

复习范围

1. 人工智能学史 (1 获得, 0 未获得)

	图灵奖	诺贝尔奖
Geffrey Hinton	1	1
John Hopfield	0	1

2. 深度学习——“深度”：神经网络的深度（层数）。

3. 前馈神经网络 (Feed-Forward Neural Network, FNN)

特点：相邻层间神经元密集连接。

4. 全连接神经网络 (Full Connect Neural Network, FCNN)

- 前向传播：计算结果并保存梯度。

- 反向传播：链式规则。

5. 卷积神经网络 (Convolutional Neural Networks, CNN)

- 结构

卷积层：提取局部特征。

池化层：降低特征维度。

全连接层。

- 卷积计算过程

原图像尺寸： $H_0 = N_0 \times M_0 \times A_0$

卷积核尺寸：B 个， $F \times F \times A_0$

Padding: P

Stride: S

新图像尺寸： $N_1 = \frac{N_0 + 2P - F}{S} + 1$, $A_1 = B$

参数量： $F \times F \times A_0 \times B$

- 池化计算过程

最大池化 (Max Pooling)：选取区域的最大值作为代表性的特征值。

平均池化 (Average Pooling)：计算区域的平均值作为代表性的特征值。

池化层参数量为 0。

- 填充 (Padding)

增加感受野，减少信息损失：确保边缘像素能被卷积核充分覆盖，得到有效处理，而不是丢失。

控制输出尺寸：通过调整填充量可以精确控制每一层的输出尺寸。

- 步幅 (Stride)：

控制输出尺寸、下采样程度：较大步幅可以减小输出的空间尺寸，降低计算复杂度，减少参数量，提高特征图的缩放比例。

调整感受野：较大步幅意味着输出单元会覆盖较大输入区域，增加感受野，减少重叠区域数量。

平衡速度与精度：较大步幅可以加速计算过程，但可能丢失细节信息；较小步幅能更精细地捕捉特征，但会增加计算成本。

- 1×1 卷积

维度变换 (降维/升维)：改变特征图的深度 (通道数)。降维有助于降低模型复杂度和计算量，同时保持大部分有用信息。

非线性引入：在卷积后添加激活函数，可以在不改变空间尺寸的情况下引入非线性

性，使模型能够学习更复杂的模式。

作为瓶颈层：在一些架构中（如 ResNet、Inception），可以用作瓶颈层（先 1×1 卷积降维，再进行其他卷积，最后 1×1 卷积恢复维度）。显著减少参数数量和计算成本，同时维持性能。

特征融合：融合不同尺度或不同来源的特征图，合并成一个新的特征表示。

● 实例

LeNet：没有使用 ReLU。

AlexNet：最早使用了 ReLU，最早使用了 GPU。

VGGnet：小卷积核（感受野上，3 个 $3 \times 3=1$ 个 7×7 ）。

GoogLeNet：使用了 ReLU。Inception。 1×1 卷积。

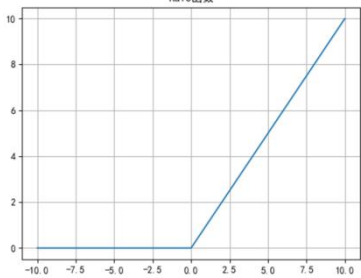
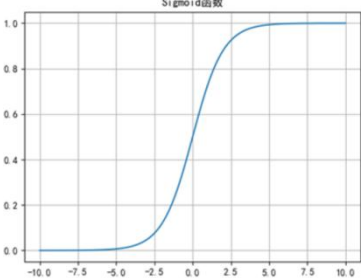
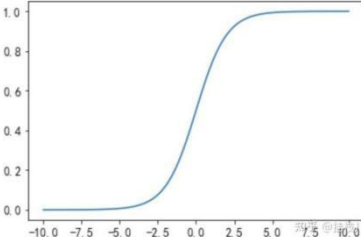
ResNet：使用了 ReLU。恒等映射直连边。残差模块。

