

## Task\_2

### 一、残差连接：

#### 1、梯度范数：

梯度是损失函数关于模型参数的偏导数向量，而梯度范数是对这个梯度向量进行量化，用来衡量梯度的大小。在深度学习中，常用的梯度范数是 L2 范数。

把梯度向量看成一个  $n \times 1$  的矩阵时，L2 范数对应于矩阵范数中的 F 范数；

对于一个梯度向量  $\nabla\theta = (\theta_1, \theta_2, \dots, \theta_n)$ ，其 L2 范数的计算公式为：

$$\|\nabla\theta\|_2 = \sqrt{\sum_{i=1}^n \theta_i^2}$$

L1 范数也会被使用，其计算公式为：

$$\|\nabla\theta\|_1 = \sum_{i=1}^n |\theta_i|$$

#### 2、梯度消失/爆炸：

梯度消失：在深度神经网络中，反向传播算法用于计算梯度以更新参数。随着网络层数的增加，在反向传播过程中，梯度会逐层传递并不断相乘激活函数的导数。如果激活函数选择不当，那么经过多层传递后，梯度会越来越小，趋近于 0，导致靠近输入层的参数更新十分缓慢，甚至几乎不再更新，使得网络难以训练。

梯度爆炸：在反向传播过程中，由于初始权重过大或者网络结构等原因，使得梯度在逐层传递时不断增加，最终导致梯度值变得非常大，参数更新幅度过大，模型无法收敛，参数值可能变为无穷大或者 NaN。

#### 3、残差连接的计算公式与求导公式：

假设一个简单的残差块，输入为  $x$ ，经过一个或多个卷积层等操作得到的输出  $F(x)$ ，残差连接的输出  $y$  计算公式为：

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

求导公式为：

$$\frac{\partial y}{\partial x} = \frac{\partial F(x)}{\partial x} + 1$$

4、残差连接解决梯度消失/爆炸的原因：

(1) 从梯度传播角度：对于普通的神经网络层，假设从  $l$  层到  $l+1$  层，梯度传递公式类似于  $\frac{\partial \mathcal{L}}{\partial W^{(l)}} = \frac{\partial \mathcal{L}}{\partial z^{(l+1)}} \cdot \frac{\partial z^{(l+1)}}{\partial W^{(l)}}$  经过多层传递后，梯度容易因为连乘等原因出现消失或者爆炸。而对于残差连接，假设一个很深的网络，其中包含多个残差块，由于  $\frac{\partial y}{\partial x} = \frac{\partial F(x)}{\partial x} + 1$  的存在，即使  $\frac{\partial F(x)}{\partial x}$  趋近于 0，梯度仍然有一项为 1，这就保证了梯度能够相对稳定地反向传播，不会出现梯度消失的情况；

(2) 从模型训练角度：残差连接使得网络学习的是输入和输出之间的残差，相比直接学习输出，学习残差更容易，因为残差通常是一个相对较小的量，使得网络训练更加稳定，间接避免了梯度消失/爆炸带来的问题；

