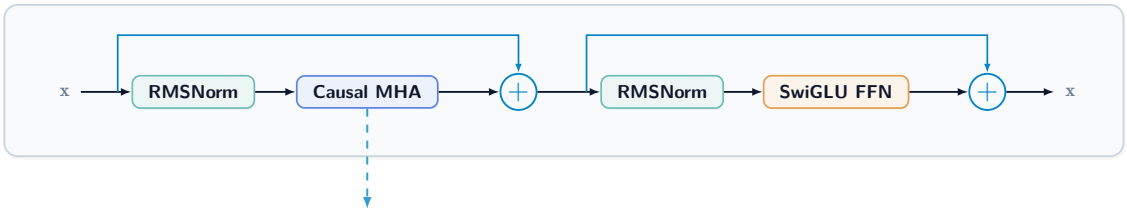
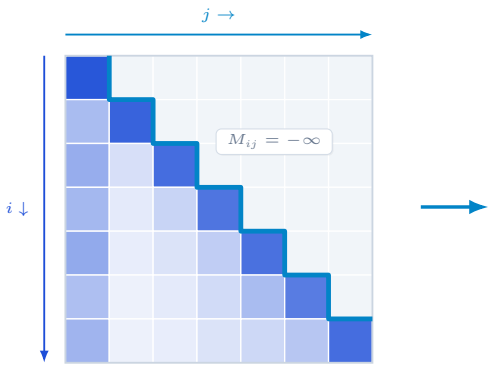
大语言模型

从理论到实践





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

causal_attention.py

```python
def causal_attention(q, k, v, mask=None):
  scores = (q @ k.transpose(-2, -1))
  scores = scores / math.sqrt(d_k)

  if mask is not None:
    scores = scores.masked_fill(
      mask == 0, -1e9)

  weights = torch.softmax(scores, dim=-1)
  return weights @ v
```

韩威　著

第 1 版

前言

大语言模型的学习常常从调用一个接口开始. 接口能够迅速呈现模型的能力，却也把数据表示、网络计算、训练目标和推理执行隐藏在同一个入口之后. 当输出偏离预期、资源需求超出预算，或实验难以复现时，仅仅熟悉调用方式已经不足以解释问题. 本书希望建立一条能够向内追溯机制、向外连接系统的学习路径.

全书分为五篇. 语言模型理论基础建立数学、表示、网络计算与生成的共同语言，并给出评估和安全的方法. 应用智能体系统从上下文学习、检索和 RAG 进入领域应用，再展开智能体架构与 Harness 执行系统. 模型训练方法连接优化、数据、监督微调、预训练、人类反馈学习，以及可验证推理训练. 训练推理系统讨论资源预算、算力基础设施、分布式训练、缓存、调度、服务和模型优化. 模型架构拓展提供双向编码、混合专家、长上下文及视觉理解与生成专题. 读者沿所选方向研究信息流、学习目标和执行成本，并用明确证据检验设计判断.

正文通过连续的解释连接问题、公式与工程含义，以明确的假设、符号和中间步骤建立论证. 数值例题用于展示计算过程，系统案例用于分析条件和取舍；各类证据都应说明能够支持和不能支持的结论.

从零实现是理解机制的一种方法. 它要求关键计算能够被阅读、追踪和验证，但不要求重新制造所有基础设施. 核心语义清楚之后，还需要学习成熟库的接口、默认值与制品契约，辨别哪些行为在迁移中保持不变，哪些数值和性能特征依赖具体实现.

本书面向能够阅读 Python，并具备基础数学与神经网络知识的读者. 绪论说明全书范围与学习路线，基础章节恢复后续推导所需的共同语言. 对于尚不熟悉的先修内容，补充学习比跳过概念更有助于形成稳定理解.

正文首次出现专业术语时给出中文名称、英文全称和常用缩写；书前符号约定与章内符号表说明数学记号，书末中文术语索引与英文术语索引支持查阅. 公式推导标明假设，构造算例与实测结果分别说明. 脚注用于补充概念、历史或口径差异，参考文献提供原始依据，交叉引用帮助读者沿问题回查相关机制.

本书采用持续修订的方式组织内容. 数学机制与基本方法尽量保持清晰稳定；涉及具体模型、软件和运行条件的内容，则通过来源、版本与实验条件说明适用范围. 任何性能数字和效果判断都应回到其证据，不能脱离任务与环境单独理解.

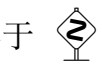
希望这本书能够帮助读者形成一种工作方式：提出具体问题，建立可解释的模型，通过实践验证关键性质，再将结论放回真实系统的约束中审视. 理解、实现与验证相

互支撑，才能把对大语言模型的熟悉逐步转化为独立判断和持续实践的能力.

