深度学习期末复习 · 全 Wiki 合集

> 由 `wiki/` 自动合并导出。公式 LaTeX、结构图 Mermaid；图片已 base64 内嵌。

目录

深度学习期末复习 Wiki

面向期末考试的复习资料。**课本（李航《机器学习方法》）= 知识全集**；本 Wiki 在此基础上， 结合**课件、课堂录音（老师强调）、同学 LSQ 的思维导图、作业**，把**最可能考的内容**梳理成 人话 + 公式 + 流程图，并标注每个知识点要掌握到什么程度。

> 实事求是说明：内容多的地方我尽量写全；个别太深或老师明确"了解即可"的，会**写梗概并标注** `> 🚧 此处需拓展`，提示你需要时再回课本/课件深挖。公式用 LaTeX、结构图用 Mermaid（GitHub / VS Code Markdown 预览可直接渲染），少量必须的真实插图复制在各章 `assets/` 下、相对引用，整个 `wiki/` 可独立携带。

📌 考试信息（来自教材封面手写，已交叉核对）

- **范围**：第 7 章 SVM、第 23 章 前馈神经网络、第 24 章 CNN、第 25 章 RNN、第 26 章 序列到序列/Transformer、第 29.1 节 优化算法。
- **题型（4 种）**：填空题、简答题、计算题、解答题。
- **可带计算器。**
- **优化器必须掌握**：SGD（批量/随机）、动量 SGD、RMSProp、Adam；额外补充 AdaDelta、AdaGrad、AdamW。

🗺️ 先看这个

00-复习重点地图.md —— 各章重要度排序 + 每个知识点"要掌握到什么程度"（背/会写/会画/会算/了解），这是复习的总导航。

📚 章节目录（按重要度排序）

优先级	章节	一句话考点
🔴 高	01-前馈神经网络	会写网络数学模型+画图、4 种激活函数公式及导数、反向传播 3 式、实现 XOR
🔴 高	02-支持向量机	三类 SVM 的模型与算法**要背**、对偶问题+KKT、软间隔、核函数
🟠 必考	04-优化算法	SGD/动量/RMSProp/Adam 等更新公式与区别（封面点名必考）
🟡 中	03-卷积神经网络	卷积/池化/输出尺寸**计算题**、CNN 定义公式要会写、三大性质
🟡 中	06-序列到序列与 Transformer	注意力公式、Transformer 七大技术、多头注意力、图 26.7
🟢 基础	05-循环神经网络	**5 种 RNN 架构要会画**、LSTM/GRU 门控、梯度消失
🖍️ 练习	07-练习题	作业原题 + 详解：NN 前向反向、CNN 卷积/池化、SVM 核+对偶（计算题主力）

图例约定

- > 🎯 考点 ： 期末很可能考、要重点准备的。
- > ✍️ 要求 ： 到底要掌握到什么程度（背 / 会写 / 会画 / 会算 / 了解）。
- > 🚧 此处需拓展 ： 我只写了梗概，需要时回课本/课件深挖。
- > ⚠️ 坑 ： 易错点、老师特别提醒处。

00 • 复习重点地图（总导航）

> 三路信号合一：**课堂录音**（老师讲了多久 + 强调原话）、**同学 LSQ 思维导图**（每个点要掌握到什么程度）、**作业/课件/手写批注。** ⚠️ 考试范围内 6 块**都要过**，下面的"优先级"只代表该花多少力气。

一览表

优先级	章节	课堂讲解	必须达到的程度（综合 LSQ + 录音）
🔴 高	第23章 前馈神经网络	~36 min（最多）	**会写**网络数学模型（线性变换用 $z = W^{\mathrm{T}}x + b$，**公式里偏置 $+b$ 必写**，不写扣分；图上画不画都行）并**画图**；**会写 4 种激活函数公式及导数**；**会写**反向传播 3 个核心式；**会**实现 XOR（例 23.3）；损失函数/学习的一般与具体形式
🔴 高	第7章 支持向量机	~35 min	三类 SVM（硬间隔/软间隔/非线性）的**模型与算法都要背**；对偶问题数学模型 + **KKT 条件**；分离超平面与决策函数；例题 7.2 会算；软间隔注意五角星区别
🟠 必考	第29.1 优化算法	融入第23章讲	**SGD/动量/RMSProp/Adam** 等更新公式与区别（封面点名"必须掌握"）；BN、Dropout 等技巧
🟡 中	第24章 卷积神经网络	~14 min	**计算题**：二维卷积、填充与步幅、输出矩阵尺寸、三维卷积、池化（最大/平均/求和）；**CNN 定义公式一定要会写**；三大性质；反向传播=一次前向+一次反向；能说出常见模型名
🟡 中	第26章 序列到序列/Transformer	~5 min（强调密度高）	注意力机制公式；RNN Search（26.2.2 定义、图 26.5）；Transformer **七大技术必须会**、关注图 26.7 与多头注意力、模型特点
🟢 基础	第25章 循环神经网络	~10 min	**5 种架构要会画**：SRNN(图25.1)、LSTM(定义25.2/图25.6)、GRU(定义25.3/图25.8)、深度RNN(图25.9)、双向RNN(图25.10)；反向传播是核心算法；梯度消失/爆炸概念

按题型倒推怎么准备

- **计算题**（可带计算器）：CNN 卷积/池化/输出尺寸（必出）、前馈网络前向+反向数值计算、SVM 对偶/核矩阵（例 7.2）。→ 见各章"计算"小节。

- **解答/简答题**：SVM 三类模型与算法推导、反向传播推导、激活函数及导数、正则化方法、Transformer 七大技术。
- **画图题**：前馈网络结构（偏置可画可不画，但数学模型公式里 $+b$ 必写）、RNN 五种架构、计算图。
- **填空题**：定义、公式要素、各优化器特点、CNN 性质、常见模型名。

各章速记导航

- 01 前馈神经网络 → 神经元/网络结构、激活函数、反向传播、正则化技巧
- 02 支持向量机 → 三类 SVM、间隔、对偶+KKT、软间隔+核
- 03 卷积神经网络 → 卷积/池化计算、CNN 定义与性质
- 04 优化算法 → 梯度下降家族（SGD→Adam）
- 05 循环神经网络 → 五种架构、LSTM/GRU
- 06 序列到序列与Transformer → 注意力、Transformer
- 07 练习题（带详解） → 作业原题 + 详解，计算题为主

> 来源说明：录音强调原话见 ../txt/voice/（带时间戳）；LSQ 思维导图原始大纲见 ../txt/aligned/xmind_outline.md；量化重要度见 ../txt/复习重点地图.md。

第 23 章 · 前馈神经网络（🔴 最高优先级）

课堂讲解最多（≈36 分钟）、强调最多、批注最多、作业最多。这是整门课的**核心**， 也是后面 CNN/RNN 的基础。

> 🎯 考点：本章几乎每种题型都会出——画图题（网络结构）、计算题（前向+反向）、 简答题（激活函数/正则化）、填空题（定义/公式要素）。

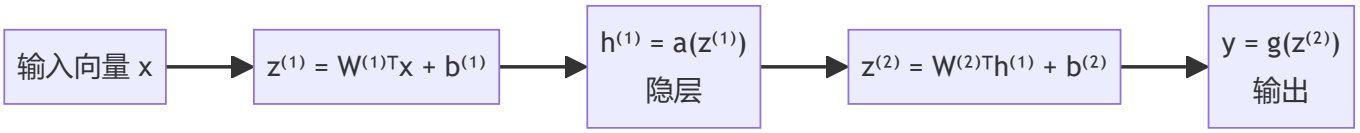
本章必须达到的程度（来自 LSQ 思维导图 + 老师录音）

知识点	要求	对应小节
网络数学模型 + 画图	**会写、会画**（公式 $z = W^{\mathsf{T}}x + b$ 的**偏置 $+b$ 必写**，不写扣分；图上偏置可画可不画）	01-神经元与网络结构
4 种激活函数	**会写公式及其导数**	02-激活函数
实现 XOR（例 23.3）	**会写**	01-神经元与网络结构
学习算法（损失/一般与具体形式）	**会写数学形式**	03-学习算法与反向传播
反向传播 3 个核心式	**会写、会推**	03-学习算法与反向传播
计算图	会画（⚠️ 书 图23.22–23.24 反向箭头印错了，应逆向）	03-学习算法与反向传播
优化算法	掌握数学形式/概念（详见 第29.1 优化算法）	03
实现技巧 / 正则化	了解概念（能简答）	04-正则化与训练技巧

> ✍️ 老师原话（录音）：「神经网络首先第一个要掌握什么叫神经元」「这本书画前馈神经网络的时候都会把偏置画上去……这是要掌握的」「导数长啥样很重要，因为后面要用导」「这 3 个式子是需要掌握的」。

一句话总览

前馈神经网络 = **多层神经元的复合非线性函数**。给定输入向量 → 逐层「线性变换 + 非线性激活」→ 输出。学习 = 用**反向传播**算出损失对所有参数（$\boldsymbol{W}, \boldsymbol{b}$）的梯度，再用**梯度下降**更新。



子页：

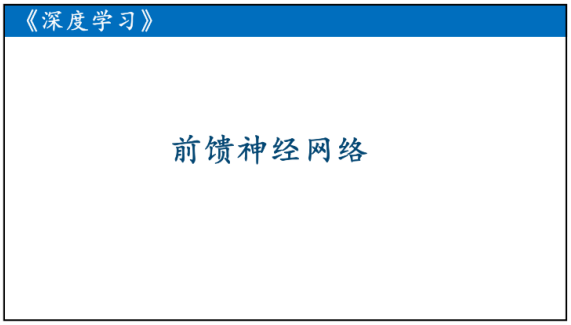
1. 神经元与网络结构
2. 激活函数（4 种 + 输出层 softmax）
3. 学习算法与反向传播
4. 正则化与训练技巧

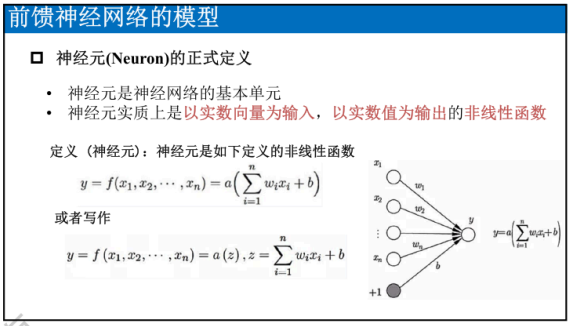
课件原图参考

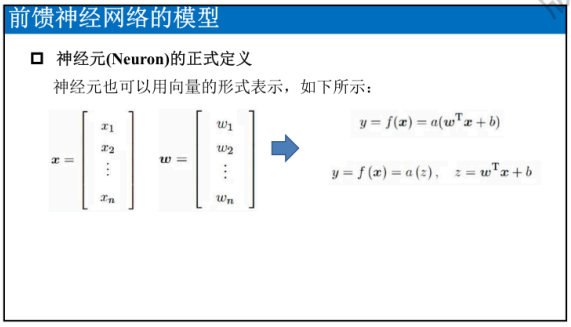
> 来自课程 PPT（4 合 1 讲义版），对照本章公式/结构图看老师的原图。

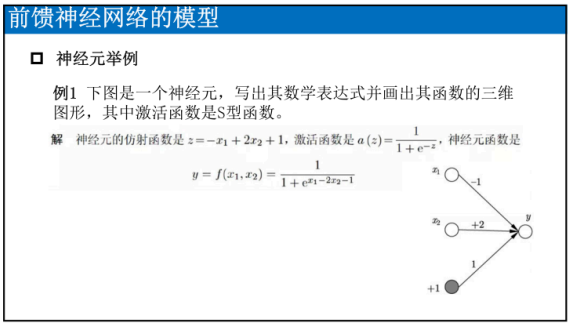
神经元定义与网络结构：

2026/6/16



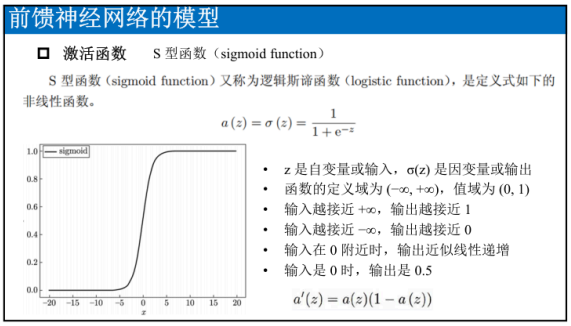


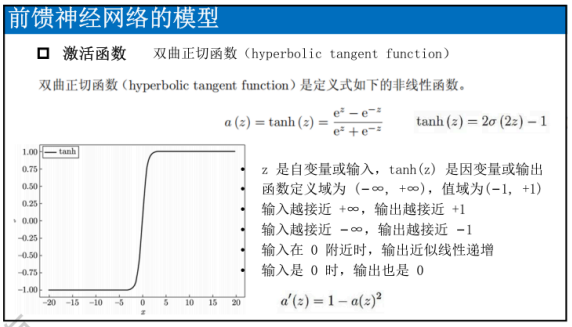


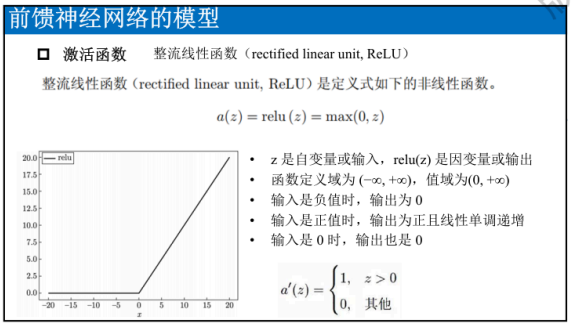


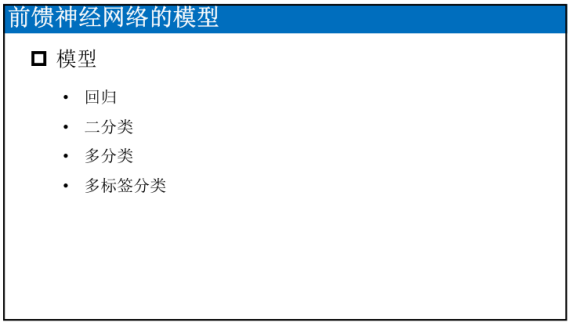
四种激活函数（Sigmoid/Tanh/ReLU）：

2026/6/16









01 • 神经元与网络结构

> ✍ 要求：网络数学模型**会写**（线性变换 $z = W^{\mathsf{T}}x + b$，**公式里偏置 $+b$ 必写**，不写扣分；图上偏置画不画都行）、结构**会画**、能实现 XOR。

1. 神经元（Neuron）

神经元是神经网络的**基本单元**：以实数向量为输入、实数为输出的**非线性函数。**

标量形式：

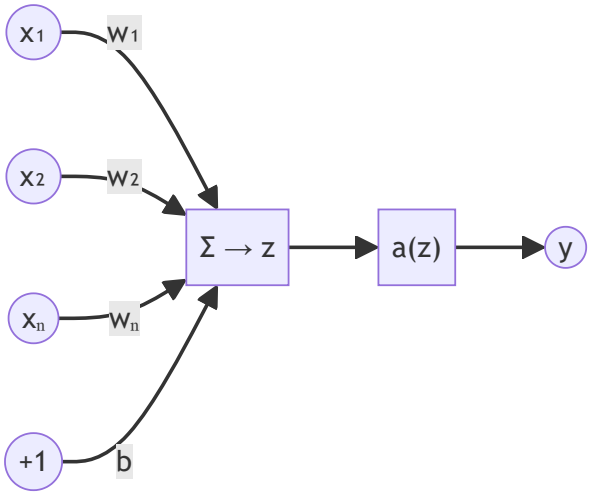
$$y = f(x_1, \ldots, x_n) = a\left(\sum_{i=1}^{n} w_i x_i + b\right) = a(z), \qquad z = \sum_{i=1}^{n} w_i x_i + b$$

向量形式（考试更常用，老师强调用转置）：

$$y = f(\boldsymbol{x}) = a(\boldsymbol{w}^{\mathsf{T}}\boldsymbol{x} + b) = a(z), \qquad z = \boldsymbol{w}^{\mathsf{T}}\boldsymbol{x} + b$$

- z 叫**仿射函数 / 净输入**（线性变换部分）；$a(\cdot)$ 是**激活函数**（非线性部分）。
- $\boldsymbol{w}$ 一般写成**列向量**；$\boldsymbol{w}^{\mathsf{T}}\boldsymbol{x}$ 就是点积。
- $\boldsymbol{x}$ 的各分量是一个样本的**属性**（如身高、体重），不是多个样本。