5、ResNet（残差网络）

(1) 核心作用：

主要功能是从图像中提取特征并进行分类。

(2) 关键组件解析：

残差块：网络的基本组成单元：

Block：基础残差块，由两个卷积层组成

Bottleneck：瓶颈残差块：由三个卷积层组成

ResNet 主网络：把残差块组合起来，形成完整的网络

初始层：先用一个大卷积（ $7 \times 7$ ）和池化层，对输入图像做初步特征提取和尺寸压缩；

多层残差块：通过 `_make_layer` 函数把多个残差块堆叠起来，每一层逐步增加特征通道数，减少图像尺寸，提取更复杂的特征。

分类层：最后用全局平均池化把特征压缩成向量，再通过全连接层输出分类结果；

## 二、卷积计算：

1、计算机采取存储，处理图像的数据结构：

在计算机系统中，图像的存储与处理依赖张量这一核心的数据结构。具体形式为：

多维数组表示：图像通常存储为三维张量： $R^{H \times W \times C}$  其中：

H 为图像高度，W 为图像宽度，C 为通道数；

且为了适配硬件（GPU 显存），张量被组织为连续内存块，并通过批处理维度，支持并行计算

2、卷积操作定义与图像处理流程：

卷积操作是利用卷积核对图像进行滑动加权求和的线性变换；

采取原则：平移不变性和局部性：

原则 #1 - 平移不变性	原则 #2 - 局部性
<ul style="list-style-type: none"><li>• x 的平移导致 h 的平移 <math>h_{i,j} = \sum_{a,b} v_{i,j,a,b} x_{i+a,j+b}</math></li><li>• v 不应该依赖于 (i, j)</li><li>• 解决方案： <math>v_{i,j,a,b} = v_{a,b}</math></li></ul> $h_{i,j} = \sum_{a,b} v_{a,b} x_{i+a,j+b}$ <p>这就是 2 维卷积 交叉相关</p>	<ul style="list-style-type: none"><li>• 当评估 <math>h_{i,j}</math> 时，我们不应该用远离 <math>x_{i,j}</math> 的参数</li><li>• 解决方案：当 <math> a ,  b  &gt; \Delta</math> 时，使得 <math>v_{a,b} = 0</math></li></ul> $h_{i,j} = \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} v_{a,b} x_{i+a,j+b}$

对应到矩阵变换即：

## 二维卷积层

$$\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix} * \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix} = \begin{bmatrix} 19 & 25 \\ 37 & 43 \end{bmatrix}$$

- 输入  $\mathbf{X} : n_h \times n_w$
- 核  $\mathbf{W} : k_h \times k_w$
- 偏差  $b \in \mathbb{R}$
- 输出  $\mathbf{Y} : (n_h - k_h + 1) \times (n_w - k_w + 1)$

$$\mathbf{Y} = \mathbf{X} \star \mathbf{W} + b$$

- $\mathbf{W}$  和  $b$  是可学习的参数

核心价值：卷积操作通过权值共享与局部连接，高效提取图像的局部特征

3、卷积操作的关键超参数：

(1) 核大小 (Kernel Size, k): 卷积核的空间维度, 影响感受野大小;

(2) 步长 (Stride): 卷积核滑动的步幅, 控制输出特征图的尺寸;

## 步幅

• 步幅是指行/列的滑动步长

• 例: 高度3 宽度2 的步幅

Input	Kernel	Output
$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 3 & 4 & 5 & 0 \\ 0 & 6 & 7 & 8 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}$	$= \begin{bmatrix} 0 & 8 \\ 6 & 8 \end{bmatrix}$

$0 \times 0 + 0 \times 1 + 1 \times 2 + 2 \times 3 = 8$   
 $0 \times 0 + 6 \times 1 + 0 \times 2 + 0 \times 3 = 6$

## 步幅

• 给定高度  $s_h$  和宽度  $s_w$  的步幅, 输出形状是

$$\lfloor (n_h - k_h + p_h + s_h) / s_h \rfloor \times \lfloor (n_w - k_w + p_w + s_w) / s_w \rfloor$$

• 如果  $p_h = k_h - 1$ ,  $p_w = k_w - 1$

$$\lfloor (n_h + s_h - 1) / s_h \rfloor \times \lfloor (n_w + s_w - 1) / s_w \rfloor$$

• 如果输入高度和宽度可以被步幅整除

$$(n_h / s_h) \times (n_w / s_w)$$

(3) 填充 (Padding): 图像边缘补充像素数, 用于保存输出尺寸或增强边界特征;

## 填充

• 填充  $p_h$  行和  $p_w$  列, 输出形状为

$$(n_h - k_h + p_h + 1) \times (n_w - k_w + p_w + 1)$$

• 通常取  $p_h = k_h - 1$ ,  $p_w = k_w - 1$

• 当  $k_h$  为奇数: 在上下两侧填充  $p_h/2$

• 当  $k_h$  为偶数: 在上侧填充  $\lceil p_h/2 \rceil$ , 在下侧填充  $\lfloor p_h/2 \rfloor$

(4) 输出通道数 (Output Channels): 卷积核的数量, 决定输出特征图的通道维度, 与模型容量正相关;

- 输入  $\mathbf{X} : c_i \times n_h \times n_w$
- 核  $\mathbf{W} : c_o \times c_i \times k_h \times k_w$
- 输出  $\mathbf{Y} : c_o \times m_h \times m_w$

$$\mathbf{Y}_{i,:,:} = \mathbf{X} \star \mathbf{W}_{i,:,:,:} \quad \text{for } i = 1, \dots, c_o$$