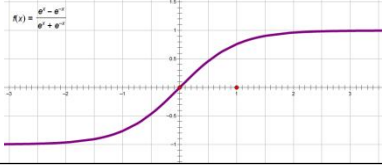
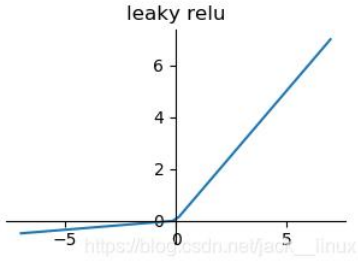
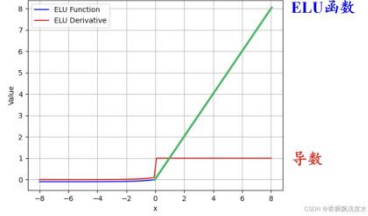
趋势：卷积核变小、层数增加，抛弃池化层、全连接层。

6. 损失函数

交叉熵损失函数更适用于分类问题，常用于衡量模型预测的概率分布与真实标记的概率分布之间的差异。

7. 激活函数

激活函数	函数	(导函数)	函数图像	优点	缺点
ReLU	$y = \max(0, x)$	$y' = \begin{cases} 1, & z > 0 \\ 0, & z \leq 0 \end{cases}$		计算简单，收敛速度快。 解决了梯度消失问题，允许误差迅速回传。 有助于稀疏激活，更简洁且减少过拟合风险。	存在“死区”问题（神经元输入小于 0 时将永远不会被激活）。 非零中心化，可能导致下一层权重更新不均衡。
Sigmoid	$y = \frac{1}{1 + e^{-z}}$	$y' = y(1 - y)$		输出范围固定在 (0, 1)，适用于二分类问题。 函数平滑且处处可微，有利于梯度计算。	易饱和，造成梯度消失问题。 非零中心化，输出总是正数，减慢收敛速度。 计算成本高。
Softmax	(对单个是 $y = e^x$) $y(x_i) = \frac{e^{x_i}}{\sum_{i=1}^n e^{x_i}}$			适用于多分类问题。	不适合作为隐藏层激活函数。 存在数值溢出问题。

tanh	$y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$		零中心化输出，提供更好的收敛性能。 适用于二分类问题。	存在易饱和和梯度消失问题。
Leaky ReLU	$y = \max(0.1x, x)$			
ELU	$y = \begin{cases} x, & x \geq 0 \\ a(e^x - 1), & x < 0 \end{cases}$			
Maxout	$y = \max(w_1^T x + b_1, w_2^T x + b_2)$			

● ReLU 相比于 Sigmoid 的优点：

梯度问题：ReLU 在正区间内不会导致梯度消失；Sigmoid 容易陷入饱和区，在区域内梯度几乎为零，导致梯度消失。

计算效率：ReLU 只需判断输入是否大于 0，Sigmoid 涉及指数运算，前者计算效率高于后者。

稀疏激活：ReLU 有助于产生稀疏激活，使模型更简洁，减少过拟合风险。

收敛速度：ReLU 可以使随机梯度下降更快收敛。

● Sigmoid 相比于 ReLU 的优点：

输出范围：Sigmoid 将输入映射到 (0, 1) 之间，ReLU 则没有进行约束，前者对特定任务（如二分类问题）更有效。

适用于特定架构或任务：在 RNNs 中，Sigmoid 被用于门控机制，控制信息流。

非线性表达能力：Sigmoid 在定义域上平滑且处处可微，非线性表达能力强。

8. 优化算法

不常用：二阶算法、牛顿法。

常用：SGD (+Momentum)、ADam、ADamW。

9. 参数

超参数：网络层数、学习率（学习率策略）、批量大小。

非超参数：网络权重（不唯一，比如倍数）。

超参数的获得：不能只在训练数据或测试数据上获得，要通过训练和验证获得。

10. 权重初始化

方法	特点	优点
高斯分布随机初始化	基于正态分布，权重应围绕零均值分布，并具有一定方差。	确保信号在传播中既不会迅速消失也不会爆炸。
Xavier 初始化	确保前向传播和反向传播过程中激活值和梯度的方差保持一致。	对于使用 tanh 或 sigmoid 激活函数的网络表现较好。
Kaiming 均匀分布初始化 (He 初始化)		对于使用 ReLU 及其变体激活函数的网络表现较好。

11. 正则化：防止过拟合

● 批量规范化 (Batch Normalization)

通过对每个小批量样本进行归一化，使输入特征具有零均值和单位方差，有助于减少梯度消失和梯度爆炸问题，从而加速收敛。其可以提高模型稳定性、泛化能力，并减少对其他正则化技术的依赖。

● 丢弃法 (dropout)