> ⚠️ 坑：$x_1, \ldots, x_n$ 是**一个样本**的 n 个属性构成的向量，**不是 n 个样本**（老师说往年很多人搞错）。



2. 前馈神经网络

前馈神经网络：由多层神经元组成；**层间全连接、层内不连接**；前一层的输出是后一层的输入；信息单向前传（无环）。

- 只能处理**向量型输入**。图像这类**矩阵**数据必须先**展平成列向量**才能喂进去（这正是 CNN 出现的动机之一）。

两层（双层）前馈神经网络

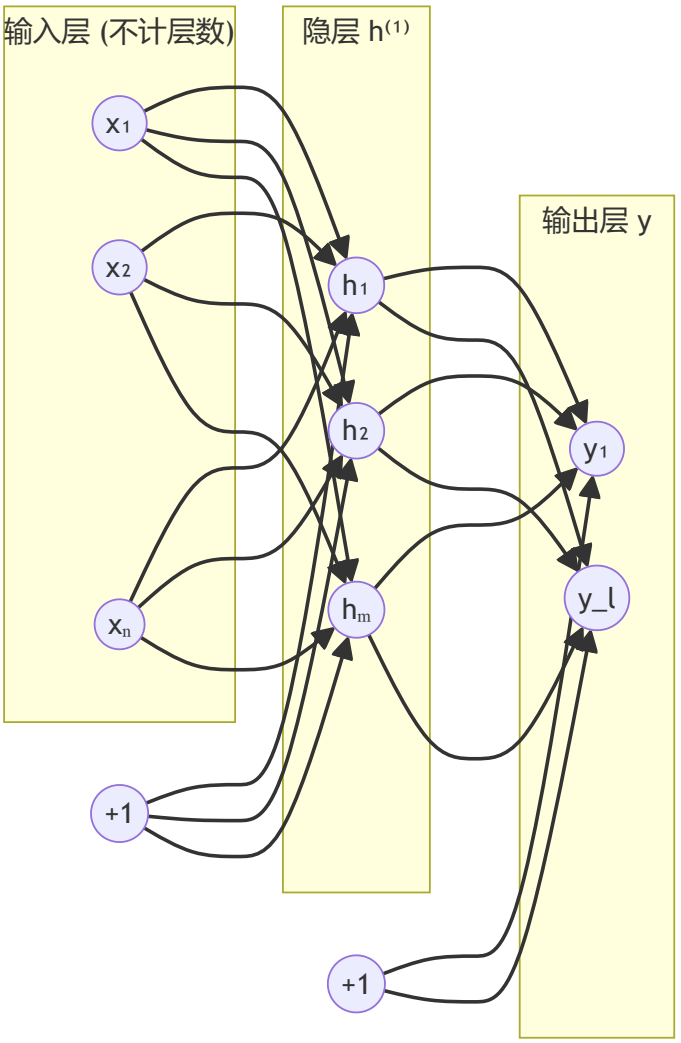
> ⚠️ 为什么叫"双层"？输入层一般**不算一层**，所以"输入层 + 隐层 + 输出层"算**两层**（= 隐层 + 输出层），也叫"单隐层"。

矩阵形式（要会写）：

$$\boldsymbol{h}^{(1)} = \boldsymbol{f}^{(1)}(\boldsymbol{x}) = a\left(\boldsymbol{W}^{(1)\mathsf{T}}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right)$$

$$\boldsymbol{y} = \boldsymbol{f}^{(2)}(\boldsymbol{h}^{(1)}) = g\left(\boldsymbol{W}^{(2)\mathsf{T}}\boldsymbol{h}^{(1)} + \boldsymbol{b}^{(2)}\right)$$

整体是复合函数 $\boldsymbol{y} = \boldsymbol{f}^{(2)}(\boldsymbol{f}^{(1)}(\boldsymbol{x}))$。$a(\cdot)$ 隐层激活，$g(\cdot)$ 输出层激活。



> 🎯 偏置：**数学模型/公式里必须写出偏置 $+b$**（$z = W^{\mathsf{T}}x + b$，老师强调不写要扣分）；**结构图上偏置画不画都行**（老师确认；课本习惯是把 +1 节点及其连接画上，照画更稳妥）。

多层（深度）前馈神经网络

定义：s 层（$s \geq 2$），第 1 到 $s-1$ 层是隐层。第 t 层第 j 个神经元：

$$h_j^{(t)} = a\left(z_j^{(t)}\right) = a\left(\sum_{i=1}^{n} w_{ji}^{(t)}\, h_i^{(t-1)} + b_j^{(t)}\right), \qquad h_i^{(0)} = x_i$$

> 本质：**深度学习 = 深度神经网络 = 具有深度结构的人工神经网络**（本课定义）。前馈网络的本质是**多次线性变换与非线性变换的组合。**

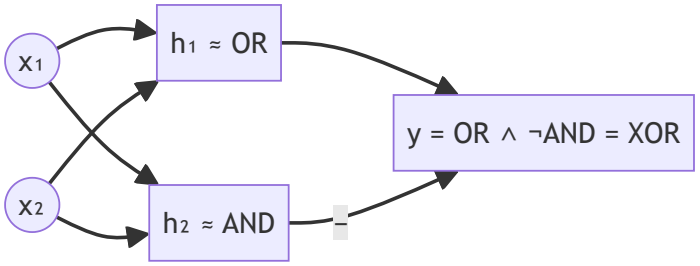
例：异或 (XOR) {#例-异或-xor}

> ✍️ 要求：**会写**实现 XOR 的网络（例 23.3）。XOR **线性不可分**，单个神经元（感知机）做不到，必须用**带隐层**的网络。

XOR 真值表：$(0,0)\to 0,\ (0,1)\to 1,\ (1,0)\to 1,\ (1,1)\to 0$。

一种经典构造（隐层 2 个神经元，激活用阶跃/ReLU 思路）：

- 隐层：$h_1 = a(x_1 + x_2 - 0.5)$（相当于 OR），$h_2 = a(x_1 + x_2 - 1.5)$（相当于 AND）。
- 输出：$y = a(h_1 - h_2 - 0.5)$，即 **OR 且非 AND** = XOR。



> 🚧 此处需拓展：例 23.3 在书 P382 附近还配了**三维函数图形**（用 Python 画），老师说"图别管，掌握数学表达式怎么写"即可。具体权重数值以课本例题为准。

> 📌 关联：前馈网络用于分类的输出层选择，见 02-激活函数。

02 · 激活函数（4 种）+ 输出层

> 🎯 考点 / ✍️ 要求：**会写公式，更要会写导数**（老师："导数长啥样很重要，因为后面要用导"——反向传播要用）。

激活函数给网络引入**非线性**。隐层常用以下几种（按发展顺序）。

1. 二值 / 阶跃激活函数

最早的激活，输出离散（如 $\{+1,-1\}$ 或 $\{0,1\}$）：

$$a(z)=\begin{cases}1, & z\ge 0\\ 0, & z<0\end{cases}$$

- 导数：除 $z=0$ 外处处为 0（不可导/不利于梯度学习）→ 所以后续被连续激活取代。

2. S 型函数（Sigmoid / Logistic）

$$\sigma(z) = \frac{1}{1+e^{-z}} \in (0,1)$$

导数（要会写，形式很漂亮）：

$$\sigma'(z) = \sigma(z)\big(1-\sigma(z)\big)$$

- 输出可当概率（二分类输出层常用）。
- ⚠️ 缺点：两端**饱和**（导数趋 0）→ **梯度消失**；输出非零中心。

3. 双曲正切（tanh）

$$\tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \in (-1,1)$$

导数：

$$\tanh'(z) = 1-\tanh^2(z)$$

- 零中心，通常比 Sigmoid 好；但同样会饱和、梯度消失。

4. 整流线性单元（ReLU）

$$\mathrm{ReLU}(z) = \max(0,z) = \begin{cases} z, & z>0 \\ 0, & z\le 0 \end{cases}$$

导数（分段、不连续）：

$$\mathrm{ReLU}'(z) = \begin{cases} 1, & z>0 \\ 0, & z<0 \end{cases}$$

- 优点：正区间不饱和、缓解梯度消失、计算快。缺点：负区间梯度为 0（"死亡 ReLU"）。

> ✍️ 小结记忆表（必背公式 + 导数）：

函数	$a(z)$	$a'(z)$	值域
Sigmoid	$\frac{1}{1+e^{-z}}$	$\sigma(1-\sigma)$	$(0,1)$
tanh	$\frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	$1-\tanh^2 z$	$(-1,1)$
ReLU	$\max(0,z)$	$\mathbb{1}[z>0]$	$[0,\infty)$
阶跃	$\mathbb{1}[z\ge 0]$	0（a.e.）	$\{0,1\}$

> 🚧 此处需拓展：若课件把"第 4 种"定为 LeakyReLU / Softplus 而非阶跃，公式为 $\mathrm{LeakyReLU}(z)=\max(\alpha z, z)$、$\mathrm{Softplus}(z)=\ln(1+e^z)$（导数即 Sigmoid）。**以课件 PPT 列出的 4 种为准。**

输出层与任务匹配 {#输出层与任务匹配}

输出层激活 $g(\cdot)$ 按**任务**选（老师重点讲）：

任务	输出层激活	损失函数
回归	线性 $g(z)=z$	平方误差
二分类	Sigmoid	二值交叉熵
多分类	**Softmax**	交叉熵
多标签分类	每个神经元各用 Sigmoid	二值交叉熵

Softmax（多分类输出，重点）

把 l 维实向量 $\boldsymbol{z}$ 映射为概率向量 $\boldsymbol{p}$（$p_k \geq 0,\ \sum_k p_k = 1$）：

$$p_k = g(z_k) = \frac{e^{z_k}}{\sum_{i=1}^{l} e^{z_i}}$$

- 名字来源：它是 $\max$ 的**光滑近似**——若 $z_k \gg z_j$，则 $p_k \approx 1$、其余 ≈ 0。
- 预测时可省略 softmax，直接取**净输入 z_k 最大**的类别（结果等价，分母对各类是常数、指数单调递增）。z_k 又叫**对数几率 (logit)**。

Softmax 的偏导（雅可比，书式 23.24）：

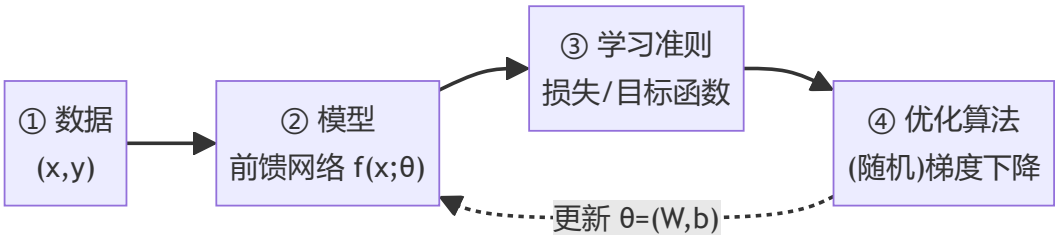
$$\frac{\partial p_k}{\partial z_j} = \begin{cases} p_k(1-p_k), & j = k \\ -p_j p_k, & j \neq k \end{cases}$$

> 📌 这条导数在多分类反向传播里要用，见 03-学习算法与反向传播。

03 · 学习算法与反向传播

> 🎯 考点（高频）：**反向传播 3 个核心式要会写会推**；损失的**一般形式与具体形式**要会写；计算图会画。 老师："这 3 个式子是需要掌握的"。

1. 学习的四要素



参数 $\boldsymbol{\theta}$ = 所有层的 $\boldsymbol{W}$ 和 $\boldsymbol{b}$ 的集合。前提：网络结构（层数、每层神经元数、各层类型）已定。

2. 损失函数

一般形式（经验风险 + 正则项）：

$$\mathcal{L}(\theta)=\underbrace{\frac{1}{N}\sum_{i=1}^{N}\ell\big(f(\boldsymbol{x}_i;\theta),\boldsymbol{y}_i\big)}_{\text{empirical loss}}+\underbrace{\lambda R(\theta)}_{\text{regularizer}}$$

目标让预测 $\boldsymbol{f}$ 与真实 $\boldsymbol{y}$ 尽量接近；等价于**极大似然** → 损失 = 负对数似然。

具体形式（按任务，要会写）：

任务	单样本损失 ℓ
回归	平方误差 $\frac{1}{2}\|\boldsymbol{f}-\boldsymbol{y}\|^2$
二分类	二值交叉熵 $-\big[y\ln f+(1-y)\ln(1-f)\big]$
多分类	交叉熵 $-\sum_k y_k\ln p_k$（$\boldsymbol{y}$ 为 one-hot）

> ⚠️ 二分类里 $y\in\{0,1\}$，所以 y 与 $1-y$ 恰好"二选一"地激活两项；多分类用 one-hot 后同理只剩真实类那一项。

3. 前向传播（Forward）

按网络公式逐层算到输出（见 01）：$\boldsymbol{z}^{(t)}=\boldsymbol{W}^{(t)\mathsf{T}}\boldsymbol{h}^{(t-1)}+\boldsymbol{b}^{(t)}$，$\boldsymbol{h}^{(t)}=a\big(\boldsymbol{z}^{(t)}\big)$，最终得 $\hat{\boldsymbol{y}}$ 与损失。

4. 反向传播（Backward）——核心 3 式 ⭐

定义**误差项** $\boldsymbol{\delta}^{(t)}=\dfrac{\partial\mathcal{L}}{\partial\boldsymbol{z}^{(t)}}$（损失对第 t 层**净输入**的导数）。

① **输出层误差项**（分类用 softmax+交叉熵、回归用平方误差时，形式都很简洁）：

$$\boxed{\boldsymbol{\delta}^{(s)} = \hat{\boldsymbol{y}} - \boldsymbol{y}}$$

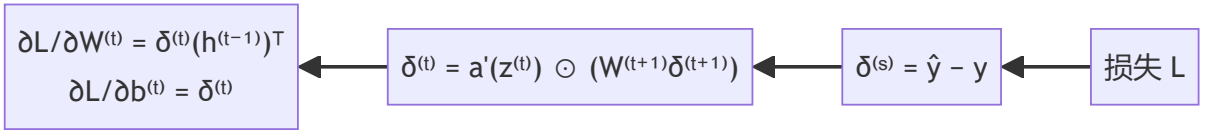
② **误差项逐层反向递推**（当前层依赖**后一层**，所以是"反向"）：

$$\boxed{\boldsymbol{\delta}^{(t)} = a'\left(\boldsymbol{z}^{(t)}\right) \odot \left(\boldsymbol{W}^{(t+1)}\boldsymbol{\delta}^{(t+1)}\right)}$$

（⊙ 为逐元素乘；这里要用到激活函数的导数 a'——所以前面才强调"导数很重要"。）

③ **损失对参数的梯度**（拿到 δ 就能求全部梯度）：

$$\boxed{\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}^{(t)}} = \boldsymbol{\delta}^{(t)}\left(\boldsymbol{h}^{(t-1)}\right)^{\mathsf{T}}, \qquad \frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^{(t)}} = \boldsymbol{\delta}^{(t)}}$$

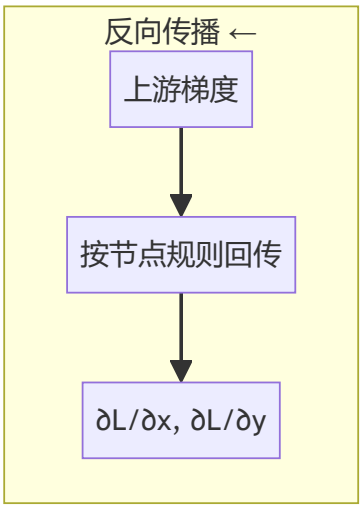


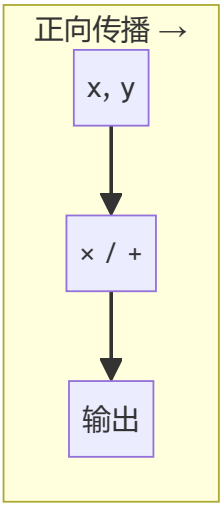
> ✍️ 记忆口诀：**输出层 δ = 预测 − 真实；往前传一层乘激活导数和权重；参数梯度 = δ × 前一层输出。**

5. 计算图 {#计算图}

计算图把复合函数拆成节点，每个节点都有**正向**（左→右求值）和**反向**（右→左传梯度）两步。

- **乘法节点** $f = x \cdot y$：反向时 $\partial f/\partial x = y$、$\partial f/\partial y = x$（梯度互换乘上游）。
- **加法节点** $f = x + y$：反向时梯度原样分发给两路。





> ⚠️ 坑（老师 & LSQ 都点了）：**课本 图 23.22–23.24 里反向传播的箭头方向印错了**，反向传播应当是**从右往左（逆向）**。考试若画计算图，反向箭头记得画成逆向。

6. 优化：梯度下降 → 小批量 SGD

参数更新（沿负梯度）：

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}, \qquad \eta: \text{ learning rate}$$

- **批量梯度下降 (GD)**：每步用**全部 N 个样本**——精确但样本量大时**内存爆炸**。
- **小批量随机梯度下降 (mini-batch SGD)**：先打乱，分成 M 组、每组 n 个（$N = M \times n$），每步只用一个小批量 (batch)。公式同上，只是把 N 换成 n。这是深度学习实际用的。

