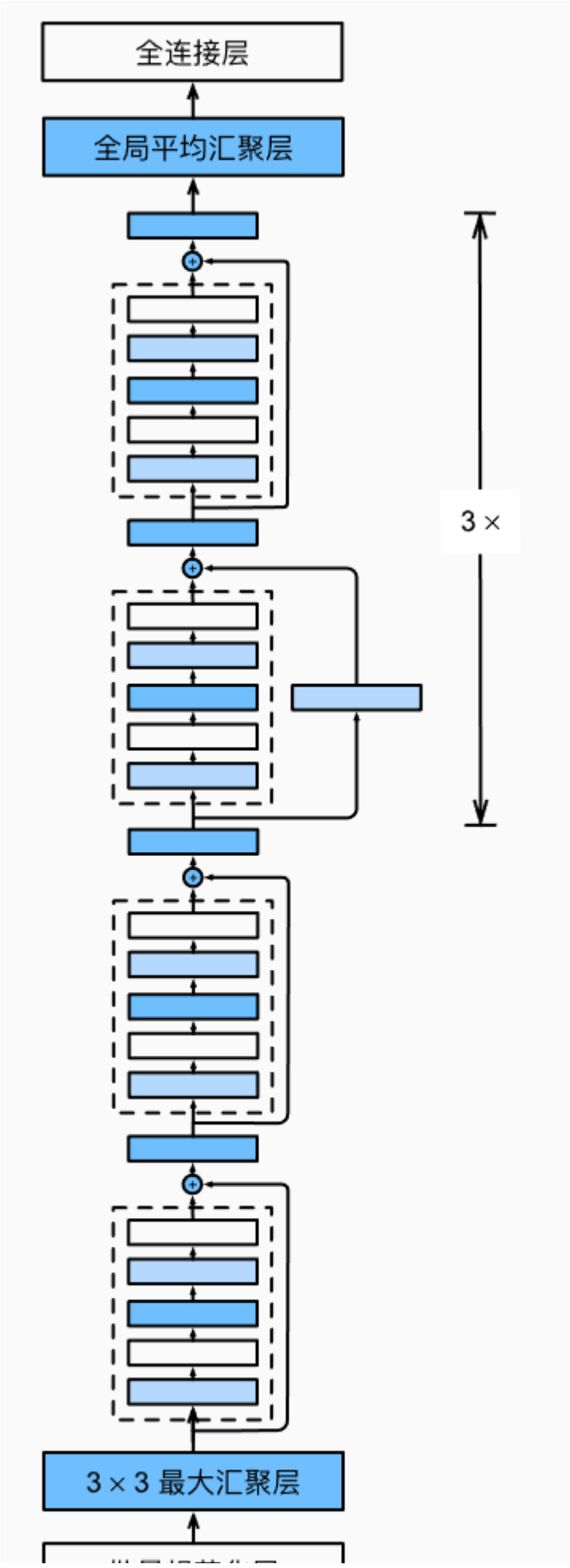
包含以及不包含1x1卷积层的残差块

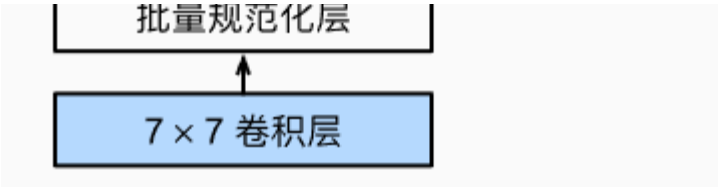


残差块里有2个有相同输出通道数的3x3卷积层。可以通过跨层数据通路，跳过这2个卷积运算，将输入直接加在最后的ReLU激活函数前。这样的设计要求2个卷积层的输出与输入形状一样，从而使它们可以相加。如果想改变通道数，就需要引入一个额外的1x1卷积层来将输入变换成需要的形状后再做相加运算。

总体结构

每个模块有4个卷积层（不包括恒等映射的1x1卷积层）。加上第一个7x7卷积层和最后一个全连接层，共有18层。


全连接层
全局平均汇聚层
3 ×
3 × 3 最大汇聚层



VGG使用块的网络

VGG网络可以分为两部分：第一部分主要由卷积层和汇聚层组成，第二部分由全连接层组成。