1×1的卷积仅涉及通道融合；

4、卷积层参数数量变化：

卷积层参数数量由公式  $P = k \times k \times C \times C'$   $k$  为核大小， $C$  为输入通道数， $C'$  为输出通道数，与全连接层相比 ( $H \times W \times C \times C'$ )

通常减少参数：因为  $k \ll H, W$  因而卷积层通过权值共享大幅减少参数；

特殊情况下，若核大小等于图像尺寸，则卷积层退化为全连接层；

5、卷积神经网络对图像处理有效原因：

(1) 局部连接与权值共享：利用图像的空间局部相关性（相邻像素关联紧密），减少参数数量的同时，强制模型学习平移不变的局部特征。

(2) 层级特征提取：浅层卷积层提取低阶特征，深层卷积层组合低阶特征为高阶语义，适配图像的分层结构；

(3) 感受野的动态拓展：随着网络加深，感受野逐渐增大，使模型能捕捉全局上下文信息；

6、卷积操作损失的改进策略：

卷积操作可能因下采样，大步长卷积导致小物体特征丢失，改进方法有：

空洞卷积：通过在卷积核中插入空洞，在不增加参数的前提下扩大感受野，保留小物体特征；

多尺度特征融合：融合不同层级的特征图，将深层语义信息与浅层细节信息结合，增强小物体检测能力；

注意力机制：引入通道或空间注意力模块（CBAM），动态加权特征图，突出小物体区域的特征；

### 三、Transformer

1、用 python 实现的注意力机制在 dot\_attention.py 里；

2、位置编码：

(1) 为什么 Transformer 本身没有任何位置意识？

Transformer 的核心是自注意力机制 (Self\_Attention)。自注意力通过计算序列中任意两个 token 的关联建模依赖，但这一过程不天然包含位置信息，从数学形式上看，自注意力核心计算为：

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{dk}})V$$

其中  $Q, K, V$  表示查询，键，值矩阵，仅由 token 嵌入 (Embedding) 线性变换得到，未显式融入位置相关的区分信息。因此若不额外设计“位置编码”，Transformer 无法区分同一 token 在序列中不同位置的语义差异

(2) 绝对，相对，可学习位置编码的定义与优劣

**绝对位置编码：**

定义：为每个位置  $i$  分配固定，独立的编码向量  $PE(i)$ ，与 token 嵌入直接相加后输入模型。Attention Is All You Need 中提到了正弦余弦编码

$$PE(i, 2j) = \sin\left(\frac{i}{10000^{2j/d_{\text{model}}}}\right), \quad PE(i, 2j + 1) = \cos\left(\frac{i}{10000^{2j/d_{\text{model}}}}\right)$$

其中  $d_{\text{model}}$  是模型维度， $j$  是维度索引。

优势：数学性质简洁，可通过三角函数的周期性天然支持长距离位置的泛化；

显示区分了 token 的绝对位置，让模型初步感知“顺序”；

劣势：仅刻画绝对位置，相对位置关系的建模能力弱，且对超长序列的外推能力有限；

**相对位置编码：**

定义：编码的核心是刻画 token 间的相对距离，Self-Attention with Relative Position Representations 中提出，在自注意力计算中引入相对位置偏置，让注意力权重同时受 token 内容和相对距离的影响。

ALiBI: Train Short , Test Long : Attention With Linear Biases Enables Input Length

Extrapolation 则通过线性相对偏置简化运算，直接为不同的 k 预定义偏置值，融入注意力分数。

优势：更贴合自然语言的语义逻辑；

对长序列外推更友好；

劣势：设计和实现更复杂，部分方法可能增大计算开销；

**可学习位置编码：**

**定义：**直接初始化一个可训练的参数矩阵  $PE \in \mathbb{R}^{L_{\max} \times d_{\text{model}}}$ ，其中  $L_{\max}$  是最大序列长度，模型通过训练自动学习每个位置的最优编码（Devlin et al., 2018 中 BERT 即采用类似思路）。