> 神经网络损失对参数是**高度非凸**优化问题（老师强调）。更细的优化器（动量/RMSProp/Adam）见 第 29.1 优化算法。

> 📌 反向传播整体 = **一次正向传播 + 一次（误差/梯度的）反向传播**。这是所有深度学习模型（含 CNN/RNN）通用的学习骨架。

04 • 正则化与训练技巧

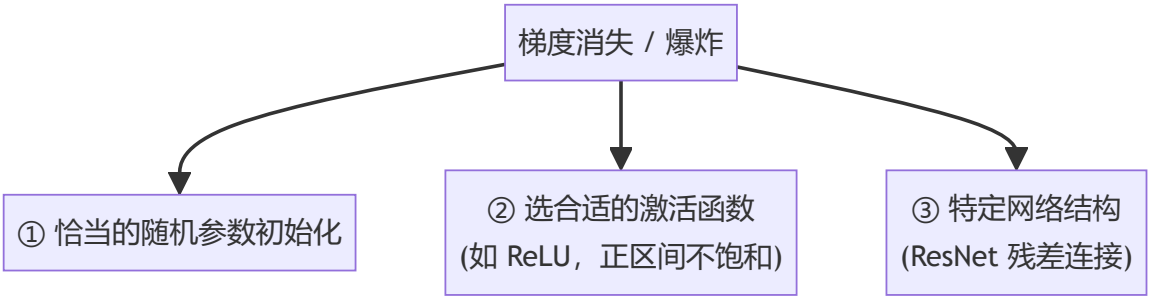
> ✍️ 要求：**了解概念、能简答**（老师列为"通过 ppt 了解"）。常作简答/填空题。

训练深网会遇到三类问题：**梯度消失/爆炸**、**内部协变量偏移**、**过拟合**。对应三组技巧。

1. 梯度消失与梯度爆炸

反向传播中梯度逐层连乘：若每层因子普遍 < 1 → 越传越小（**消失**）；普遍 > 1 → 越传越大（**爆炸**）。

三种解决思路：



- **ResNet 残差连接**：让梯度有"高速通道"直接回传，有效防止梯度消失/爆炸（CNN/Transformer 都用）。

2. 内部协变量偏移（Internal Covariate Shift）

定义：训练时**中间层参数不断变化 → 该层输入分布也跟着变**，导致后层要反复适应、学习变慢。

两类归一化（本质不同，但都缓解此问题）：

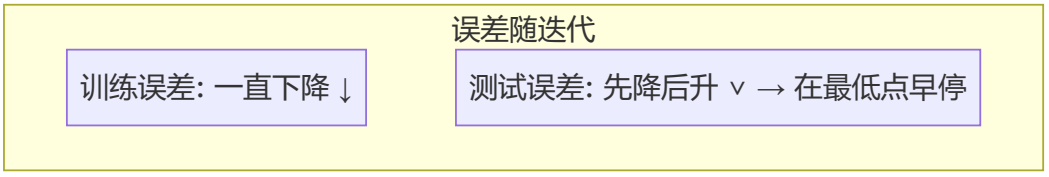
方法	归一化对象
批归一化 (BN)	对一个**小批量**在每个特征维上归一化
正则化 / 层归一化	对**单个样本/神经元**层面约束

> 🚧 此处需拓展：BN 的具体算法（求批量均值方差 → 标准化 → 再加可学习的缩放 γ 与平移 β）课本有完整流程，需要时回 ppt/课本看算法步骤。

3. 正则化（缓解过拟合）

目标：提高**泛化能力**（测试集上的表现）。常见 **4 种**：

方法	做法	备注
L1 正则化	损失加 $\lambda\|\theta\|_1$	产生稀疏
L2 正则化 / 权重衰减	损失加 $\lambda\|\theta\|_2^2$	两者近似等价
Dropout	每次迭代随机**冻结一部分神经元**不参与训练	等价于训练许多"退化子网络"
早停 (Early Stopping)	训练误差一直降，但**测试误差先降后升**；在测试效果最好处停	见下图



$$\mathcal{L}_{\text{reg}}(\theta) = \mathcal{L}(\theta) + \lambda R(\theta), \quad R(\theta) = \|\theta\|_1 \ (\text{L1}) \text{ or } \|\theta\|_2^2 \ (\text{L2})$$

本章一句话回顾

> 网络数学模型与画图（含偏置）→ 4 种激活函数公式及导数 → 损失（一般式+具体式）→ 反向传播 3 式 → 优化（mini-batch SGD）→ 训练技巧（梯度问题/BN/正则化）。这条主线贯穿全章，也是 CNN/RNN 的基础。

第 7 章 · 支持向量机 SVM（🔴 最高优先级）

课堂讲解 ≈35 分钟（最多之一），第三节课又重讲了 20 分钟。**数学模型老师说"背也得背下来"**。

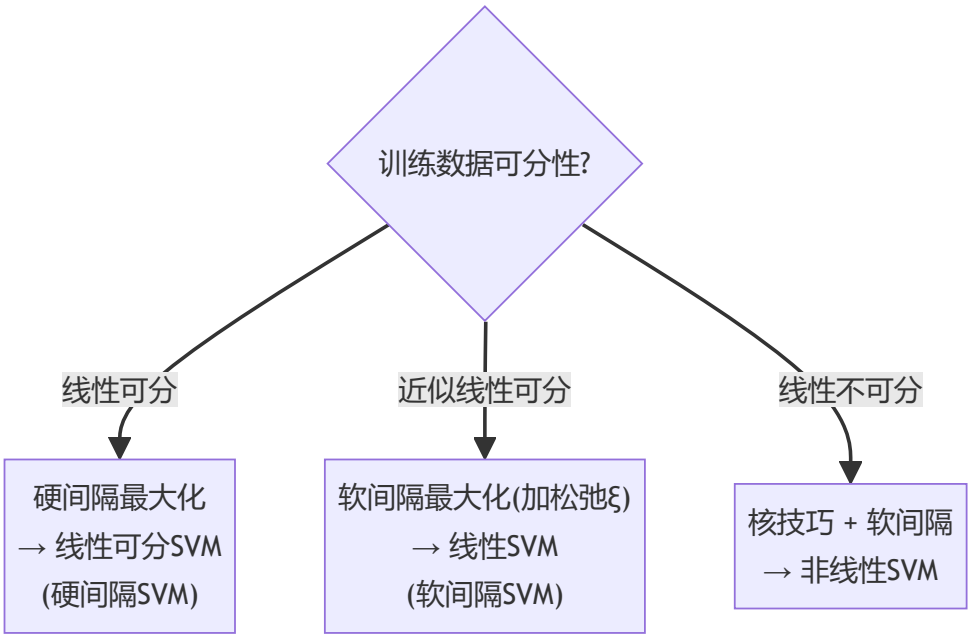
> 🎯 考点：解答/计算题主力——三类 SVM 的模型与算法推导、对偶问题、KKT、核矩阵计算（例 7.2）。 ✍️ LSQ 原话："他说数学模型背也得背下来"；软间隔处"特别注意我打五角星的区别"；"三类可分要记一下，他复习完还提问"。

本章要求（LSQ + 录音）

知识点	要求	小节
基本概念、三类 SVM	**要记**（按训练数据类型分 3 类）	01-基本概念与三类SVM
函数间隔 / 几何间隔	了解概念	02-间隔与硬间隔SVM
间隔最大化、支持向量定义	会写、会推	02-间隔与硬间隔SVM
对偶问题数学模型 + KKT 条件	**要背、会推**	03-对偶问题与KKT
分离超平面 / 决策函数	会写	03-对偶问题与KKT
例题 7.2	**会算**	03-对偶问题与KKT
软间隔（⚠️ 五角星区别）	会写模型与算法	04-软间隔与核函数
合页损失、核函数	了解概念	04-软间隔与核函数

> ppt 明确要求掌握：① **硬间隔线性可分 SVM 的模型、算法**；② **软间隔线性 SVM 的模型、算法**；③ **非线性 SVM 的模型、算法**（基于变换的 SVM 与基于核的 SVM）。

一图总览：三类 SVM 怎么来的



> 核心思想一句话：**找一个分离超平面，使两类样本的间隔最大。**间隔最大 → 解唯一（感知机解不唯一）。

子页：

1. 基本概念与三类 SVM
2. 间隔与硬间隔 SVM
3. 对偶问题与 KKT（核心）
4. 软间隔与核函数

📷 课件原图参考

> 来自课程 PPT（4 合 1 讲义版）。

硬间隔 / 几何间隔：

硬间隔线性支持向量机——最优分离超平面

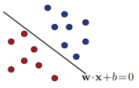
- SVM的最优分离超平面
- 训练样本线性可分：

存在一个超平面

$$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$$

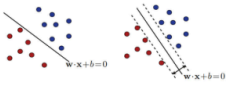
使得对于任意的样本$(\boldsymbol{x}_i, y_i)$，存在$(\boldsymbol{w},\ b)$（向量, 标量）使得

$$y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+\boldsymbol{b}\right)>\boldsymbol{0}$$



硬间隔线性支持向量机——函数间隔

- 函数间隔



- $(\boldsymbol{w},b)$确定的超平面 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$ 对于样本点$\boldsymbol{x}_i$点的函数间隔:

$$\hat{\gamma}_i=y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+\boldsymbol{b}\right)$$

- $(\boldsymbol{w},b)$确定的超平面 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$ 对于样本集合$\{\boldsymbol{x}_i\}$ 的函数间隔:

$$\hat{\gamma}=\min\hat{\gamma}_i$$

硬间隔线性支持向量机——几何间隔

- 几何间隔

当实例$\boldsymbol{x}_i$能正确分类时，

点到超平面的距离为：$y_i\left(\frac{\boldsymbol{w}^{T}\boldsymbol{x}_i+\boldsymbol{b}}{||\boldsymbol{w}||}\right)$

- $(\boldsymbol{w},b)$确定的超平面 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$ 对于样本点$\boldsymbol{x}_i$点的几何间隔：

$$\gamma_i=y_i\left(\frac{\boldsymbol{w}^{T}\boldsymbol{x}_i+\boldsymbol{b}}{||\boldsymbol{w}||}\right)$$

- $(\boldsymbol{w},b)$确定的超平面 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$ 对于样本集$\{\boldsymbol{x}_i\}$的

几何间隔：$\gamma=\min_i\gamma_i$，即 $y_i\left(\frac{\boldsymbol{w}^{T}\boldsymbol{x}_i+\boldsymbol{b}}{||\boldsymbol{w}||}\right)\geq\gamma$

- 函数间隔与几何间隔：$\gamma_i=\frac{\hat{\gamma}_i}{||\boldsymbol{w}||}$ $\gamma=\frac{\hat{\gamma}}{||\boldsymbol{w}||}$

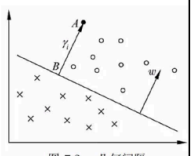


图 7.2 几何间隔

硬间隔线性支持向量机——原始问题

- 间隔的选择
- 函数间隔与几何间隔关系：

$$\gamma_i=\frac{\hat{\gamma}_i}{||\boldsymbol{w}||}\quad\gamma=\frac{\hat{\gamma}}{||\boldsymbol{w}||}$$

函数间隔是一个相对量，但并不直接反映样本点到超平面的实际距离。

当超平面固定后，可以通过对$(\boldsymbol{w},b)$比例伸缩. 并不改变超平面的位置或方向，影响函数间隔的大小。

超平面： $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$

函数间隔： $y\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+\boldsymbol{b}\right)$

对偶问题与支持向量：

硬间隔线性支持向量机——最大间隔分类器

□ 线性可分支持向量的最大间隔法

输入：线性可分的训练数据集$\boldsymbol{D}=\{\boldsymbol{x}_i,y_i\}\quad i=1,2,\ldots,N$

输出：最大间隔分离超平面和分类决策函数。

通过求解如下最优化问题来得到最优分类器的参数$(\boldsymbol{w}^*,b^*)$：

$$\min_{w,b}\frac{1}{2}||\boldsymbol{w}||^2$$

$$s.t.\ y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+\boldsymbol{b})\geq 1,\ i=1,2,..,N$$

最大间隔分离超平面：$(\boldsymbol{w}^*)^{\mathrm{T}}\boldsymbol{x}+b^*=0$

分类决策函数：$f(\boldsymbol{x})=\boldsymbol{sign}((\boldsymbol{w}^*)^{\mathrm{T}}\boldsymbol{x}+b^*)$

定理7.1：对于线性可分的训练数据集，上述优化问题的解存在且唯一

硬间隔线性支持向量机——支持向量

□ 支持向量

在线性可分的条件下，与最优分离超平面距离最近的点称为支持向量，即支持向量是使得约束条件$y_i((\boldsymbol{w}^*)^T\boldsymbol{x}_i+b)\geq 1$等号成立的点：

$$y_i((\boldsymbol{w}^*)^T\boldsymbol{x}_i+b)=1$$

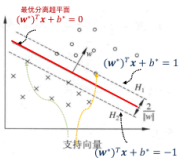
正实例（y_i=1）的支持向量在超平面H_1上

$$(\boldsymbol{w}^*)^T\boldsymbol{x}+b^*=1$$

负实例（y_i=−1）的支持向量超平面 H_2上

$$(\boldsymbol{w}^*)^T\boldsymbol{x}+b^*=-1$$

H_1和 H_2的距离：$\frac{2}{||w||}$



硬间隔线性支持向量机——对偶问题

□ 原问题

1、通过利用拉格朗日对偶性，将SVM的原问题转换为一个对偶问题

2、对偶问题通常更易于求解，在很多情况下，比直接应用通用的QP优化工具箱更高效

3、SVM的对偶方法有效地处理高维数据和非线性分类问题，是监督学习的重要算法

硬间隔线性支持向量机——对偶问题

□对偶问题

➢ 支持向量机原优化问题

$$\min_{w,b}\ \frac{1}{2}||\boldsymbol{w}||^2$$

$$s.t.\ -y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)+1\leq 0,\ i=1,2,..,N$$

用拉格朗日乘子法转换为对偶问题（对偶问题更容易求解）：

定义为拉格朗日函数：$\boldsymbol{L}(\boldsymbol{w},\boldsymbol{b},\boldsymbol{\alpha})=\frac{1}{2}||\boldsymbol{w}||^2+\sum_{i=1}^{N}\boldsymbol{\alpha}_i(-y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)+1)$

$\boldsymbol{\alpha}$为拉格朗日乘子，$\boldsymbol{\alpha}_i\geq\boldsymbol{0}$

由于$L(w,b,\alpha)\leq\frac{1}{2}||w||^2$

所以 $\max_{\alpha}\ L(w,b,\alpha)\leq\frac{1}{2}||w||^2$

01 • 基本概念与三类 SVM

> ✍️ 要求：三类**要记**（老师复习完当场提问过）。

1. SVM 是什么

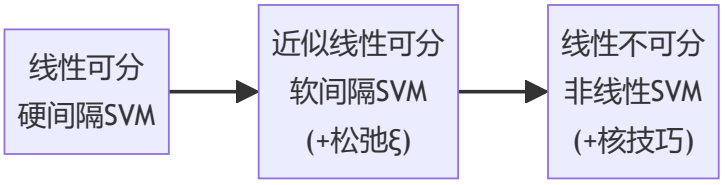
- **支持向量机 (SVM)**：一种**二分类**模型（可通过"一对一/一对多"转化为多分类——本质仍是多个二分类）。
- **核心思想**：通过**最大化类别间的间隔**来寻找**最优分离超平面**，把不同类别的数据分开。
- 学习策略 = **间隔最大化**，可形式化为**凸二次规划**问题，也等价于**正则化的合页损失最小化。**

2. 三类 SVM（按训练数据类型）

数据情况	学习方法	得到的模型	别名
线性可分	硬间隔最大化	线性可分 SVM	硬间隔 SVM

数据情况	学习方法	得到的模型	别名
近似线性可分	软间隔最大化（引入松弛变量 ξ）	线性 SVM	软间隔 SVM
线性不可分	核技巧 + 软间隔最大化	非线性 SVM	（核方法）

> 简单模型是复杂模型的基础与特例：硬间隔 ⊂ 软间隔 ⊂ 非线性。



3. 分离超平面与决策函数

分离超平面：

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$

由法向量 $\boldsymbol{w}$（决定方向）和截距 b 决定。法向量指向的一侧为正类。

分类决策函数：

$$f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w} \cdot \boldsymbol{x} + b)$$

> 数据线性可分时，能正确分开的超平面有**无穷多个**；感知机用"误分类最小"求得的解**不唯一**。 而 SVM 用**间隔最大化**求最优超平面，**解唯一** —— 这是 SVM 区别于感知机的关键。

4. 输入空间 → 特征空间

- 线性（可分/不可分）SVM：假设输入空间与特征空间元素**一一对应**，把输入直接映射为特征向量。
- 非线性 SVM：用一个**非线性映射** ϕ 把输入映到（更高维的）特征空间，再在特征空间里做线性 SVM。
- SVM 的学习都是在**特征空间**进行的。

> 📌 下一步：怎么定义"间隔"、怎么把"间隔最大"写成优化问题 → 02-间隔与硬间隔SVM。

02 • 间隔与硬间隔 SVM

> ✍️ 要求：函数/几何间隔了解；**间隔最大化模型会写、支持向量定义要会。**

1. 函数间隔 vs 几何间隔

对样本 $(\boldsymbol{x}_i, y_i)$、超平面 $(\boldsymbol{w}, b)$：

函数间隔（表示分类的正确性与确信度）：

$$\hat{\gamma}_i = y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)$$

数据集的函数间隔取所有样本的最小值：$\hat{\gamma} = \min_i \hat{\gamma}_i$。

> ⚠️ 坑：函数间隔有个问题——若把 $(\boldsymbol{w}, b)$ 同比例放大为 $(2\boldsymbol{w}, 2b)$，超平面**没变**，函数间隔却翻倍。所以需要"几何间隔"。