阅读说明

第 1 版

本书用外侧页边的危险弯道标记标示可暂时跳过的扩展内容. 未标记的正文属于所在章节的学习主线，包括核心概念、基本机制、必要推导与基础算例；推导较长并不意味着可以省略.



带有该标记的内容包括可独立研读的扩展证明、专题分析、方法比较或综合问题. 首次阅读可以暂时跳过，掌握相关主线知识后再回读；从事对应专题时，这些内容仍可能十分重要.

节首外侧页边的图标适用于该节正文及其下属小节. 例题按自身内容标记，图标不影响其推导条件与结论的完整性.

图标表示学习路径上的可延后性，并非简单的难易排名. 扩展内容应能够从主线中独立移开，后续主线论述不得依赖其中尚未交代的定义或结论. 危险弯道图标采用单色形式，黑白打印不影响辨认.

建议先完成共同基础，再沿所选方向阅读相关章节，结合问题回读带图标的部分. 遇到不熟悉的符号可查书前符号约定，术语可查书末中文术语索引或英文术语索引.

全书的知识组织

全书分为五篇. 第一篇“语言模型理论基础”从数学与神经网络语言进入文本表示、注意力、Transformer 整体结构、位置表示和自回归生成，篇末给出评估与安全方法. 初读评估与安全时，先掌握评估对象、独立测试集、失败切片、威胁模型和工具授权，专题分析随任务深入.

第二篇“应用智能体系统”依次讨论上下文学习、信息检索、RAG、领域应用、智能体架构与 Harness 运行系统. 第三篇“模型训练方法”从优化、数据和训练稳定性进入监督微调、参数高效适配和工程优化，再讨论预训练、强化学习、人类反馈学习、可验证推理训练.

第四篇“训练推理系统”先建立资源预算和算力基础，再分别展开分布式训练与推理服务，后者沿缓存、调度和服务架构进入算子与模型优化. 第五篇“模型架构拓展”讨论双向编码、混合专家、长上下文、开放权重模型以及视觉理解与生成；相关专题可作为具体任务的前置阅读.

共同基础及方向选择

阅读路线由共同基础和三条方向组成：应用智能体系统、模型训练方法、训练推理系统．第二至第四篇分别展开三条方向，第五篇按模型与任务提供专题．读者可以选择一个方向深入，也可以在补齐前置知识后组合多个方向．

共同基础以理解一次模型请求为目标．先从第一章建立模型、训练和系统的职责，再从第二章、第三章和第十七章掌握条件概率、交叉熵的含义、张量形状、梯度更新与数据划分．随后依次选读第四章的分词与嵌入，第五章的查询、键、值与因果掩码，第六章的整体结构与前向数据流，以及第七章的顺序信息．第八章补齐下一词元分布、采样、终止及事实性边界．

共同基础还包括第九章中的评估对象、独立测试集和失败切片，以及第十章中的威胁模型与工具授权．完成这一范围后，应能沿词元、张量、条件分布和生成决策解释一次请求，说明参数更新与运行时上下文各自的作用，并提出可检验的效果指标．完整反向传播、优化推导和系统性能模型随专项学习展开；进入某项推导前，应补齐其使用的定义与假设．

应用智能体系统

入门沿第十一章、第十二章、第十三章和第十四章展开，学习上下文组织、检索、证据组装和业务验证．学习终点是能将任务拆成输入、证据、模型输出与评价方法，并区分检索失败、生成失败和业务约束失败．

深入智能体时，依次阅读第十五章和第十六章．其中 Harness 指模型之外的执行与控制系统，负责工具调用、状态持久化、权限、执行隔离和失败恢复．该专题以理解任务执行和终止语义为目标；模型计算、批处理和服务容量由系统方向承接．长上下文和多模态内容按应用需要选读．

模型训练方法

专项前置是完整的概率、自动微分、优化与注意力反向传播，以及 Transformer 和自回归训练目标．入门依次阅读第十八章、第二十七章、第十九章、第二十章、第二十一章和第二十二章，建立数据、监督信号、可训练参数、资源与恢复之间的关系．学习终点是能设计适配方案，并用独立测试集检验目标能力、遗忘与失败切片．

深入内容按训练目标分支．预训练方向阅读第二十三章，研究数据配比与计算预算；对齐方向依次阅读第二十四章、第二十五章和第二十六章，研究人类反馈、奖励建模、偏好优化与可验证奖励．需要扩大训练规模时，再衔接系统方向的分布式训练．