优势：灵活度高，无需手动设计编码规则，对特定数据集和任务，可能学到更贴合的位置关联；

劣势：泛化性受限，训练时的最大长度，会限制模型对更长序列的适配能力；

（3）LLM 常用的位置编码：

**正弦余弦编码：**经典 Transformer 标配，仍被部分基础模型采用；

**可学习位置编码：**BERT 模型及其衍生模型的首选，通过训练适配下游任务；

**相对位置编码变体：**

**ALiBi：**因长序列外推能力（“Train Short, Test Long”），被 Mistral 等模型借鉴，简化计算且适配超长输入；

**\*\* Rotary Position Embedding (RoPE) \*\*** (Su et al., 2021)：通过旋转矩阵编码相对位置，在 LLaMA、GPT - NeoX 等模型中广泛应用，兼顾绝对与相对位置建模，且支持高效外推。

3、层归一化：

**定义：**Layer Normalization 是对神经网络某一层的单个样本，在特征维度上做归一化

$$\text{LayerNorm}(x) = \gamma \odot \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

其中  $\mu$  是特征维度的均值， $\sigma^2$  是方差， $\gamma$ （缩放）、 $\beta$ （偏移）是可学习参数， $\epsilon$  是避免除零的小量。

**必要性：**

（1）解决内部协变量偏移，神经网络训练中，隐层输入的分布会随着参数更新而持续变化，导致训练不稳定，收敛慢。LayerNorm 通过归一化，固定隐层输入的均值和方差，加深训练；

（2）提升模型的鲁棒性，减少参数更新对隐层分布的剧烈影响

(3) 对 Transformer 尤为关键，自注意力输出的维度可能因序列长度而变化，按特征维度归一化后，适配动态序列输入；

### LayerNorm 和 BatchNorm 的区别：

(1) 从归一化维度上说：LayerNorm 是在**单个样本的特征维度**上归一化，BatchNorm 是在一批样本的**样本维度**上归一化

(2) 适用场景上：LayerNorm 适用于序列建模，**小批量或者动态序列**；BatchNorm 适用于计算机视觉，**固定维度且大批量**的场景

(3) 模型假设上：LayerNorm 需要假设同一层内**特征间存在关联**，需要独立归一化；BatchNorm 需要假设同一 batch 内，样本同分布；

### 4、Mask

Causal Mask 是在自回归任务中，为保证“因果性”而引入的**掩码**，它强制模型在预测第  $k$  个 token 时，仅仅关注前  $k-1$  个 token 的内容；

数学形式：在注意力计算时，对注意力分数矩阵  $QK^T$  施加一个下三角形矩阵掩码（下三角为 0，上三角全为  $-\infty$ ，经过 Softmax 后，上三角的权重被“屏蔽”。

作用：（核心保证因果推理）：

让模型学习自回归生成逻辑，避免模型利用“未来信息”作弊，确保生成任务的合理性；

### 5、束搜索（Beam Search）

为什么 LLM 输出序列是一个搜索问题：

LLM 生成文本时，本质是逐 token 的概率分布采样，给定前缀  $x_1, x_2 \dots x_t$ ，模型预测下一个 token  $x_{t+1}$  的概率分布  $P(x_{t+1} | x_1, x_2, \dots, x_t)$ ，由于每个位置的选择会影响后续生成，生成完整路径需要从指数级可能的路径中，选一个合理的（**概率较高且语义连贯的**）

因此生成序列可抽象为在概率图中搜索一条路径，每个结点是 token 的位置，边是 token 转移的概率，目标是找一条概率积（和）最大，语义合理的路径；

束搜索的价值：

通过维护多条候选，探索更多潜在的高概率路径，在生成质量与计算开销间做权衡，对长文本生成，束搜索能显著优于贪心搜索

### 6、Transformer 模型规模与梯度爆炸缓解：

(1) Transformer 为什么可以做到“这么大”：

并行性优势：自注意力虽需全局关联，但计算可并行化。通过矩阵运算优化（如多头



注意力的并行实现)，训练大规模模型时候能高效利用 GPU 集群的算力。

模块化设计：Transformer 的“encoder-decoder”架构，层归一化，残差连接等模块，可灵活堆叠扩展，且各模块稳定性支撑深度模型训练；

注意力机制的长程建模：自注意力可直接建模长距离依赖，让大规模模型能有效处理长文本，适配复杂任务需求；

(2) 哪些设计缓解的梯度爆炸：

残差连接：通过  $LayerOutput = LayerNorm(x + layer(x))$ ，短路连接使得梯度可直接通过残差路径反向传播，避免因**多层堆叠**导致的梯度消失/爆炸；