几何间隔（对 $\boldsymbol{w}$ 规范化，得到真实的点到超平面距离）：

$$\gamma_i = \frac{y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)}{\|\boldsymbol{w}\|} = \frac{\hat{\gamma}_i}{\|\boldsymbol{w}\|}$$

- 关系：$\gamma = \dfrac{\hat{\gamma}}{\|\boldsymbol{w}\|}$。当 $\|\boldsymbol{w}\| = 1$ 时，函数间隔 = 几何间隔。

2. 硬间隔最大化（线性可分 SVM）

目标：找到几何间隔最大的超平面。

$$\max_{\boldsymbol{w},b} \ \gamma \quad \text{s.t.} \quad \frac{y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)}{\|\boldsymbol{w}\|} \geq \gamma, \ \forall i$$

令函数间隔 $\hat{\gamma} = 1$（可这样取，不影响最优超平面），上式化为标准形式：

$$\boxed{\min_{\boldsymbol{w},b} \ \frac{1}{2}\|\boldsymbol{w}\|^2 \quad \text{s.t.} \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1, \ i = 1, \ldots, N}$$

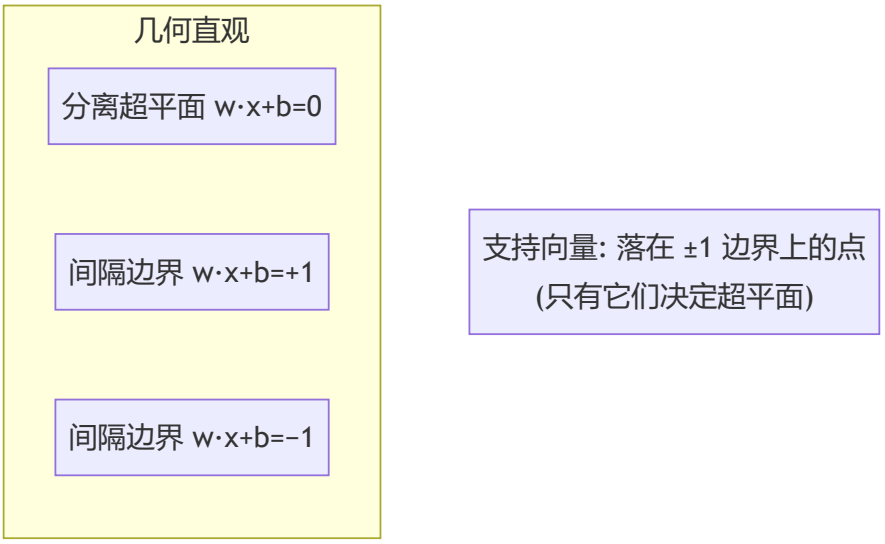
> 🎯 这就是**硬间隔 SVM 的原始问题**（要会写）：最大化间隔 ⇔ 最小化 $\frac{1}{2}\|\boldsymbol{w}\|^2$。 它是一个**凸二次规划**问题，**解存在且唯一。**

3. 支持向量

支持向量 = 离最优分离超平面**最近**的点，即满足

$$y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)=1$$

的样本（函数间隔正好 $=1$，落在边界 $\boldsymbol{w}\cdot\boldsymbol{x}+b=\pm1$ 上）。



> 关键性质：**只有支持向量影响最优超平面**，其余点去掉也不变。这也是"支持向量机"得名的原因。

> 📌 怎么实际求解？通过**拉格朗日对偶**转化（更高效）→ 03-对偶问题与KKT。

03 • 对偶问题与 KKT（核心，要背）

> 🎯 高频考点。老师："数学模型背也得背下来"。这页的式子建议**默写到能复现**。

1. 为什么转对偶

原始问题（硬间隔）：$\min\ \frac{1}{2}\|\boldsymbol{w}\|^2$ s.t. $y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)\geq 1$。 直接解不方便，用**拉格朗日乘子法**转成**对偶问题**：① 求解更高效；② 自然引出**内积** → 方便后面上**核函数**。

2. 构造拉格朗日函数

每个约束配一个乘子 $\alpha_i\geq 0$。先把约束写成 ≤ 0 的形式：$1-y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)\leq 0$。

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) - 1 \right], \quad \alpha_i \geq 0$$

原始问题 $= \min_{\boldsymbol{w},b} \max_{\boldsymbol{\alpha}} L$；对偶问题 $= \max_{\boldsymbol{\alpha}} \min_{\boldsymbol{w},b} L$。

3. 对 $\boldsymbol{w}, b$ 求偏导并代回

令 $\frac{\partial L}{\partial \boldsymbol{w}} = 0$, $\frac{\partial L}{\partial b} = 0$:

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i, \qquad \sum_{i=1}^{N} \alpha_i y_i = 0$$

代回 L，消去 $\boldsymbol{w}, b$，得到**只含 $\boldsymbol{\alpha}$ 和样本内积**的对偶问题。

4. 对偶问题（⭐ 要背）

$$\boxed{\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \qquad \alpha_i \geq 0,\ i = 1, \ldots, N$$

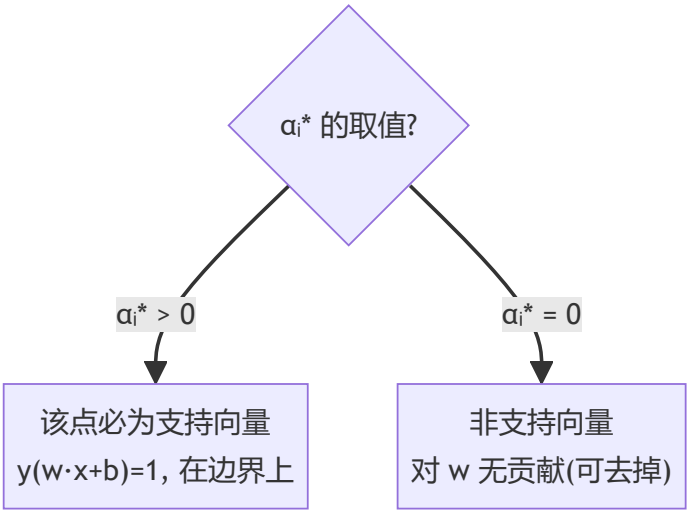
（也常写成等价的最小化形式：$\min_{\boldsymbol{\alpha}} \ \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) - \sum_i \alpha_i$，约束同上。）

5. KKT 条件（⭐ 要背）

设 $(\boldsymbol{w}^*, b^*)$ 是原始解、$\boldsymbol{\alpha}^*$ 是对偶解，则满足 KKT：

$$\begin{aligned}
&\alpha_i^* \geq 0 \\
&y_i(\boldsymbol{w}^* \cdot \boldsymbol{x}_i + b^*) - 1 \geq 0 \\
&\alpha_i^* \left[y_i(\boldsymbol{w}^* \cdot \boldsymbol{x}_i + b^*) - 1 \right] = 0 \quad \text{(complementary slackness)}
\end{aligned}$$

互补松弛的含义（重点理解）：



- $\alpha_i^* > 0 \Rightarrow$ 该点是**支持向量**（在边界 $y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)=1$ 上）。
- 非支持向量的 $\alpha_i^* = 0$，对 $\boldsymbol{w}$ 没有贡献。

6. 求出超平面与决策函数

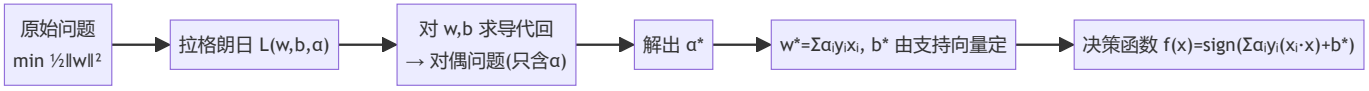
由支持向量解出参数：

$$\boldsymbol{w}^* = \sum_i \alpha_i^* y_i \boldsymbol{x}_i, \qquad b^* = y_j - \sum_i \alpha_i^* y_i (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) \ \ (j:\ \text{any support vector})$$

决策函数（只与支持向量的内积有关，为核函数埋下伏笔）：

$$f(\boldsymbol{x}) = \operatorname{sign}\left(\sum_i \alpha_i^* y_i (\boldsymbol{x}_i \cdot \boldsymbol{x}) + b^*\right)$$

求解流程



例题 7.2 {#例题-72}

> ✍️ 要求：**会算**。给定几个带标签的样本点，求分离超平面与决策函数。

解题套路：

1. 写出对偶问题（代入各样本两两内积 $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$）。
2. 用约束 $\sum_i \alpha_i y_i = 0$ 消元，对目标求极值解出 $\boldsymbol{\alpha}^*$。
3. 由 $\boldsymbol{w}^* = \sum_i \alpha_i^* y_i \boldsymbol{x}_i$ 求 $\boldsymbol{w}^*$；取一个 $\alpha_j^* > 0$ 的支持向量求 b^*。

4. 写出分离超平面 $\boldsymbol{w}^{*} \cdot \boldsymbol{x} + b^{*} = 0$ 与决策函数。

> 🚧 此处需拓展：例 7.2 的具体数字计算请对着课本例题练 1–2 遍（可带计算器）。作业里也有同类题；LSQ 提示"作业 4.2 的数字要重新算、4.3 是右下角那块"。SVM 的**异或问题太难，老师说不用看**。

04 • 软间隔与核函数

> 🎯 考点：软间隔的模型/算法（⚠️ 五角星区别！）、核函数定义与常用核、非线性 SVM 决策函数。

1. 软间隔最大化（线性 SVM）

现实数据常有**噪声/异常点**，不是严格线性可分。给每个样本一个**松弛变量** $\xi_i \geq 0$，允许它"违规"一点：

原始问题：

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i \quad \text{s.t.} \quad y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) \geq 1 - \xi_i, \ \ \xi_i \geq 0$$

- 约束从硬间隔的 ≥ 1 放宽成 $\geq 1 - \xi_i$。
- $C > 0$ 是**惩罚参数**：C 越大越不容忍误分类（趋近硬间隔），越小越宽松。

2. ⚠️ 五角星区别：软间隔对偶 vs 硬间隔对偶

软间隔的**对偶问题**与硬间隔**几乎一模一样**，唯一区别在 α_i 的约束多了上界 C：

$$\max_{\boldsymbol{\alpha}} \ \sum_i \alpha_i - \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) \quad \text{s.t.} \quad \sum_i \alpha_i y_i = 0, \ \ \boxed{0 \leq \alpha_i \leq C}$$

> ⭐ **这就是老师"打五角星"的区别**：
>
> - 硬间隔：$\alpha_i \geq 0$
> - 软间隔：$0 \leq \alpha_i \leq \boldsymbol{C}$（多了上界 C）
>
> 目标函数和等式约束完全相同。记住这一点，两个模型就都拿下了。

软间隔支持向量更丰富：$\alpha_i = 0$ 非支持向量；$0<\alpha_i

3. 合页损失（Hinge Loss）

软间隔 SVM 等价于**正则化的合页损失最小化**：

$$\min_{w,b} \sum_{i=1}^{N} \big[1 - y_i(w \cdot x_i + b)\big]_+ + \lambda \|w\|^2$$

其中 $[z]_+ = \max(0, z)$。即"分类正确且间隔够大就不罚，否则线性惩罚"。

> 🚧 此处需拓展：合页损失曲线与 0-1 损失、逻辑损失的对比，老师列为"了解概念"，需要时看 ppt。

4. 核函数（非线性 SVM）

数据线性不可分时，用**非线性映射 ϕ** 把样本送到高维特征空间，在那里做线性 SVM。

关键观察：对偶问题和决策函数里，样本**只以内积形式 $x_i \cdot x_j$ 出现**。于是定义**核函数**：

$$K(x, z) = \phi(x) \cdot \phi(z)$$

直接用 K 替换内积，**无需显式计算高维 ϕ**——这就是**核技巧 (kernel trick)**。

对偶问题（核化） 与 **决策函数（核化）**：

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j), \quad 0 \le \alpha_i \le C, \ \sum_i \alpha_i y_i = 0$$

$$f(x) = \text{sign}\left(\sum_i \alpha_i^{\backslash *} y_i K(x_i, x) + b^{\backslash *} \right)$$

常用核函数：

核	表达式
线性核	$K(x, z) = x \cdot z$
多项式核	$K(x, z) = (x \cdot z + 1)^p$
高斯核 (RBF)	$K(x, z) = \exp\left(-\frac{\|x - z\|^2}{2\sigma^2} \right)$

> 核方法比 SVM 更一般——核技巧 = 隐式地在高维特征空间学线性模型。

本章一句话回顾

> 间隔（函数/几何）→ 硬间隔原始问题 $\min \frac{1}{2}\|w\|^2$ → 拉格朗日对偶 + KKT → 支持向量 → 软间隔（约束 $0 \le \alpha \le C$，⚠️五角星）→ 核函数（内积换 K）。三类 SVM 的"模型 + 算法"都要会写。

第 24 章 • 卷积神经网络 CNN（🟡 中，计算题重点）

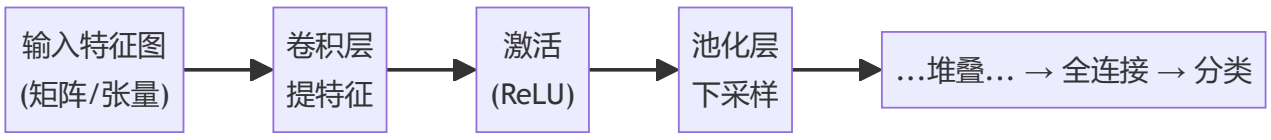
> 🎯 考点：**计算题必出**——二维卷积、填充与步幅、输出矩阵尺寸、三维卷积、池化（最大/平均/求和）。 ✍️ LSQ："CNN 定义的**数学模型公式一定要会写**"；"凡是书上的例题都有，考试会考"（老师原话）。

本章要求

知识点	要求	小节
二维卷积计算	**会算**	01-卷积与池化计算
填充与步幅、输出尺寸	**会算公式**	01-卷积与池化计算
三维卷积	看例题、会算	01-卷积与池化计算
池化（最大/平均/求和）	**会算**	01-卷积与池化计算
CNN 定义（数学模型）	**一定会写**	02-CNN定义与性质
三大性质	会简答	02-CNN定义与性质
反向传播	概念（一次前向+一次反向）	02-CNN定义与性质
图像分类常见模型	能举出名字	02-CNN定义与性质

为什么要 CNN

前馈网络只能吃**向量**，图像是**矩阵**，展平会丢空间结构、参数爆炸。CNN 直接处理矩阵/张量数据， 靠**局部连接 + 参数共享**大幅减少参数。



> 数据类型对应（老师口头强调，常考填空）：**CNN—矩阵数据；前馈—向量数据；RNN—序列数据。**

子页：

1. 卷积与池化计算（含例题）
2. CNN 定义与性质

课件原图参考

> 来自课程 PPT（4 合 1 讲义版）。

二维卷积：

2026/6/16

二维卷积

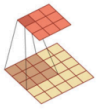
□ 二维卷积定义

给定一个 I×J 输入矩阵 $\boldsymbol{X}=[x_{ij}]_{I\times J}$，一个 M×N 核矩阵 $\boldsymbol{W}=[w_{mn}]_{M\times N}$，满足 $M\ll I, N\ll J$。让核矩阵在输入矩阵上从左到右再从上到下按顺序滑动，在滑动的每一个位置，核矩阵与输入矩阵的一个子矩阵重叠。求核矩阵与每一个子矩阵的内积，产生一个 K×L 输出矩阵 $\boldsymbol{Y}=[y_{kl}]_{K\times L}$，称此运算为卷积（convolution）或二维卷积。写作 $\boldsymbol{Y}=\boldsymbol{W}*\boldsymbol{X}$，其中 $\boldsymbol{Y}=[y_{kl}]_{K\times L}$。

$$y_{kl}=\sum_{m=1}^{M}\sum_{n=1}^{N}w_{m,n}x_{k+m-1,l+n-1}$$

其中 k = 1, 2, …, K; l = 1, 2, …, L; K = I − M + 1; L = J − N + 1。

□ 卷积核又被称为滤波器（filter）。



二维卷积

• 二维卷积-例题：给定输入矩阵 X 和核矩阵 W：

$$\boldsymbol{X}=\begin{bmatrix}3&2&0&1\\0&2&1&2\\2&0&0&3\\2&3&1&2\end{bmatrix},\quad \boldsymbol{W}=\begin{bmatrix}2&1&2\\0&0&3\\0&0&2\end{bmatrix}$$

求卷积 Y = W * X。

二维卷积

• 二维卷积-例题_解：

W 作用在 X 上，并不超出 X 的范围。可得：

$$y_{11}=\sum_{m=1}^{3}\sum_{n=1}^{3}w_{mn}x_{mn}=11$$

$$y_{12}=\sum_{m=1}^{3}\sum_{n=1}^{3}w_{mn}x_{m,n+1}=18$$

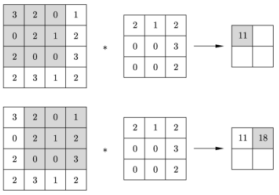
同样可计算 y_{21}，y_{22}，得到输出矩阵 Y：

$$\boldsymbol{Y}=\begin{bmatrix}2&1&2\\0&0&3\\0&0&2\end{bmatrix}*\begin{bmatrix}3&2&0&1\\0&2&1&2\\2&0&0&3\\2&3&1&2\end{bmatrix}=\begin{bmatrix}11&18\\6&22\end{bmatrix}$$

