CNN卷积神经网络

CNN被用来进行图像分类问题，输入照片，输出的是分类概率构成的的向量
对于计算机来说图像是一个三维的Tensor`(length, width, channels)`
对于RGB三原色组成的图片来说，每一个图片由三维`(L,W,3)`张量构成，前两维表示的是图片大小，最后一维度表示的是对应的R,G,B的颜色强度
如果将图像的三维张量拉直得到`L*W*3`大小的输入向量，在全连接层乘以该层神经元个数`N`，每一个元素都对应一个weight，则第一层的weight就有`L*W*3*N`个

全连接网络的特点与局限

- **弹性最大**：每个神经元连接所有输入特征，可以观察整个图片。
- **参数庞大**：每个神经元对每一个输入像素都设有权重，参数量极大。
- **忽略空间结构**：没有利用图像的局部连续性和空间关系。

卷积神经网络的核心思想

1. 局部感受野（Receptive Field）

- 每个卷积神经元只连接输入图像的一个**局部区域**（感受野），而非整张图像。
- 利用图像的空间局部性：相邻像素间相关性强。
- 感受野尺寸通常较小（如3×3或5×5），通过多层堆叠可捕捉更高层次语义特征。
- 感受野间**通常部分重叠**（步长stride < 卷积核尺寸），防止边缘特征丢失。

- 每个感受野涵盖所有通道（例如RGB三通道），即感受的是一个 `高 × 宽 × 通道数` 的小立方体。

2. 参数共享（Weight Sharing）

- 一个卷积核（filter）在整个输入上滑动进行互相关运算，**使用同一组权重参数**。
- 该卷积核学习识别一种特定的局部模式（如边缘、纹理等），无论出现在哪个位置都能识别。
- 共享参数显著减少网络总参数量，相比全连接层更高效。
- 一个卷积层通常含多个卷积核，每个卷积核负责学习不同特征。

3. 特征平移不变性（Translation Invariance）

- 同一特征可能出现在图像不同位置。
- 卷积核滑动检测保证该特征无论在哪都能被捕捉。
- 不必为每个位置设独立神经元，节省参数。
- 赋予模型对空间位置变化的鲁棒性，即目标移动也能识别。

4. 卷积(Convolution)

超参数	作用说明
卷积核数量（filters）	决定每层输出的特征图数量，影响网络学习的特征维度（越多越强表达力）
卷积核大小（kernel size）	控制感受野范围，常用如 `3×3`、`5×5`，太大会损失细节，太小捕捉不了高级特征
步长（stride）	控制卷积核滑动的步幅，越大则输出越小，降低计算量，但可能损失细节
填充（padding）	是否在输入边缘补0（或其他值）来保持空间大小，`same` 或 `valid` 是常见设置
激活函数（activation）	如 ReLU、LeakyReLU、Sigmoid、Tanh，用于引入非线性

超参数	作用说明
使用偏置（bias）	控制每个卷积核是否带有一个可学习的偏置项

$$\text{输出} = \sigma\left(\sum_{c=1}^{C_{in}}\sum_{m=0}^{K_h-1}\sum_{n=0}^{K_w-1} W_{c,m,n}\cdot X_{c,i+m,j+n} + b\right)$$

- `X`：输入数据，形状为 (H_{in}, W_{in}, C_{in})。
- `W`：卷积核参数，大小 (K_h, K_w, C_{in})。
- `b`：偏置项。
- σ：激活函数（如 ReLU）。
  每一个卷积核的输出对应一个channel，每个输出通道可以识别特定模式

`1*1` 卷积层

不识别空间模式，只是融合通道，将多个channel融合为一个channel

5. 卷积层参数及输出关系

名称	说明
输入	形状为 (H_{in}, W_{in}, C_{in})
卷积核	大小 (K_h, K_w, C_{in})，覆盖输入所有通道
卷积核个数	(N_k)，决定输出通道数（特征图数量）
输出	形状 (H_{out}, W_{out}, N_k)
神经元个数	总数 = $(H_{out} \times W_{out} \times N_k)$

6. 池化（Pooling）

- 池化操作是在卷积层输出的每个特征图（矩阵）上，使用**滑动窗口**对局部区域进行汇聚处理，窗口大小常见为2×2或3×3。

- 在每个滑动窗口内进行**降维汇聚操作**，常见方式包括：
  - 最大池化（Max Pooling）：取窗口内最大值
  - 最小池化（Min Pooling）：取窗口内最小值
  - 平均池化（Average Pooling）：取窗口内平均值
- 每个滑动窗口被汇聚为一个单一数值，**从而有效减少特征图的空间尺寸（宽度和高度）**。
- 池化有助于降低计算复杂度和增强模型对输入平移的小幅变化的鲁棒性，但也可能导致部分细节信息的丢失。

7. 填充(Padding)

- Padding（填充）是指在**输入图像的边缘添加额外像素**（通常是0），以便在卷积操作中控制输出特征图的空间尺寸。
- 使用padding操作一般是为了防止边缘信息丢失，以及控制输出大小