层归一化：固定隐层输入的均值和方差，减少**参数更新**对梯度的剧烈影响

注意力机制的梯度特性：自注意力的梯度计算基于权重的 softmax 分布，其**梯度传播相对平滑**；

#### 四、Diffusion 模型：

1、正态分布：

**正态分布**也称作**高斯分布**，是一种在自然界和社会现象中广泛存在的**连续概率分布**；在数学上，若一个随机变量  $X$  服从位置参数为  $\mu$ ，尺度参数为  $\sigma$  的正态分布，记作  $X \sim N(\mu, \sigma^2)$  其概率密度函数为

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

其中  $\mu$  是均值，决定了分布的中心位置， $\sigma$  是标准差，决定了分布的离散程度；正态分布图像呈现钟型。约 68% 的数据落在  $(\mu - \sigma, \mu + \sigma)$  间，约 95% 的数据落在  $(\mu - 2\sigma, \mu + 2\sigma)$  间，约 99.7% 的数据落在  $(\mu - 3\sigma, \mu + 3\sigma)$  间

2、高斯噪声：

是一种具有正态分布概率密度函数的噪声。在信号处理和图像处理领域较为常见。它的产生可能来源于电子电路中的热噪声，图像传感器中的噪声等。在图像中，高斯分布表示为随机分布的灰度值变化；

3、什么是扩散模型？正向过程，反向过程是：

扩散模型 (Diffusion Model) 是一类基于马尔可夫链的**生成模型**，它通过逐渐向数据中添加噪声并学习如何去噪来生成新的数据样本。扩散模型的核心思想源于非平衡热力学，旨在**学习数据的分布并且能够生成符合该分布的新样本**。

**正向过程**：也称为扩散过程，从真实的数据开始，逐渐添加少量高斯噪声，将数据样本逐渐转化为纯高斯噪声分布，数学表达如下：

$$x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon_{t-1}$$

其中 $\beta_t$ 是一个控制噪声添加量的噪声序列 $0 < \beta_1 < \beta_2 < \dots < \beta_T < 1$ ， $\epsilon_t$ 是服从标准正态分布的噪声向量，随着 $t$ 增大， $x_t$ 逐渐远离原始数据分布，趋向一个各向同性的高斯分布

**反向过程**：也称为去噪过程，它是正向过程的逆过程。从纯高斯噪声开始，逐步去除噪声，恢复到原始的数据分布，从而生成新的数据样本。反向过程通过神经网络，来预测给定 $x_t$ 的情况下如何得到 $x_{t-1}$ 。先预测 $\epsilon_t$ 再结合公式：

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( x_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \hat{\epsilon}_t \right) + \sigma_t \epsilon_t'$$

其中 $\alpha_t = 1 - \beta_t$ ， $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$ ， $\sigma_t$ 是控制生成随机性的参数， $\epsilon_t'$ 是服从标准正态分布的噪声向量。

#### 4、Diffusion 模型如何进行图像生成：

Diffusion 模型进行图像生成主要依赖于反向去噪过程：

(1)：从一个符合标准高斯分布的噪声图像 $x_T$ 开始，即生成一个具有随机像素值且符合高斯分布的图像；

(2)：通过训练好的去噪神经网络，从 $x_T$ 开始，逐步预测并去除噪声，生成 $x_{T-1}, x_{T-2}, \dots, x_0$ 等，按照上述公式，每次先由 $x_t$ 预测噪声 $\epsilon_t$ ，再反向计算 $x_{t-1}$

(3)：当迭代到 $x_0$ 时，理论上其应符合训练数据的分布。在训练过程中，通过最小化预测噪声与真实噪声的差异，来优化去噪神经网络的参数，使得网络能够准确预测噪声并进行去噪，生成高质量的图像；

#### 5、Diffusion 一定是 CNN 吗？

基于 CNN 能有效捕捉图像中的局部空间信息，在早期 Diffusion 模型实现中，常常使用 U\_Net 这种基于 CNN 的架构来进行去噪操作。

但随着研究发展，Diffusion 模型也可结合其他类型的神经网络架构，例如 Transformer 架构，可以更好捕捉长距离依赖关系，在处理一些需要全局理解的图像生成任务时，具有一定的优势。

因而 Diffusion 的去噪网络部分可以依据任务需求和设计选择不同的神经网络架构，而不局限于 CNN；