训练推理系统

先阅读第二十七章和第二十八章，建立工作负载需求与硬件供给的关系．此后按目标选择两个分支：推理服务分支沿第三十章、第三十一章和第三十二章学习缓存、调度和请求生命周期；训练系统分支在掌握训练目标、自动微分与第十九章后，进入第二十九章．

深入时选择第三十三章的算子与解码优化，或第三十四章的模型优化；蒸馏与量化感知训练须补齐训练前置．学习终点是能建立容量与性能模型，区分估算和实测，解释并行、调度及恢复的取舍，并设计质量约束下的测量方案．

模型架构拓展

数据、评估与安全贯穿三条方向：应用关注知识来源、任务效果与工具权限，模型训练关注采样、泛化与训练资产，系统关注吞吐、延迟与服务隔离．详细方法在对应章节展开，并在所选方向中结合具体对象使用．

第三十五章、第三十六章、第三十七章和第三十八章按模型与任务选择．视觉理解沿第三十九章进入第四十章；视觉生成沿第四十一章进入第四十二章，并补齐概率与视觉架构前置．涉及具体模型时，先掌握其使用的机制，再分析配置与技术报告．

路线中的主题选读表示当前学习目标所需的范围，章内危险弯道表示相对于该章主线可延后的内容．两者分别服务于方向选择和章内阅读；深入一个主题时，仍需完整掌握其依赖的概念与推导．

内容体系

正文解释定义、假设、推导和系统关系，例题用明确的数据与中间量展示计算．数学推导、构造算例与实测结果具有不同的证据能力，应分别说明其前提与适用范围．

例如，公式推导可以证明某种归一化具有行和为一的性质；小矩阵例题可以逐项展示其计算过程；程序对照可以检查特定实现是否符合该定义．三项全部成立，也仍不能单独证明模型在某个领域具有足够的答案正确率．理论性质、实现正确性与任务效果必须分别建立证据．

本书按照概念依赖组织章节，核心定义与必要推导均在正文展开．涉及实验性结论时，应明确对象、数据、训练预算与测量条件，避免把局部观察推广为普遍规律．

本书的“从零”指从基本表示和运算理解机制，并据此形成独立实现与判断的能力．它不要求重写驱动、编译器或所有数值库．完成原理验证后，使用成熟实现仍需核对其默认值、数值约定和边界条件．[1]

[1]同一个数学算子可以有不同的存储布局、融合策略和求和顺序；实数算术中的等价不必表现为浮点结果逐位相同．

符号及数学约定

除非局部另行声明，本书以大写字母表示矩阵，小写字母表示向量、标量或索引. 形状中的轴顺序具有语义，交换轴必须同时改变后续运算. 同一字母在不同学科中可能有不同含义，章内符号表与紧邻公式的定义优先；局部复用不代表对象相同. 批量计算通常采用行向量约定；低秩适配等局部推导使用列向量时会明确说明，不能只按矩阵名称判断是否需要转置.

符号	约定含义
B, T, S	批量大小、目标或当前序列长度、源序列长度；有效长度可以因样本而异.
d, d_k, d_v	模型隐藏维数、单头查询与键维数、单头值维数.
H, H_q, H_{kv}	注意力头数、查询头数、键值头数；视觉章节中的 H, W 局部表示图像高宽.
L, d_f	网络层数、前馈网络中间维数；训练损失通常写为 $\mathcal{L}$，业务损失等局部记号另行定义.
$V, \mathcal{V}$	词表大小、词表集合；注意力中 V 表示值矩阵时，词表大小另加下标.
θ, η	模型参数、学习率；采样温度通常用 τ.
$x_{<t}, x_{\leqslant t}$	位置 t 之前、不晚于位置 t 的序列前缀.
$\mathbb{R}^{m\times n}$	由 m 行 n 列实数矩阵构成的空间.
$X^{\mathsf{T}}, \odot$	转置、逐元素乘法；普通并置表示矩阵乘法或标量乘法.
$\mathbf{1}[\cdot]$	条件成立时取一、否则取零的指示函数；$\mathbf{1}$ 也可表示同维全一向量，依上下文区分.
$\mathbb{E}, \mathrm{Var}$	期望与方差；所有统计结论均依赖所声明的采样分布.
$\|x\|_2, \|W\|_F$	向量欧氏范数、矩阵 Frobenius 范数；矩阵 $\|W\|_2$ 表示谱范数.
$\nabla_\theta, \mathrm{d}x$	对参数的梯度、积分或微分中的微分记号.
$\pi(a \mid s), V^\pi, Q^\pi$	强化学习中的策略、状态价值与动作价值；与注意力矩阵 Q, V 分属不同语境.
γ, λ	强化学习中常表示折扣与迹衰减；排队中的 λ 表示到达率，依章内定义.
$\beta_t, \alpha_t, \bar{\alpha}_t$	离散扩散中的噪声方差、单步信号保留量与累计保留量.
t, τ	时间、位置或采样温度的局部记号；离散步与连续路径时间不能直接互换.
$q_\phi(z \mid x), p_\theta(x)$	变分近似后验与生成分布；下标指出可学习参数.