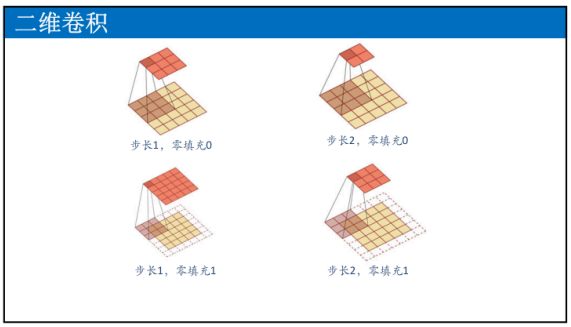
二维卷积

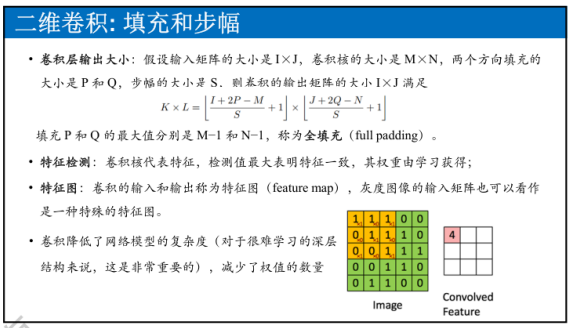
• 二维卷积-例题_解：

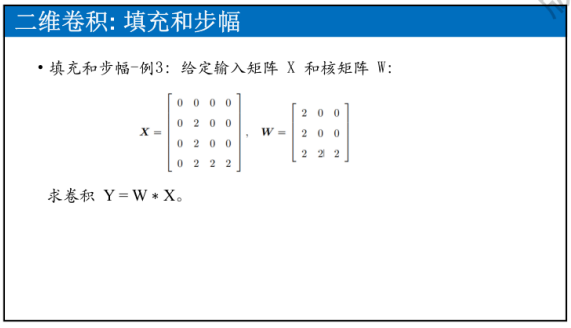
输入矩阵是 4×4 矩阵，核矩阵是 3×3 矩阵，输出矩阵是 2×2 矩阵。

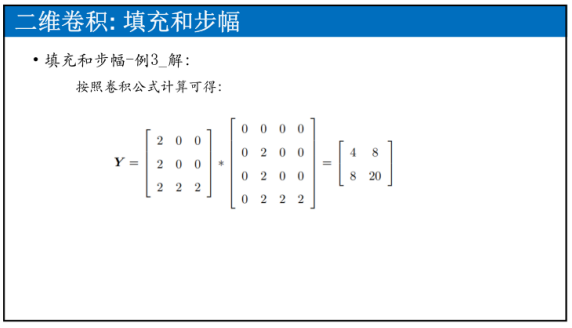


池化与输出尺寸：









01 • 卷积与池化计算（计算题重点）

> 🎯 必出计算题（可带计算器）。把下面的**输出尺寸公式**和**逐元素求和**练熟。

1. 二维卷积（机器学习里其实是"互相关"）

> ⚠️ 概念坑：数学上的卷积要把核翻转；本书/CNN 里说的"卷积"其实是**互相关**（不翻转），二者不满足交换律。 因为核是学出来的，翻不翻转效果一样，所以工程上直接用互相关。

给定输入矩阵 $\boldsymbol{X}=[x_{ij}]_{I\times J}$、卷积核 $\boldsymbol{W}=[w_{mn}]_{M\times N}$（$M\ll I,\ N\ll J$），核在输入上从左到右、从上到下滑动，每个位置做**子矩阵与核的内积**，得到输出 $\boldsymbol{Y}=[y_{kl}]_{K\times L}$：

$$\boldsymbol{Y}=\boldsymbol{W}*\boldsymbol{X},\qquad y_{kl}=\sum_{m=1}^{M}\sum_{n=1}^{N} w_{mn}\,x_{k+m-1,\,l+n-1}$$

基本输出尺寸（无填充、步幅 1）：$K=I-M+1,\quad L=J-N+1$。

2. 例题（务必会算）{#例题}

输入 $\boldsymbol{X}$（4×4）与核 $\boldsymbol{W}$（3×3）：

$$\boldsymbol{X}=\begin{bmatrix}3&2&0&1\\0&2&1&2\\2&0&0&3\\2&3&1&2\end{bmatrix},\qquad \boldsymbol{W}=\begin{bmatrix}2&1&2\\0&0&3\\0&0&2\end{bmatrix}$$

输出尺寸 $K\times L=(4-3+1)\times(4-3+1)=2\times2$。逐位置算（核与对应 3×3 子矩阵逐元素乘后求和）：

- y_{11}：左上 3×3 子块 $\begin{bmatrix}3&2&0\\0&2&1\\2&0&0\end{bmatrix}\to 2\cdot3+1\cdot2+2\cdot0+3\cdot1+2\cdot0=\mathbf{11}$
- y_{12}：右上子块 $\begin{bmatrix}2&0&1\\2&1&2\\0&0&3\end{bmatrix}\to 2\cdot2+1\cdot0+2\cdot1+3\cdot2+2\cdot3=\mathbf{18}$
- y_{21}：左下子块 $\begin{bmatrix}0&2&1\\2&0&0\\2&3&1\end{bmatrix}\to 2\cdot0+1\cdot2+2\cdot1+3\cdot0+2\cdot1=\mathbf{6}$
- y_{22}：右下子块 $\begin{bmatrix}2&1&2\\0&0&3\\3&1&2\end{bmatrix}\to 2\cdot2+1\cdot1+2\cdot2+3\cdot3+2\cdot2=\mathbf{22}$

$$\boldsymbol{Y}=\begin{bmatrix}11&18\\6&22\end{bmatrix}$$

> ⚠️ LSQ 提示：作业 4.2"数字要重新算"，注意题目里给的数据。本书例题**不加偏置** b；若题目要求加 b，每个输出再 $+b$。

3. 填充与步幅、输出尺寸 {#输出尺寸}

- **填充 (Padding)** P,Q：在输入四周补 0，保边缘信息、控制输出大小。
- **步幅 (Stride)** S：核每次滑动的格数，S 越大输出越小。

通用输出尺寸公式（要背、会算）：

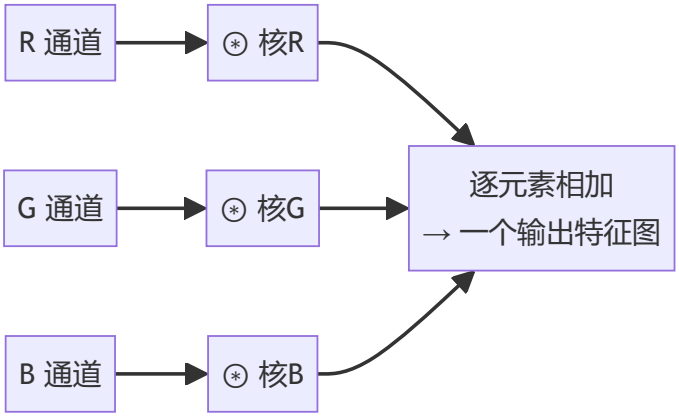
$$K=\left\lfloor\frac{I+2P-M}{S}\right\rfloor+1,\qquad L=\left\lfloor\frac{J+2Q-N}{S}\right\rfloor+1$$

> 例：$I=J=7$、核 3×3、$P=Q=1$、$S=2\to K=\lfloor(7+2-3)/2\rfloor+1=\lfloor3\rfloor+1=4$

> 。输出 4×4。

4. 三维卷积 {#三维卷积}

输入有多个通道（如 RGB 3 通道）时：用 **3 个不同的二维卷积核**分别对 3 个通道做二维卷积， 再把 3 个输出矩阵**对应相加**，得到一个二维输出（一个输出通道）。多个输出通道就用多组这样的核。



5. 池化 / 汇聚 (Pooling) {#池化}

对特征图分块下采样，减小尺寸、增强不变性。三种（会算）：

池化	取值
最大池化	每个窗口取**最大值**
平均池化	每个窗口取**平均值**
求和池化	每个窗口取**和**

> 例：对 $\begin{bmatrix}1 & 3\\ 2 & 4\end{bmatrix}$ 做 2×2 池化：最大 $= 4$，平均 $= 2.5$，求和 $= 10$。 池化窗口大小、步幅同样用上面的输出尺寸公式。

> 📌 输入/输出又称**特征图 (feature map)**。二维卷积的输入输出是矩阵；三维卷积处理多通道张量。

02 • CNN 定义与性质

> ✍️ 要求：**CNN 定义的数学模型公式一定要会写**（老师强调）；三大性质会简答；反向传播懂概念；能举常见模型名。

1. CNN 的定义（数学模型，要会写）

一个卷积层把上一层特征图 $\boldsymbol{X}$ 经"**卷积 + 偏置 + 激活**"得到输出特征图，常再接**池化**：

$$\boldsymbol{Y} = a\big(\boldsymbol{W} * \boldsymbol{X} + b\big)$$

其中 $*$ 是二维（互相关）卷积，$a(\cdot)$ 是激活函数（常用 ReLU）。 CNN = **多个"卷积层(+池化)"堆叠，最后接全连接层做分类。**

2. CNN 三大性质 {#三大性质}

性质	含义
表示效率（参数共享）	同一个卷积核在整张图滑动复用 → 参数量远小于全连接，表示更高效
不变性	对平移（及一定程度的旋转/缩放）具有不变/等变性 → 物体换位置仍能识别
感受野 (Receptive Field)	输出上一个点"看得到"的输入区域；网络越深，感受野越大，能捕捉更大范围特征

> 🎯 这三条常考简答/填空。一句话记：**参数共享省参数、平移不变更鲁棒、感受野随深度变大。**

3. 反向传播（概念）

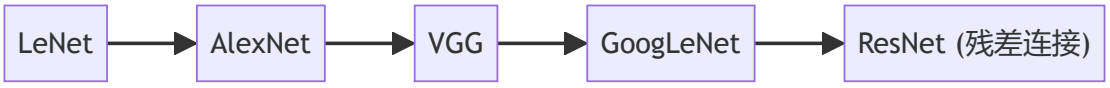
和前馈网络一样：**一次正向传播（算输出/损失）+ 一次反向传播（回传梯度、更新核与偏置）。** 卷积层的反向传播本质仍是链式法则，只是梯度也以"卷积"的形式回传。

> 🚧 此处需拓展：卷积层反向传播的具体推导（对核、对输入的梯度）较繁，老师列为"掌握概念"，会说"完成一次前向 + 一次反向梯度传播"即可。需要时看课本。

4. 图像分类中的常见模型 {#常见模型}

> ✍️ 要求：**能举出名字即可**（填空/简答）。

- **LeNet**：最早的 CNN（手写数字识别）。
- **AlexNet**：深度 CNN 引爆点（ImageNet）。
- **VGG**：用大量 3×3 小卷积堆叠。
- **GoogLeNet / Inception**：多尺度并行卷积。
- **ResNet**：**残差连接**，可训练极深网络、缓解梯度消失（见 前馈/04 正则化技巧）。



本章一句话回顾

> 卷积/池化**会算**（输出尺寸公式 + 逐元素求和）→ CNN 定义 $\boldsymbol{Y} = a(\boldsymbol{W} * \boldsymbol{X} + b)$ **会写** → 三大性质（参数共享/不变性/感受野）→ 反向传播=一次前向+一次反向 → 常见模型能点名。

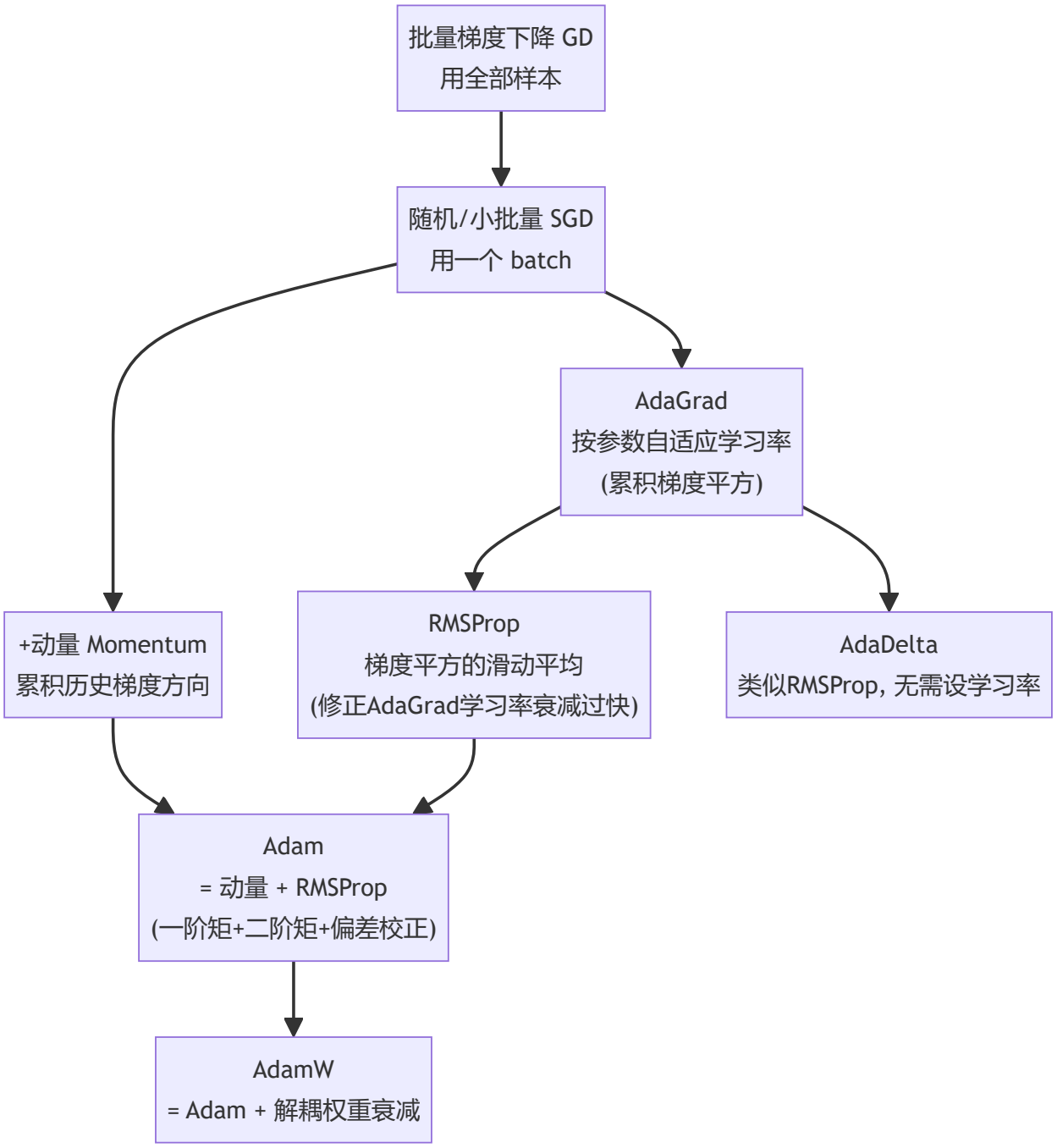
第 29.1 节 · 优化算法（🟠 封面点名必考）

> 🎯 教材**封面手写**明确标注："**优化器必须掌握**"，并列出 7 个。课堂里相关内容主要融在第 23 章讲，所以独立讲解时间短，但**必考**——主要考**更新公式与各优化器区别**（填空/简答）。

要掌握的优化器（封面列出）

类别	优化器
必考	SGD（批量 / 随机）、动量 SGD（Momentum）、RMSProp、Adam
额外补充（旁注"还有"）	AdaDelta、AdaGrad、AdamW

它们的演化关系（一张图理清）



> ✍ 复习要点：能写出每个优化器的**更新公式**，能一句话说清**它解决了前一个的什么问题。**

详见 01-梯度下降与优化器家族。

> 📌 配套：BN、Dropout 等训练技巧见 前馈神经网络/04-正则化与训练技巧。

01 • 梯度下降与优化器家族

> 🎯 必考。建议把每个的**更新公式**和**一句话动机**背下来。记号：$g_t = \nabla_\theta \mathcal{L}$ 为当前梯度，η 学习率，ϵ 防除零小量。

0. 基础：梯度下降

沿负梯度方向更新参数：

$$\theta_{t+1} = \theta_t - \eta\, g_t$$

- **批量梯度下降 (BGD)**：每步用**全部 N 个样本**算梯度——准但慢、内存大。
- **随机梯度下降 (SGD)**：每步用**1 个**（或一个**小批量** mini-batch）样本——快、可在线，但梯度有噪声、震荡。
- 小批量是实际默认：$N = M \times n$，每步用 n 个样本的平均梯度。

> 问题：纯 SGD 在"峡谷"地形震荡、对所有参数用同一学习率、易陷在鞍点 → 引出下面的改进。

1. 动量法 Momentum（解决震荡 / 加速收敛）

引入"速度" v，累积历史梯度（像小球带惯性）：

$$v_t = \beta v_{t-1} + g_t, \qquad \theta_{t+1} = \theta_t - \eta\, v_t$$

（$\beta \approx 0.9$。）一致方向上累积加速、来回方向上相互抵消 → **减少震荡、加快收敛。**

2. AdaGrad（按参数自适应学习率）

累积**梯度平方** r，让更新频繁的参数学习率变小、稀疏参数学习率大：

$$r_t = r_{t-1} + g_t^2, \qquad \theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{r_t} + \epsilon}\, g_t$$