在训练过程中随机失活部分神经元（输出 0），减少模型对训练数据的过度依赖，阻止特征间自适应。可以防止过拟合，提高鲁棒性，间接减小计算量，实现模型平均效果。

● L1 正则化：使权重稀疏。

● L2 正则化：是权重分散。

● 早停。

12. 欠拟合和过拟合

	描述	训练收敛的网络	
		训练误差	测试误差
欠拟合	模型未训练收敛	大	大
过拟合	模型在训练集上表现良好，但不能推广到测试集	小	大

13. 梯度消失和梯度爆炸

	定义	问题	改善方法
梯度消失	网络层数增加，反向传播过程中梯度逐渐变小，甚至趋于零。 使用 sigmoid 或 tanh 等激活函数时，它们的导数在远离原点时变得非常小。	导致权重更新缓慢，甚至停止更新，从而影响模型的学习能力，使网络难以训练。	使用 ReLU 及其变种作为激活函数。 采用 Batch Normalization。 选择合适的权重初始化方法（如 Xavier 初始化或 He 初始化）。

梯度爆炸	在反向传播过程中，梯度值变得非常大。 网络权重更新过程中，因梯度的累积效应引起。	导致权重更新过大，使模型的学习过程不稳定，甚至发散，无法收敛到最优解。	使用 ReLU 及其变种作为激活函数。 采用 梯度截断 (Gradient Clipping)，即限制梯度大小，防止其变得过大。 使用 Adam 优化器，自动调整学习率，控制梯度大小。 采用 Batch Normalization，帮助稳定梯度，减少梯度爆炸影响。
------	---	-------------------------------------	---

14. 循环神经网络 (Recurrent Neural Network, RNN)

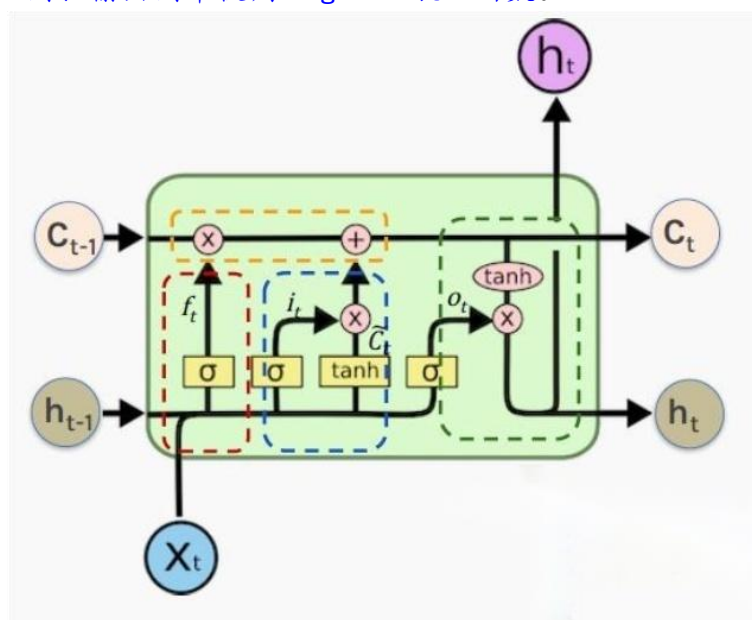
● 循环核计算

隐藏状态更新: $h_t = f_w(h_{t-1}, x_t) = \tanh(w_{xh}x_t + w_{hh}h_{t-1} + b_h)$

输出生成: $y_t = f_{wy}(h_t) = \text{softmax}(w_{hy}h_t + b_y)$

● 长短期记忆网络 (Long Short-Term Memory networks, LSTM)

遗忘门、输入门、输出门中使用 sigmoid 激活函数。



细胞状态 (Cell State)

贯穿模型，在整个序列上进行少量的线性交互，使信息流动而不发生太多改变。其中黄色框部分对应细胞状态的更新，其根据遗忘门和输入门的结果进行。

遗忘门 (Forget Gate)

对应上图红色框部分，决定从细胞状态中丢弃哪些信息。其查看前一时刻的隐藏状态 h_{t-1} 和当前输入 x_t ，输出介于 0 和 1 之间的值给细胞状态 c_{t-1} 的每个元素。0 意味着完全丢弃，1 意味着完全保留。

输入门 (Input Gate)

对应上图蓝色框部分，包含一个用于确定要更新哪些部分的 sigmoid 层，和一个创建新候选值向量的 tanh 层。

输出门 (Output Gate)