符号	约定含义
$\widehat{x}, \widehat{p}$	估计、预测或有限样本统计量，具体性质由构造方式决定.
$\varnothing$	空集合或协议定义的空条件；空条件并不总是零向量.
$\mathrm{stopgrad}(x)$	前向保留数值、反向阻断该路径梯度的运算.

概率质量用于离散取值，概率密度用于连续变量. 离散分布满足 $\sum_x p(x) = 1$；密度满足 $\int f(x)\,\mathrm{d}x = 1$，单点密度值不是单点概率. Logit 指归一化之前的实数分数，Softmax 指沿指定轴进行的归一化指数变换.

本书使用自然对数. Kullback–Leibler 散度（KL 散度）定义为

$$D_{\mathrm{KL}}(p\|q) = \sum_x p(x) \log \frac{p(x)}{q(x)}.$$

约定 $p(x) = 0$ 的项为零；若某处 $p(x) > 0$ 而 $q(x) = 0$，散度为正无穷. 该量一般不对称，也不满足距离所需的全部性质. 连续分布使用相应密度积分.

公式中的 $O(\cdot)$ 描述在给定假设下的渐近增长阶，不等于实际运行时间. 资源公式明确区分参数数量、元素数量、字节数与时间；GB 采用十进制，GiB 采用二进制. 示例中的代数计算不作为设备实测结果.

形状说明默认省略广播前后的隐式复制，不代表内存中实际复制. 概率公式中的求和或积分范围以局部定义为准；条件分布需要说明条件事件和支持集. 边界值可能导致零分母、退化分布或未定义指标，不能仅为得到一个数而代入一般情形公式.

目录

前言 i

阅读说明 iii

符号与数学约定 vi

第一部分　语言模型理论基础 1

第一章　绪论 2

1.1 语言建模的研究对象 2
1.2 语言模型的发展历程 3
1.3 规模定律及涌现能力 6
1.4 概率预测、学习目标及生成决策 7
1.5 模型、算法及系统的分工 8
1.6 知识体系及学习路径 10

第二章　数学基础 12

2.1 向量、矩阵及线性变换 12
2.2 导数、梯度及链式法则 13
2.3 概率基础 13
2.4 随机变量及分布 14
2.5 联合概率、条件概率及链式分解 15
2.6 期望、方差及经验分布 16
2.7 最大似然及负对数损失 17
2.8 熵、交叉熵及 KL 散度 19
2.9 序列损失 21

第三章　神经网络基础 24

3.1 前馈神经网络 24
3.2 训练、梯度下降及反向传播 25

3.3 张量、形状及线性映射 . . . 25
3.4 矩阵梯度 . . . 27
3.5 计算图及自动微分 . . . 28
3.6 分类损失的导数及数值稳定性 . . . 30
3.7 广播、归约及嵌入查表 . . . 35
3.8 有限差分及梯度核验 . . . 36

第四章 文本表示 38
4.1 模型输入构造 . . . 38
4.2 神经语言模型 . . . 40
4.3 神经序列模型 . . . 40
4.4 文本表示的层次 . . . 41
4.5 字节对编码 . . . 43
4.6 WordPiece 分词模型 . . . 45
4.7 Unigram 分词模型 . . . 46
4.8 可逆性、偏移及语言覆盖 . . . 50
4.9 词表规模、序列长度及计算代价 . . . 51
4.10 特殊词元及序列协议 . . . 52
4.11 填充、因果可见性及损失选择 . . . 53
4.12 分词器及模型制品的一致性 . . . 54
4.13 嵌入表示 . . . 56
4.14 输入输出权重共享及语义几何 . . . 57

第五章 注意力机制 60
5.1 查询键值机制 . . . 60
5.2 缩放点积及归一化 . . . 62
5.3 掩码规定可见性 . . . 64
5.4 因果单头注意力 . . . 66
5.5 多头注意力及输出投影 . . . 68
5.6 交叉注意力及结构关系 . . . 69
5.7 注意力的反向传播 . . . 70
5.8 计算规模及键值共享 . . . 72
5.9 注意力结构性质 . . . 73