> ⚠️ 缺点：r 单调增 → 学习率**单调衰减、后期过小**，可能提前停止学习。

3. RMSProp（修正 AdaGrad 衰减过快）

把"累加"改成**指数滑动平均**，让 r 不会无限增大：

$$r_t = \beta r_{t-1} + (1-\beta) g_t^2, \qquad \theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{r_t} + \epsilon}\, g_t$$

（$\beta \approx 0.9$。）适合非平稳目标，深度学习常用。

4. AdaDelta（RMSProp 的近亲，几乎不用设学习率）

在 RMSProp 基础上，用**参数更新量的滑动平均**替代全局学习率，**无需手工设 η**。

> 🚧 此处需拓展：AdaDelta 完整公式（含 $\Delta\theta$ 的均方根项）较繁，考"概念/区别"即可——记住"它是不用设学习率的自适应法"。

5. Adam（动量 + RMSProp，最常用）⭐

同时维护梯度的**一阶矩 m**（动量）和**二阶矩 v**（RMSProp），并做**偏差校正**：

$$
\begin{aligned}
m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t && (\text{1st moment, momentum}) \\
v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 && (\text{2nd moment, RMSProp}) \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t} && (\text{bias correction}) \\
\theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t
\end{aligned}
$$

（常用 $\beta_1 = 0.9,\ \beta_2 = 0.999$。）兼具动量的方向平滑与自适应学习率，鲁棒、收敛快，是默认首选。

6. AdamW（Adam + 解耦权重衰减）

Adam 里把 L2 正则塞进梯度会和自适应学习率耦合、效果打折。AdamW 把**权重衰减从梯度里解耦**，单独作用在参数上：

$$
\theta_{t+1} = \theta_t - \eta \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda \theta_t \right)
$$

正则更干净，是现代大模型常用优化器。

一句话对比表（背这个）

优化器	一句话
SGD	沿负梯度走，最基础
Momentum	加惯性，减震荡、加速
AdaGrad	按参数自适应学习率，但后期学习率过小
RMSProp	用梯度平方滑动平均，修正 AdaGrad 衰减

优化器	一句话
AdaDelta	类 RMSProp，几乎不用设学习率
Adam	**动量 + RMSProp + 偏差校正，最常用**
AdamW	Adam + 解耦权重衰减

> 🎯 答题提示：填空常考"哪个引入动量""哪个自适应学习率""Adam = 谁 + 谁"；简答常考"Adam 相比 SGD 的优点""AdaGrad 的缺点及 RMSProp 如何改进"。

第 25 章 • 循环神经网络 RNN（🟢 基础，但 5 种架构要会画）

> 课堂讲得相对少（≈10 分钟），但 LSQ 明确记："**要求掌握的模型与网络架构（要会画！）**"。 老师："它的图和公式是这一章的重点，要能一一对应"。

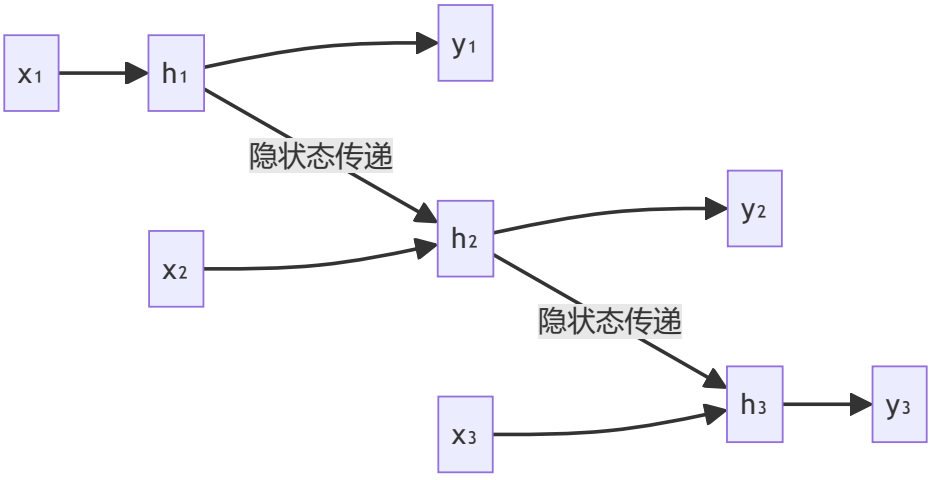
本章要求

知识点	要求
5 种架构（会画图 + 写公式）	SRNN(图25.1)、LSTM(定义25.2/图25.6)、GRU(定义25.3/图25.8)、深度RNN(图25.9)、双向RNN(图25.10)
学习算法	反向传播(BPTT)是**核心算法**，懂概念
梯度消失/爆炸	概念（长序列时严重，LSTM/GRU 缓解）
门控机制	基于门控的有 **LSTM 与 GRU**

> 数据类型对应（常考填空）：**RNN—序列数据**；CNN—矩阵；前馈—向量。

为什么需要 RNN

序列数据（文本、语音、时间序列）有**时间依赖**，前馈/CNN 不擅长。RNN 用**隐状态 h_t 沿时间传递信息**，让"现在"能记住"过去"。



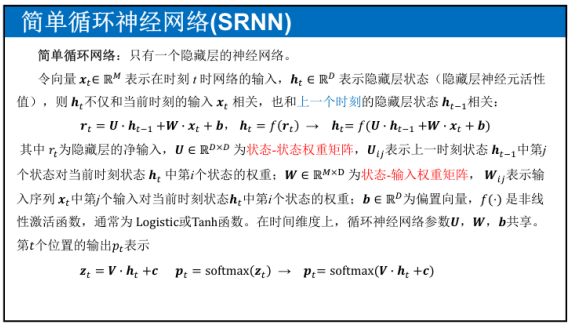
详见 01-五种RNN架构。

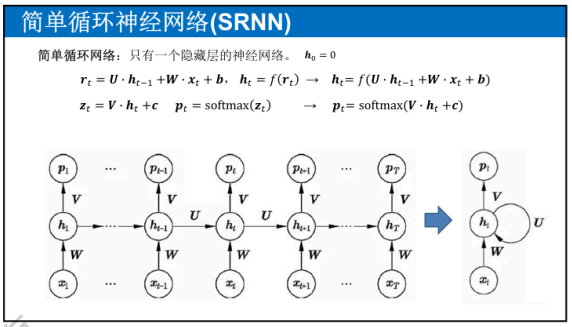
课件原图参考

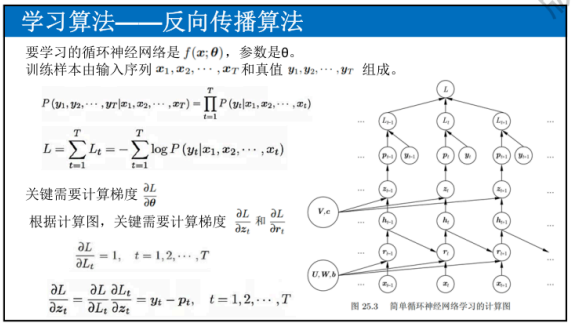
> 来自课程 PPT（4 合 1 讲义版）。

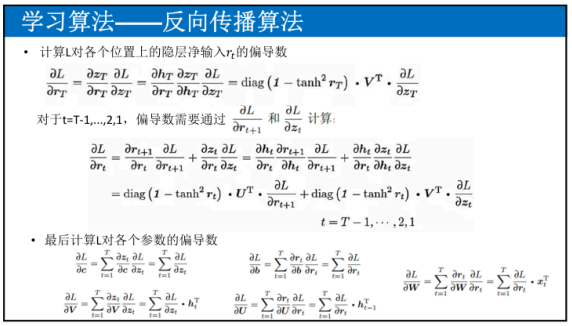
简单循环神经网络 SRNN：

2026/6/16









门控 RNN / LSTM：

基于门控的循环神经网络

- **基于门控的循环神经网络（Gated RNN）**：为了改善循环神经网络长程依赖问题，引入门控机制来控制信息的累积速度，包括有选择地加入新的信息和有选择地遗忘之前的信息，这一类网络可以称为基于门控的循环神经网络。
- **门控机制（Gating Mechanism）**：在数字电路中，门是一个二值变量{0,1}，0代表关闭状态，不允许任何信息通过；1代表开放状态，允许所有信息通过。
- **“软”门**：取值在（0,1）之间，表示按一定的比例允许信息通过

基于门控的循环神经网络——LSTM

- 长距离依存关系会被S-RNN逐渐“遗忘”
- **长短期记忆网络**(LSTM, long short-term memory)基本想法: 记录并使用之前所有位置的状态
  - 记忆元（memory cell）用于记录之前位置的状态信息
  - 门控（gated control）用于控制状态信息的使用，是一个向量，每一维取值在0和1之间
    - 遗忘门（forget gate）
    - 输入门（input gate）
    - 输出门（output gate）
  - 状态
- 循环神经网络上每一个位置上的状态、记忆元、门控构成一个单元（unit）

基于门控的循环神经网络——LSTM

- 第t个位置上的单元以当前位置的输入x_t、之前位置的记忆元c_{t-1}、之前位置的状态h_{t-1}为输入
- 以当前位置的状态h_t和当前位置的记忆元c_t为输出

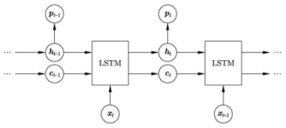


图 25.5 LSTM 的网络架构

- 在每一个位置上有输入x_t、状态h_t、输出p_t，特殊的还有记忆元c_t。状态和记忆元的信息在 LSTM 单元之间传递

基于门控的循环神经网络——LSTM

LSTM引入一个新的内部状态（internal state）$\boldsymbol{c}_t \in \mathbb{R}^D$进行线性的循环信息传递，同时非线性地输出信息给隐藏层的外部状态$\boldsymbol{h}_t \in \mathbb{R}^D$，状态更新过程可由下式表示：

$$\begin{aligned} \boldsymbol{i}_t &= \sigma(\boldsymbol{W}_i \cdot \boldsymbol{x}_t + \boldsymbol{U}_i \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) & \tilde{\boldsymbol{c}}_t &= \tanh(\boldsymbol{W}_c \cdot \boldsymbol{x}_t + \boldsymbol{U}_c \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \\ \boldsymbol{f}_t &= \sigma(\boldsymbol{W}_f \cdot \boldsymbol{x}_t + \boldsymbol{U}_f \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) & \boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \\ \boldsymbol{o}_t &= \sigma(\boldsymbol{W}_o \cdot \boldsymbol{x}_t + \boldsymbol{U}_o \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) & \boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) \end{aligned}$$

其中$\tilde{\boldsymbol{c}}_t \in [0,1]^D$是通过非线性函数得到的候选状态，$\boldsymbol{c}_t$记录了到当前时刻为止的历史信息（记忆单元）；$\boldsymbol{W}_*$为可学习参数，$* \in \{i, f, o, c\}$

$\boldsymbol{f}_t \in [0,1]^D$是遗忘门，控制上一个时刻的内部状态$\boldsymbol{c}_{t-1}$需要保留多少信息到$\boldsymbol{c}_t$

$\boldsymbol{i}_t \in [0,1]^D$是输入门，控制当前时刻的候选状态$\tilde{\boldsymbol{c}}_t$需要向$\boldsymbol{c}_t$输入多少信息，$\boldsymbol{f}_t$和$\boldsymbol{i}_t$互补

$\boldsymbol{o}_t \in [0,1]^D$是输出门，控制当前时刻的内部状态$\boldsymbol{c}_t$需要输出多少信息给外部状态$\boldsymbol{h}_t$

01 • 五种 RNN 架构（会画 + 会写公式）

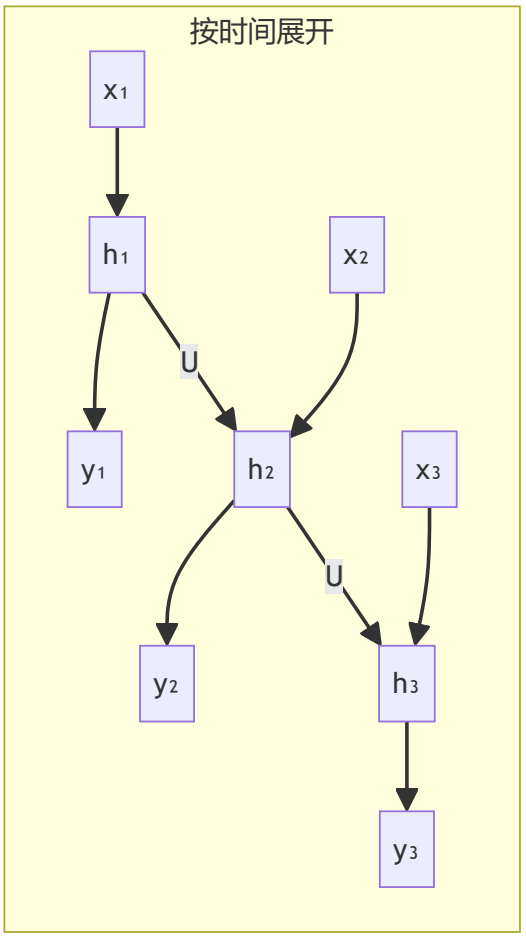
> 🎯 考点：老师"图和公式要能一一对应"。下面每种都给**结构图 + 公式**。考试画图时把**时间展开**和**门**画清楚。

1. 简单循环神经网络 SRNN（图 25.1）

隐状态由"上一时刻隐状态 + 当前输入"决定，再产生输出：

$$\boldsymbol{h}_t = a\big(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b}\big), \qquad \boldsymbol{y}_t = g\big(\boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{c}\big)$$

- $\boldsymbol{U}$：隐→隐（循环权重）；$\boldsymbol{W}$：输入→隐；$\boldsymbol{V}$：隐→输出。**三组权重在所有时间步共享。**



> ⚠️ SRNN 长序列时**梯度消失/爆炸**严重（同样的 $\boldsymbol{U}$ 连乘）→ 难记住远处信息。LSTM/GRU 就是来解决它的。

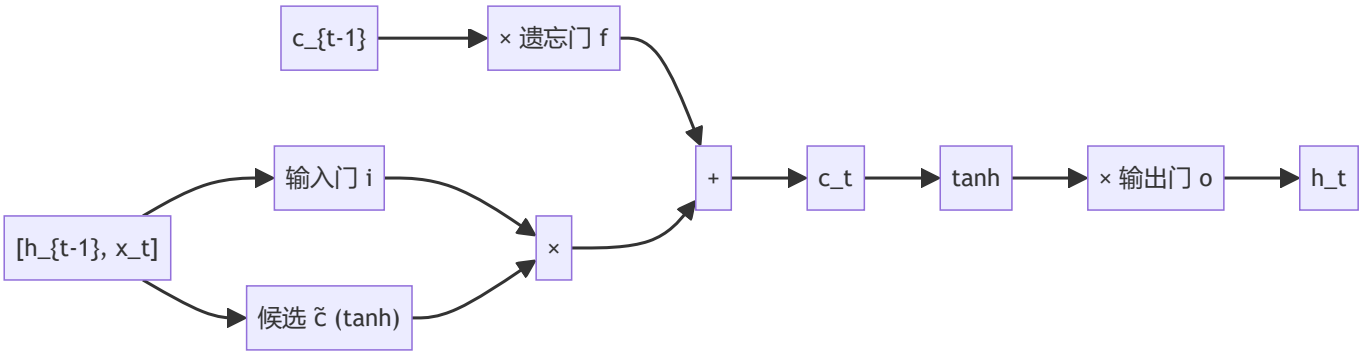
2. 长短期记忆网络 LSTM（定义 25.2 / 图 25.6）

引入**细胞状态** $\boldsymbol{c}_t$（信息高速公路）和**三个门**控制信息流：

门	作用	公式（σ=Sigmoid）
遗忘门 $\boldsymbol{f}_t$	丢弃多少旧记忆	$\sigma(\boldsymbol{W}_f[\boldsymbol{h}_{t-1},\boldsymbol{x}_t]+\boldsymbol{b}_f)$
输入门 $\boldsymbol{i}_t$	写入多少新信息	$\sigma(\boldsymbol{W}_i[\boldsymbol{h}_{t-1},\boldsymbol{x}_t]+\boldsymbol{b}_i)$
输出门 $\boldsymbol{o}_t$	输出多少	$\sigma(\boldsymbol{W}_o[\boldsymbol{h}_{t-1},\boldsymbol{x}_t]+\boldsymbol{b}_o)$

候选记忆与更新：

$$\tilde{\boldsymbol{c}}_t=\tanh(\boldsymbol{W}_c[\boldsymbol{h}_{t-1},\boldsymbol{x}_t]+\boldsymbol{b}_c),\quad \boldsymbol{c}_t=\boldsymbol{f}_t\odot\boldsymbol{c}_{t-1}+\boldsymbol{i}_t\odot\tilde{\boldsymbol{c}}_t,\quad \boldsymbol{h}_t=\boldsymbol{o}_t\odot\tanh(\boldsymbol{c}_t)$$