对应上图绿色框部分，决定细胞状态的输出。sigmoid 层决定细胞状态的哪些部分将输出，tanh 层缩放细胞状态，与 sigmoid 门的输出相乘，得到本时刻输出。

- 门控循环单元网络 (Gated Recurrent Unit networks, **GRU**)

只有两个门控机制 (重置门和更新门)，没有单独的细胞状态。

- **GRU 和 LSTM** 相比于传统 RNN，主要改善了梯度消失问题。

15. 注意力机制

- 组件包括键 (Key)、值 (Value)、查询 (Query)，不包括池化。

- 多头自注意力 (Multi-Head Self-Attention) 计算过程

■ 对于输入序列 $X = [x_1, x_2, \dots, x_n]$ ，其维度是 $[n, d_{\text{model}}]$ ，

■ 由权重矩阵 W^Q 、 W^K 、 W^V 生成查询、键、值：

$$Q = XW^Q$$

$$K = XW^K$$

$$V = XW^V$$

Q、K、V 的维度是 $[d_{\text{model}}, d_k]$ ， $d_k < d_{\text{model}}$ ，

■ 将其分割成 h 个头，每个头维度为 $\frac{d_k}{h}$ ：

$$Q_{\text{split}} = \text{Split}(Q)$$

$$K_{\text{split}} = \text{Split}(K)$$

$$V_{\text{split}} = \text{Split}(V)$$

■ 对每个头计算注意力分数 (点乘查询与键， d_k 缩放，softmax，加权求和)：

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

■ 合并头，其中 W^O 是一个可学习参数矩阵，维度为 $[hd_k, d_{\text{model}}]$ ，用来向 d_{model} 维映射：

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W^O$$

- 在注意力机制中使用位置编码

原始的自注意力机制不包含序列中元素顺序信息，计算仅基于元素间关系。为使模型能有效处理序列数据，并理解不同元素间的相对或绝对位置关系，引入位置编码。位置编码保持了序列信息，使其可以学习更复杂的模式，有助于提高模型的理解能力和表达能力。另外，位置编码还使得模型可以处理任意长度的序列，支持变长输入。

16. Transformer

在 Transformer 解码器中使用掩蔽注意力 (Masked Attention)

确保模型在生成序列时不依赖未来信息，只关注到它之前的元素。这能模拟真实场景，保证因果关系，维持了逻辑性和连贯性，这对于生成符合语法和语义规范的句子非常重要。

17. 自监督学习：使用输入中的一部分预测另一部分，即将部分输入作为标签，无须人工标注标签。

18. 生成式模型

- 预训练 (Pre-training)：在大型、通用数据集上训练模型，让模型学习到输入数据中的基本特征和模式。
- 微调 (Fine-tuning)：在预训练后，将模型应用到具体任务上，并根据该

任务的相关数据集进行进一步训练，调整参数，以优化特定任务的表现。

- **分两步：**
 - **迁移学习：**预训练模型已经在大量数据上学到了丰富的特征，在新任务上只需对模型进行少量调整就可以获得良好效果。
 - **充分利用数据，适应不同任务需求：**有效结合预训练大数据集带来的**泛化能力**和微调小数据集中特有的领域信息。
 - **节省时间和成本：**获取大规模标注数据是困难而昂贵的，预训练-微调允许在较小规模的数据集上快速开发和部署模型，大大减少了从零开始训练所需的时间和计算资源，同时还能保持较高的性能水平。
- **采用自监督学习的预训练方法：**
 - **数据效率：**使模型能够利用丰富的未标注资源学习有用的表示，而不需要为每个样本都提供昂贵的手工标注。
 - **泛化能力：**通过在广泛无关数据上训练，可以学到更通用的语言理解能力和其他领域知识，提高模型对不同任务的适应性和泛化能力。
 - **减少过拟合风险：**相比于在小规模标注数据集上直接训练，在更大规模的数据集上预训练减少了过拟合风险。
 - **加速后续学习：**预训练得到的特征可以帮助模型更快地收敛到特定任务的最佳解。

19. 物体检测与图像分割

- 语义分割与实例分割。
- 物体检测：R-CNN，FastR-CNN。
- 转置卷积：可学习的上采样方法。

20. 可视化与理解

- 可视化：卷积核、输出层。
- 理解：重要像素、显著性。
- 对抗性扰动。
- 风格迁移。