8. CNN网络结构

- 通过反复的卷积（Convolution）和池化（Pooling）层，逐步提取输入数据的多层次特征。
- 特征提取完成后，通常将多通道的二维特征图**拉直（Flatten）**成一维向量。
- 最后，将该向量输入到**全连接层（Fully Connected Layer）**，完成分类或回归等任务。

批量归一化是一种流行并且有效的技术，可以持续加速深度神经网络的收敛速度

批量标准化

批量归一化的目的

1. 数据预处理的方式通常会对最终结果产生巨大的影响，标准化将参数的量级进行统一，可以很好地与优化器配合使用

2. 对于典型的多层感知机或者卷积神经网络，中间层的变量可能有更加广的变化范围，这些变量分布的偏移可能会阻碍网络的收敛
3. 更深层的网络非常复杂，容易过拟合，所以正则化就十分重要

批量标准化(BN)的操作步骤

批量规范化应用于单个可选层（也可以应用到所有层），其原理如下：在每次训练迭代中，我们首先规范化输入，即通过减去其均值并除以其标准差，其中两者均基于当前小批量处理。 接下来，我们应用比例系数和比例偏移
从形式上来说，用$\mathbf{x} \in \mathcal{B}$表示一个来自小批量$\mathcal{B}$的输入，批量规范化BN根据以下表达式转换$\mathbf{x}$：

$$\mathrm{BN}(\mathbf{x}) = \boldsymbol{\gamma} \frac{\mathbf{x} - \hat{\boldsymbol{\mu}}_\mathcal{B}}{\hat{\boldsymbol{\sigma}}_\mathcal{B}} + \boldsymbol{\beta}.$$

其中γ和β是拉伸参数和偏移参数，是需要和其他模型一起学习的参数
从形式上来看，我们计算出batch的$\hat{\boldsymbol{\mu}}_\mathcal{B}$和$\hat{\boldsymbol{\sigma}}_\mathcal{B}$，如下所示：

$$\hat{\boldsymbol{\mu}}_\mathcal{B} = \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x} \in \mathcal{B}} \mathbf{x}, \; \hat{\boldsymbol{\sigma}}_\mathcal{B}^2 \quad = \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x} \in \mathcal{B}} (\mathbf{x} - \hat{\boldsymbol{\mu}}_\mathcal{B})^2 + \epsilon.$$

请注意，我们在方差估计值中添加一个小的常量$\epsilon > 0$，以确保我们永远不会尝试除以零，即使在经验方差估计值可能消失的情况下也是如此。估计值$\hat{\boldsymbol{\mu}}_\mathcal{B}$和$\hat{\boldsymbol{\sigma}}_\mathcal{B}$通过使用平均值和方差的噪声（noise）估计来抵消缩放问题。 乍看起来，这种噪声是一个问题，而事实上它是有益的。

ResNet残差神经网络

1. 学习残差的角度

传统网络学习完整映射的难度

- 在传统的神经网络中，网络的目标是学习 **输入 x 到输出 y 的完整映射**。如果目标函数 y=f(x) 非常复杂，网络需要通过多层非线性变换来学习这个映射。
- 随着网络层数的增加，梯度在反向传播过程中逐渐消失或爆炸，导致网络难以训练。这个问题特别明显在 **深层网络** 中。

残差学习的简化

- ResNet 通过 **残差学习** 将网络的学习任务从学习 **整个映射** y=f(x)，转变为学习 **残差** F(x)=f(x)−x
  - 残差部分 F(x) 表示 **输入和输出之间的差异**，而输入 x 本身被直接加到输出中形成最终的结果。
  - 残差学习的目标不再是学习从 x 到 y 的直接映射，而是 **学习输入和输出之间的差异**，即 y=F(x)+x
- **为何学习残差更容易？**
  - 如果 F(x) 较小，意味着输出 y 与输入 x 很相近，那么网络就只需要学习较小的差异。
  - 如果 F(x) 较大，网络就会调整较大的差异部分，从而得到最终的输出。
  - 这种方法通过 **简化学习目标**（只学习差异），使得训练过程更加稳定，避免了深层网络中常见的 **梯度消失** 问题。

2. 从减少梯度消失的角度

ResNet 通过引入 **残差连接** 来减缓梯度消失问题。网络的每个残差块的输出可以表示为：

$$y = F(x) + x$$

其中：

- $F(x)$ 是网络的变换部分。
- x 是输入。
  通过这种设计，残差网络将梯度传播转化为以下形式：

对于一个残差块，损失函数 L 相对于输入 x 的梯度为：

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y} \cdot \left(\frac{\partial F(x)}{\partial x} + \frac{\partial x}{\partial x} \right)$$

由于残差连接$y = F(x) + x$ 中包含了直接传递的输入x，所以反向传播时，梯度可以通过两条路径进行传播：

1. 通过变换部分 $F(x)$ 的梯度传播，表示变换部分对梯度的影响。
2. 直接通过输入 x 传播，表示残差部分的梯度贡献。