> 细胞状态 $\boldsymbol{c}_t$ 的加性更新让梯度更容易回传 → 缓解梯度消失，能记住长程依赖。

3. 门控循环单元 GRU（定义 25.3 / 图 25.8）

LSTM 的简化版，**只有两个门**（重置门 $\boldsymbol{r}_t$、更新门 $\boldsymbol{z}_t$），没有单独的细胞状态：

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t]), \quad \boldsymbol{z}_t = \sigma(\boldsymbol{W}_z[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t])$$

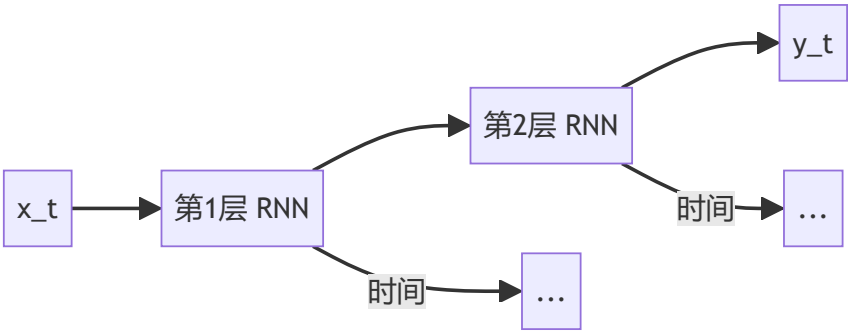
$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}[\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}, \boldsymbol{x}_t]), \quad \boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t$$

- 参数比 LSTM 少、更快，效果常相近。

> 🎯 LSQ 提示（常考填空/简答）：**基于门控机制的 RNN 有 LSTM 与 GRU。**

4. 深度循环神经网络（图 25.9）

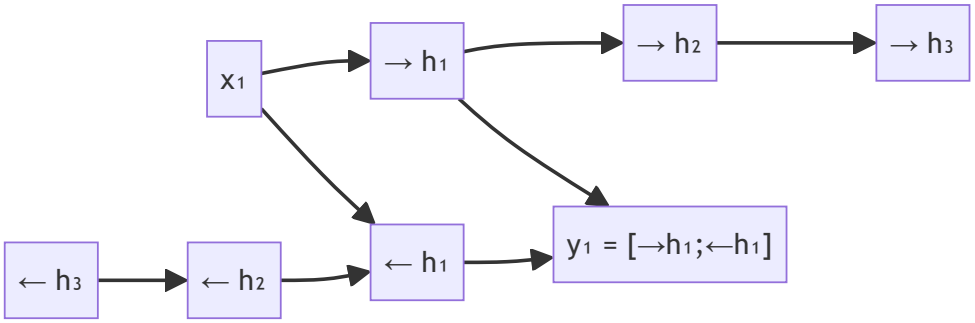
把多个 RNN 层**纵向堆叠**：下层的隐状态序列作为上层的输入序列，增强表达能力。



5. 双向循环神经网络（图 25.10）

同时有**正向**（从前往后）和**反向**（从后往前）两条 RNN，把两者的隐状态拼起来 → 每个时刻都能利用**整句**的上下文。

$$\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}_t}; \overleftarrow{\boldsymbol{h}_t}]$$



学习算法与梯度问题

- **学习算法 = 沿时间的反向传播 (BPTT)**，是 RNN 的**核心算法**（概念即可）。
- 长序列时梯度连乘 → **梯度消失/爆炸**；LSTM/GRU 的门控 + 加性记忆缓解之。

本章一句话回顾

> SRNN（基础，会梯度消失）→ LSTM（三门 + 细胞状态）→ GRU（两门，简化）→ 深度 RNN（纵向堆叠）→ 双向 RNN（正反向拼接）。**5 种都要会画图、会写公式，图与公式一一对应。**

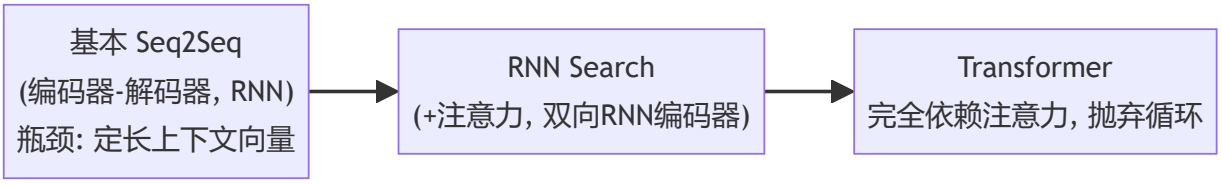
第 26 章 • 序列到序列模型 / Transformer（🟡 中）

> 课堂讲解时间短（≈5 分钟）但**强调密度极高**（注意力/Transformer 反复点到）。LSQ："整体过一遍 ppt"。 老师原话："**这 7 个技术必须要会**"。别因为讲得快就跳过。

本章要求

知识点	要求
基本模型（编码器-解码器）	了解概念
RNN Search 模型	26.2.2 **数学定义**、图 26.5
注意力机制公式	会写
Transformer 七大技术	**必须会**（关注图 26.7）
多头注意力	重点关注
模型特点	老师强调，要会简答

演进脉络（一句话串起来）



详见 01-注意力与Transformer。

📷 课件原图参考

> 来自课程 PPT（4 合 1 讲义版）。

注意力机制：

注意力机制

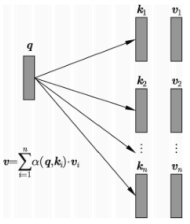
定义 26.1（注意力） 假设有键-值数据库（key-value store），存储键-值对数据 $\{(\boldsymbol{k}_1, \boldsymbol{v}_1), (\boldsymbol{k}_2, \boldsymbol{v}_2), \cdots, (\boldsymbol{k}_n, \boldsymbol{v}_n)\}$，其中每一个键-值对 $(\boldsymbol{k}_i, \boldsymbol{v}_i)$ 的键和值都是实数向量。另有查询（query）$\boldsymbol{q}$，也是实数向量。向量 $\boldsymbol{q}$ 和 $\boldsymbol{k}_i$ 的维度相同，向量 $\boldsymbol{k}_i$ 和 $\boldsymbol{v}_i$ 的维度一般也相同。考虑从键-值数据库中搜索与查询 $\boldsymbol{q}$ 相似的键所对应的值。注意力是实现检索的一种计算方法。计算查询 $\boldsymbol{q}$ 和各个键 $\boldsymbol{k}_i$ 的归一化相似度 $\alpha(\boldsymbol{q}, \boldsymbol{k}_i)$，以归一化相似度为权重，计算各个值 $\boldsymbol{v}_i$ 的加权平均 $\boldsymbol{v}$，将计算结果 $\boldsymbol{v}$ 作为检索结果返回。

$$\boldsymbol{v} = \sum_{i=1}^{n} \alpha(\boldsymbol{q}, \boldsymbol{k}_i) \cdot \boldsymbol{v}_i$$

满足

$$\sum_{i=1}^{n} \alpha(\boldsymbol{q}, \boldsymbol{k}_i) = 1$$

注意力机制



加法注意力使用一层神经网络计算相似度：

$$e(\boldsymbol{q}, \boldsymbol{k}_i) = \sigma\left(\boldsymbol{w}^{\mathrm{T}} \cdot [\boldsymbol{q}; \boldsymbol{k}_i] + b\right) \tag{26.10}$$

其中，σ 是 S 型函数，输入是 $\boldsymbol{q}$ 和 $\boldsymbol{k}_i$ 的拼接，$[;]$ 表示向量的拼接。

乘法注意力使用内积或尺度变换的内积计算相似度：

$$e(\boldsymbol{q}, \boldsymbol{k}_i) = \boldsymbol{q}^{\mathrm{T}} \cdot \boldsymbol{k}_i \tag{26.11}$$

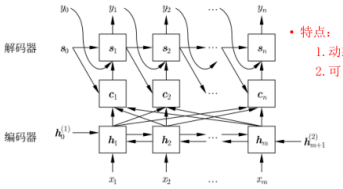
$$e(\boldsymbol{q}, \boldsymbol{k}_i) = \frac{\boldsymbol{q}^{\mathrm{T}} \cdot \boldsymbol{k}_i}{\sqrt{d}} \tag{26.12}$$

其中，d 是向量 $\boldsymbol{q}$ 和 $\boldsymbol{k}_i$ 的维度。尺度变换保证相似度的取值在一定范围内，避免学习时发生梯度消失。

- 注意力将与键相似的值的组合作为检索结果，是一种“软的”而不是“硬的”检索。
- 注意力的模型复杂度，也就是参数个数，不随键-值数据库规模的增大而增大。

RNN Search模型

- RNN Search模型对基本模型进行两个大的改动:
  1.用双向 LSTM实现编码器，
  2.用注意力实现从编码器到解码器的信息传递。



- 特点：
  1. 动态的中间表示（上下文向量）
  2. 可以处理任意长度的输入单词序列

RNN Search模型

编码器使用双向 LSTM。编码基于整个输入序列，是非自回归过程。正向 LSTM 的状态是

$$\boldsymbol{h}_j^{(1)} = a\left(\boldsymbol{x}_j, \boldsymbol{h}_{j-1}^{(1)}\right), \quad j = 1, 2, \cdots, m \tag{26.13}$$

这里 $\boldsymbol{h}_j^{(1)}$ 是正向的当前位置的状态；$\boldsymbol{h}_{j-1}^{(1)}$ 是前一个位置的状态；$\boldsymbol{x}_j$ 是当前位置的输入单词的词向量；a 是处理单元，如 LSTM 单元；假设 $\boldsymbol{h}_0^{(1)} = \boldsymbol{0}$。反向 LSTM 的状态是

$$\boldsymbol{h}_j^{(2)} = a\left(\boldsymbol{x}_j, \boldsymbol{h}_{j+1}^{(2)}\right), \quad j = m, m-1, \cdots, 1 \tag{26.14}$$

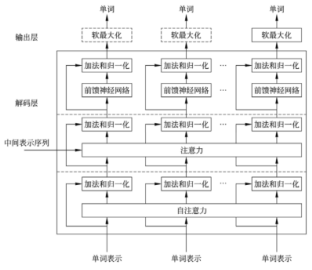
这里 $\boldsymbol{h}_j^{(2)}$ 是反向的当前位置的状态；$\boldsymbol{h}_{j+1}^{(2)}$ 是前一个位置的状态；$\boldsymbol{x}_j$ 是当前位置的输入单词的词向量；a 是处理单元，如 LSTM 单元；假设 $\boldsymbol{h}_{m+1}^{(2)} = \boldsymbol{0}$。在各个位置对正向和反向状态进行拼接，得到各个位置的状态，也就是中间表示。

$$\boldsymbol{h}_j = [\boldsymbol{h}_j^{(1)}; \boldsymbol{h}_j^{(2)}], \quad j = 1, 2, \cdots, m \tag{26.15}$$

这里 $[;]$ 表示向量的拼接。

Transformer 架构（含图 26.7）：

Transformer模型



Transformer模型: 嵌入表示

✓ 词嵌入（Embedding）

token index 输入到可学习的嵌入矩阵E 中，获得固定维度的向量(通常维度为 512)

示例（假设维度为 4）

Token	Embedding 向量
the	[0.1, 0.3, 0.2, 0.4]
book	[0.5, 0.6, 0.1, 0.2]
is	[0.2, 0.9, 0.3, 0.3]
on	[0.7, 0.4, 0.5, 0.1]
table	[0.9, 0.8, 0.6, 0.5]

这些嵌入向量为 Transformer 模型的初始输入表示序列。将整个句子的嵌入结果按行拼接为一个矩阵/张量表示，每一行是一个 token 的向量

Transformer模型: 位置编码

✓ 位置编码（Positional Encoding）

因为 Transformer 完全基于注意力机制，不像 RNN 有内部“顺序性”，需要人为引入位置信息，所以需为每个位置添加一个 位置向量（可以是可学习的，也可以是固定的 sin-cos 编码）。为了得到不同位置所对应的编码，Transformer 结构使用不同频率的正余弦函数：

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

其中pos表示单词在序列中的位置，2i与2i+1代表奇数位置和偶数位置，d_{model}表示编码总维度。序列中每个单词的位置都会生成一个位置向量，并与其词向量相加，再送入后续、网络层。训练过程中，模型逐渐学会利用位置信息。

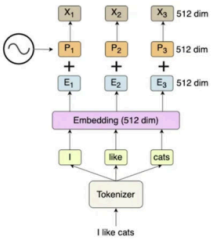
✓ 好处：

正、余弦函数取值范围 [−1, +1]，与原始词嵌入相加后不会显著偏离，从而保留了单词的语义信息。

由于三角函数的线性关系，第 pos + k 个位置的编码可以由第 pos 个位置的编码线性表示，这意味着位置编码中自然包含了单词之间的距离信息。

Transformer模型: 位置编码

✓ 位置编码（Positional Encoding）



✓ pos：位置（第几个 token，从 0 开始）

✓ i：维度索引（第几维）

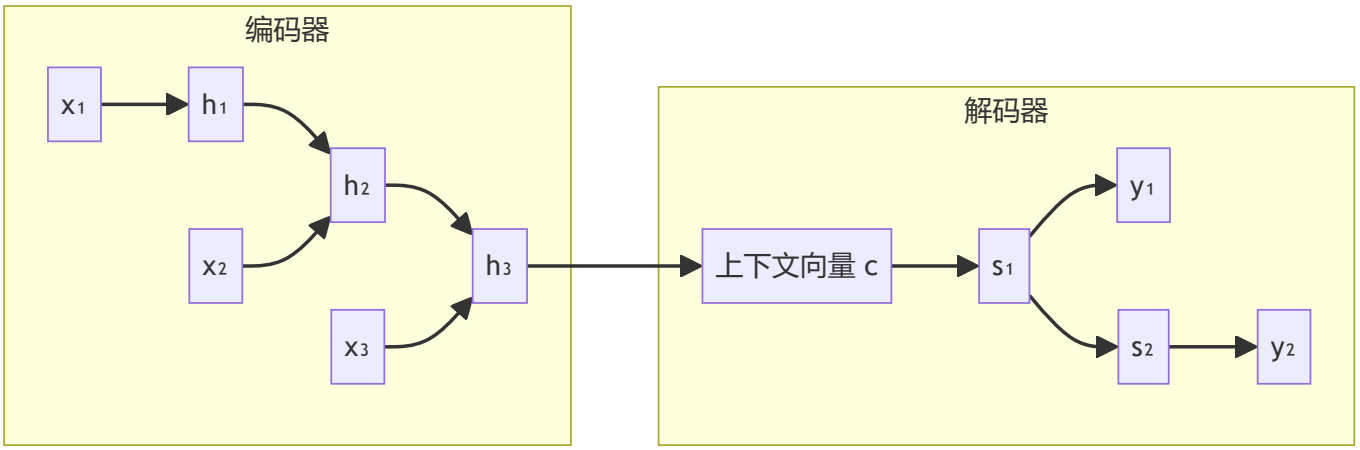
✓ d_{model}：向量总维度，比如 512

01 • 注意力与 Transformer

> 🎯 考点：注意力公式、RNN Search、Transformer 七大技术、多头注意力、模型特点。

1. 基本序列到序列（编码器-解码器）

把一个序列映射成另一个序列（翻译、摘要等）。**编码器**（RNN）把输入序列压成一个**上下文向量 c**，**解码器**（RNN）从 c 逐步生成输出序列。



> ⚠️ 瓶颈：把整句压进**一个定长向量 c**，长句会丢信息 → 引出**注意力**。

2. 注意力机制（公式要会写）

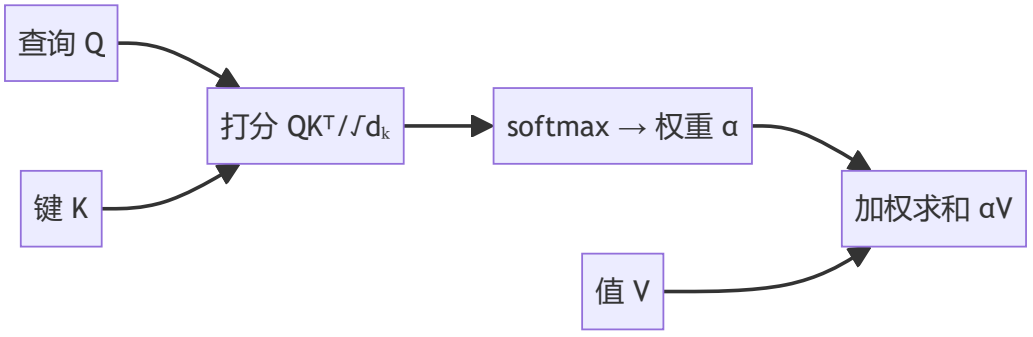
思想：解码每一步，**动态地从编码器所有隐状态里"按需取信息"**，而不是只用一个 c。 注意力 = 用一个**查询 (query)** 去一组**键值对 (key-value)** 里加权检索：

$$\alpha_i = \mathrm{softmax}_i\big(\mathrm{score}(\boldsymbol{q}, \boldsymbol{k}_i)\big) = \frac{\exp(\mathrm{score}(\boldsymbol{q}, \boldsymbol{k}_i))}{\sum_j \exp(\mathrm{score}(\boldsymbol{q}, \boldsymbol{k}_j))}, \qquad \mathrm{Attention}(\boldsymbol{q}, \{\boldsymbol{k}_i, \boldsymbol{v}_i\}) = \sum_i \alpha_i \boldsymbol{v}_i$$

缩放点积注意力（Transformer 用，矩阵形式）：

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

（除以 $\sqrt{d_k}$ 防止点积过大导致 softmax 梯度过小。）



3. RNN Search 模型（26.2.2）

= **带注意力的 seq2seq**：编码器用**双向 RNN**得到每个位置的隐状态作为 key/value； 解码每步用当前解码状态作 query，对编码器隐状态做注意力，得到**随步变化的上下文向量 c_t**。

> ✍️ 要求：26.2.2 的**数学定义**要会；**图 26.5**（LSQ 打了 ✓，可能要求会画——以 ppt 要求为准）。

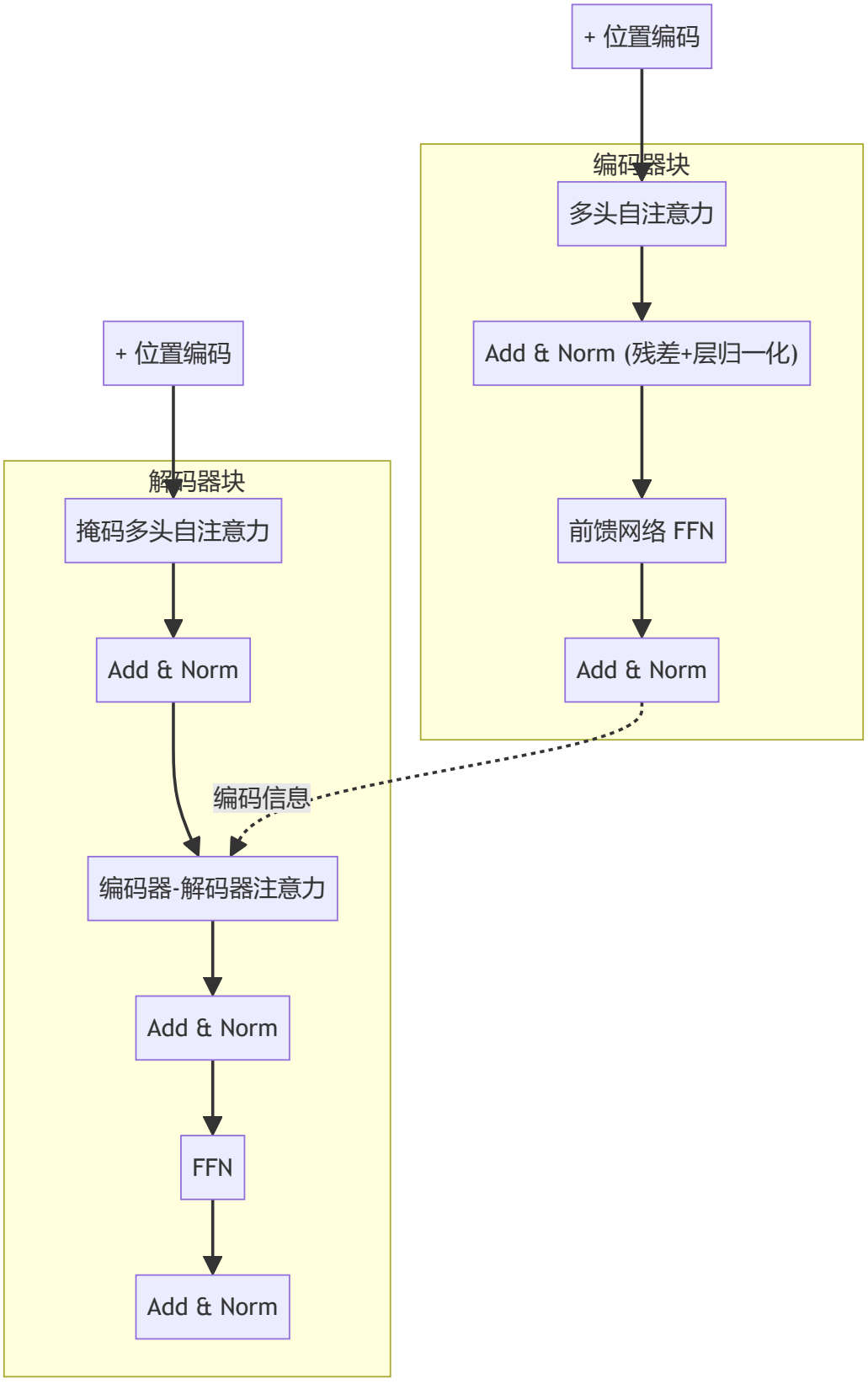
4. Transformer（图 26.7）

完全依赖注意力、**抛弃循环结构**的编码器-解码器。因为没有循环，可**并行**、更擅长**长程依赖**。

七大技术（老师："这 7 个必须会"）

> 🚧 此处需拓展：以下为按课堂录音 + 标准 Transformer 重建的 7 项，**请对照 ppt 图 26.7 的官方表述核对措辞。**

1. **自注意力 (Self-Attention)**：序列内部每个位置对其它所有位置做注意力。
2. **多头注意力 (Multi-Head Attention)**：并行多组注意力（不同子空间），再拼接 → 捕捉多种关系。
3. **编码器-解码器注意力 (Cross-Attention)**：解码器用 query 去查编码器输出（信息传递）。
4. **掩码自注意力 (Masked Self-Attention)**：解码器中遮住未来位置，保证自回归（只能看已生成的）。
5. **位置编码 (Positional Encoding)**：Transformer 不含位置信息，需额外注入（常用正弦/余弦编码）。
6. **残差连接 (Residual Connection)**：每个子层加残差，防止信息/梯度丢失（老师："没有残差连接很容易使位置信息丢失"）。
7. **层归一化 (Layer Norm) + 前馈网络 (FFN)**：每子层后接 LN；每块含一个逐位置前馈网络。



多头注意力（重点关注）

把 $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$ 线性投影成 h 组，各自做注意力，再拼接、再投影：

$$\mathrm{MultiHead} = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_h)\boldsymbol{W}^O, \quad \mathrm{head}_i = \mathrm{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V)$$

> 老师："多头本质上就是注意力的并列，表示成矩阵的形式。"

位置编码

因为没有循环/卷积，模型本身不知道词的先后，需要把**位置信息**加到输入嵌入上（常用不同频率的正弦/余弦函数）。

模型特点（会简答）

- 完全基于注意力，**无循环、可高度并行**（训练快）。
- 擅长捕捉**长程依赖**。
- 靠**位置编码**补充顺序信息，靠**残差 + 层归一化**稳定训练。

本章一句话回顾

> 基本 seq2seq（编码器-解码器，定长上下文有瓶颈）→ 注意力（query 对 key-value 加权检索，公式要会写）→ RNN Search（双向 RNN + 注意力）→ Transformer（完全靠注意力，**七大技术必须会**，重点多头注意力 + 图 26.7）。

第 7 部分 · 练习题（带详解）

> 🎯 取自本课**作业原题**，聚焦考试主力题型（**计算题**），答案均经程序核算。 解答放在折叠块里（HTML 里可点开、PDF 里默认展开），建议**先自己做再看**。

目录

> 题型对应（见 00-复习重点地图）：计算题占大头，且**可带计算器**——把下面这些算熟，计算题基本稳了。

01 · 计算题精练（带详解）

> 答案均经程序核算。先做再看解答。

题 1 · 前馈网络：前向 + 反向传播（作业二•5）

两层前馈网络：输入 $\boldsymbol{x} \in \mathbb{R}^2$ →第一层线性变换→ ReLU →第二层线性变换→ 输出 $\hat{\boldsymbol{y}}$。

> ⚠️ **这道题是本课最容易丢分的约定坑，务必看完下面三部分。 本课程/课本的线性变换统一用 $\boldsymbol{z} = \boldsymbol{W}^{\mathsf{T}}\boldsymbol{x} + \boldsymbol{b}$（即 $\boldsymbol{w}$ 与 $\boldsymbol{x}$ 做内积）**，输出层同理 $\hat{\boldsymbol{y}} = \boldsymbol{W}_2^{\mathsf{T}}\boldsymbol{h} + \boldsymbol{b}_2$。 依据：课本原文"每一层的输入到净输入的仿射变换实际根据 $\boldsymbol{z} = \boldsymbol{W}^{\mathsf{T}}\boldsymbol{x}$ 进行"。 **偏置**：本题虽标"不含偏置"（数值上 $\boldsymbol{b} = \boldsymbol{0}$），但**公式里必须把 $+\boldsymbol{b}$ 写出来**（老师强调，$\boldsymbol{z}$ 与 $\hat{\boldsymbol{y}}$ 两式都要写，不写扣分）；结构图上画不画都行。

原题（照抄作业）： 输入 $\boldsymbol{x}=[1,2]^{\mathsf{T}}$；第一层 $\boldsymbol{W}_1=\begin{bmatrix}1 & 2\\ -1 & 0\end{bmatrix}^{\mathsf{T}}$；第二层 $\boldsymbol{W}_2=[1,2]^{\mathsf{T}}$；激活 ReLU；$y=1$；$L=\frac{1}{2}(\hat{y}-y)^2$；不含偏置。

(1) 求前向 $\hat{y}$ 与 L；**(2)** 求 $\partial L/\partial \boldsymbol{W}_1,\ \partial L/\partial \boldsymbol{W}_2$。 老师给的标准答案是 $\boxed{\hat{y}=5,\ L=8}$。

▶ 📝 解答（三部分：原题为何对不上 / 怎么改 / 标准算法）

> ⚠️ **两个易错点**：① 把 $\boldsymbol{W}_1$ 的 $^{\mathsf{T}}$ 先转置成 $\begin{bmatrix}1 & -1\\ 2 & 0\end{bmatrix}$，又用 $\boldsymbol{z}=\boldsymbol{W}\boldsymbol{x}$ → 乘错矩阵 → 得 $\hat{y}=4$。 ② 用 $\boldsymbol{z}=\boldsymbol{W}\boldsymbol{x}$ 代替本课的 $\boldsymbol{z}=\boldsymbol{W}^{\mathsf{T}}\boldsymbol{x}$，或公式里漏写偏置 $+\boldsymbol{b}$。 牢记本课约定：$\boldsymbol{z}=\boldsymbol{W}^{\mathsf{T}}\boldsymbol{x}+\boldsymbol{b}$。

题 2 • CNN：二维卷积（作业四•补充1）

输入图像 I 与卷积核 W（偏置 $b=0$）：

$$I=\begin{bmatrix}0&0&1&1&0\\0&1&0&0&1\\0&0&1&0&1\\1&1&0&1&0\\0&0&0&0&0\end{bmatrix},\qquad W=\begin{bmatrix}0&1&0\\1&0&0\\0&0&0\end{bmatrix}$$

分别求：(1) 步幅 $S=1$、填充 $P=0$；(2) $S=2,P=0$；(3) $S=1,P=1$；(4) $S=2,P=1$ 的输出特征图及尺寸。

> 尺寸公式：$K=\left\lfloor\dfrac{M+2P-W}{S}\right\rfloor+1$。该核只挑两个位置：$y=I[k,l{+}1]+I[k{+}1,l]$。

▶ 📝 解答

题 3 • CNN：池化（作业四•补充2）

特征图 $F=\begin{bmatrix}2&4&6&1\\6&4&3&2\\0&2&2&1\\7&3&0&1\end{bmatrix}$，用 2×2、步幅 2 的池化。求平均池化与最大池化结果。

▶ 📝 解答

题 4 • SVM：核函数 + 对偶 + 决策（异或例）

> ⚠️ 老师说 SVM 异或问题"太难、不一定考"；但它是**核 SVM 计算的标准范例**，把流程吃透，例题 7.2 同理可解。

4 个样本（异或型，线性不可分）：

$$x_1=[-1,-1], y_1=-1;\quad x_2=[-1,1], y_2=1;\quad x_3=[1,-1], y_3=1;\quad x_4=[1,1], y_4=-1$$

取多项式核 $K(x,z)=(1+x^{\mathsf{T}}z)^2$（对应映射 $\phi(x)=[1,x_1^2,\sqrt{2}x_1x_2,x_2^2,\sqrt{2}x_1,\sqrt{2}x_2]^{\mathsf{T}}$）。 求：(1) 核矩阵 K；(2) 对偶问题；(3) 决策函数。

▶ 📝 解答

> 🔗 相关原理：SVM/03-对偶问题与KKT、SVM/04-软间隔与核函数。

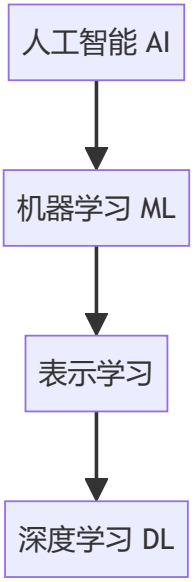
02 • 概念题速答（简答 / 填空）

> 取自各章作业的概念题，按"简答能答上、填空不丢分"整理。

---

1. 人工智能/机器学习/表示学习/深度学习的隶属关系（作业一•1）

四者是**层层包含**关系（韦恩图：一个套一个）：



深度学习 ⊆ 表示学习 ⊆ 机器学习 ⊆ 人工智能

- **人工智能**：让机器模拟/延伸人的智能。
- **机器学习**：从有限观测数据学出一般规律，用于预测。
- **表示学习**：能自动学出有效特征的机器学习方法。
- **深度学习**：用有深度的模型自动学习**多层次特征表示**（底层→中层→高层），也称深度神经网络。

> ⚠️ 画韦恩图：四个**同心椭圆**从外到内 AI⊃ML⊃表示学习⊃DL。

---

2. 三种激活函数及表达式（作业二•2）

名称	表达式	导数
Sigmoid	$\sigma(z)=\dfrac{1}{1+e^{-z}}$	$\sigma(1-\sigma)$
tanh	$\dfrac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	$1-\tanh^{2} z$
ReLU	$\max(0,z)$	$\mathbb{1}[z>0]$

> 详见 前馈/02-激活函数。

---

3. 神经元（感知机）模型的局限（作业二•3）

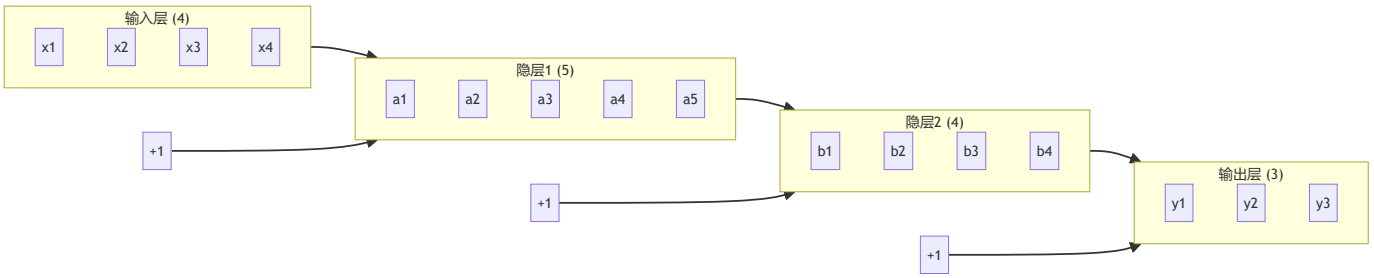
单个神经元/感知机是**线性分类器**，**不能解决线性不可分问题**，典型如**异或 (XOR)**。 需要引入**隐层**（多层网络）才能解决。详见 前馈/01 例:异或。

---

4. 多层前馈网络结构图（作业二•1 / 三•4）

> ✍️ 画图要点：**标注符号**、**层间全连接**、**偏置 +1 可画可不画**（图上随意，但数学模型公式里 $z=W^{\mathsf{T}}x+b$ 的 $+b$ 必写）。中间多个隐层可用省略号；注意是**全连接**。

例：输入 $\boldsymbol{x}=(x_1,x_2,x_3,x_4)$、隐层1 为 5 个神经元、隐层2 为 4 个神经元、输出 $\boldsymbol{y}=(y_1,y_2,y_3)$。



（实际作答时把每个结点和权重 $w_{ji}^{(t)}$、偏置 $b_j^{(t)}$ 标全。）

---

5. CNN 的三个主要构成层及数学表达（作业四•补充3）

1. **卷积层**：$\boldsymbol{Y = W * X + b}$（局部连接、参数共享）。
2. **激活层**：逐元素非线性 $a(\cdot)$，常用 ReLU。
3. **池化层**：对邻域取最大/平均，下采样、增强不变性。

---

6. 卷积 / 池化输出尺寸公式（作业四•补充4）

输入 $M \times N$、核 $W \times W$、填充 P、步幅 S：

$$M_1 = \left\lfloor \frac{M + 2P - W}{S} \right\rfloor + 1, \qquad M_2 = \left\lfloor \frac{N + 2P - W}{S} \right\rfloor + 1$$

池化通常不填充（$P = 0$）：$M_1 = \left\lfloor \frac{M - W}{S} \right\rfloor + 1$，同理 M_2。

---

7. RNN 相关（作业五）

- **5 个循环神经网络的例子**：简单循环网络 SRNN、LSTM、GRU、深度循环网络、双向循环网络。
- **2 个基于门控机制的循环网络**：**LSTM** 与 **GRU**。
- 三类网络对应的数据：**前馈→向量；CNN→矩阵；RNN→序列。**

> 详见 RNN/01-五种RNN架构。