第六章 Transformer 网络结构 76
6.1 网络层的职责及张量约定 . . . 76
6.2 前馈网络、激活及门控 . . . 78
6.3 残差连接及梯度传播 . . . 80

6.4 LayerNorm 及 RMSNorm 81
6.5 Pre-LN 及 Post-LN 的结构差异 82
6.6 三类模块结构及四种可见性 84
6.7 张量路径及参数核算 87
6.8 训练及推理接口 90

第七章 位置表示 93
7.1 位置表示的必要性 93
7.2 绝对位置及正弦编码 94
7.3 旋转位置编码的构造 96
7.4 RoPE 梯度 99
7.5 位置方案的边界及一致性 100

第八章 自回归语言建模 102
8.1 用条件分布定义序列 102
8.2 下一词元分布 103
8.3 教师强制及移位监督 104
8.4 真实前缀及生成前缀 106
8.5 幻觉、事实性及证据支持 107
8.6 采样分布变换 109
8.7 贪心及束搜索 111
8.8 终止、约束及重复退化 113
8.9 条件生成及缓存的边界 115

第九章 模型评估 117
9.1 评估对象 117
9.2 常用数据集及基准协议 118
9.3 分类、概率及决策代价 124
9.4 生成指标的测量对象 125
9.5 评审员测量模型 127
9.6 区间估计 128
9.7 污染、切片及选择偏差 130
9.8 在线实验及多目标决策 131

第十章 模型安全 133
10.1 安全问题的对象及范围 133
10.2 风险度量边界 135
10.3 提示注入及指令边界 136

10.4 工具授权及执行事务 137
10.5 数据及模型供应链 139
10.6 隐私及多租户隔离 140
10.7 模型提取及未经授权的蒸馏 142
10.8 思维链保护及思考签名 143
10.9 安全评估及发布控制 145
10.10 监测及事件响应 146

第二部分　应用智能体系统　149

第十一章　上下文学习　150
11.1 固定模型上的条件学习 150
11.2 上下文样本配置 151
11.3 分步生成的概率结构 153
11.4 自一致性及多数投票 154
11.5 候选验证及覆盖率 156
11.6 预算及自适应停止 157
11.7 实验迁移性 159

第十二章　信息检索　160
12.1 检索对象及语料结构 160
12.2 倒排索引及概率排序 161
12.3 双编码器及对比学习 163
12.4 向量相似度及精确搜索 164
12.5 近似索引的两条路线 165
12.6 混合检索及学习排序 166
12.7 权限、版本及增量更新 167
12.8 检索评价指标 169

第十三章　检索增强生成　171
13.1 检索增强的概率含义 171
13.2 召回、重排及训练关系 173
13.3 证据预算及选择 175
13.4 查询变换及检索控制 176
13.5 引用、冲突及拒答 179
13.6 知识服务的数据及执行契约 180
13.7 授权边界及缓存一致性 182
13.8 分层评估及反事实分析 183

第十四章　领域应用 **186**
14.1 任务定义及方法选择 186
14.2 领域知识及数据契约 188
14.3 知识及文档密集型应用 188
14.4 业务流程及工业应用 189
14.5 专业服务及研发应用 190
14.6 业务效果的计算及归因 190
14.7 失败归因及改进 192

第十五章　智能体架构 **193**
15.1 部分可观测的决策闭环 193
15.2 Harness 执行控制体系 195
15.3 反应、规划及工作流 196
15.4 状态机及有限终止 197
15.5 信息价值计算 199
15.6 记忆的生命周期 200
15.7 单智能体及多智能体协作 201
15.8 结构化动作及失败恢复 203
15.9 架构评估及证据边界 205

第十六章　智能体运行系统 **206**
16.1 任务判定契约 206
16.2 运行架构及数据模型 207
16.3 持久执行及状态转换 208
16.4 外部动作、幂等及未知结果 209
16.5 预算、异步及取消 210
16.6 工具接入及人工协作 211
16.7 编码 Harness 及个人智能体平台 211
16.8 Computer Use 的观察及行动闭环 213
16.9 浏览器控制协议及适配 214
16.10 知识助手及工单处理的链路 215
16.11 测试、观测及发布 218

第三部分　模型训练方法 **219**

第十七章　优化及泛化 **220**
17.1 训练目标及统计目标 220
17.2 梯度下降及曲率 221

17.3 随机梯度及批量大小 224
17.4 初始化及信号传播 227
17.5 正则化及其作用机制 228
17.6 偏差、方差及泛化 230
17.7 数据隔离及分布漂移 232
17.8 优化失败及过拟合诊断 233

第十八章 训练数据工程基础 **235**
18.1 统计对象及数据分类 236
18.2 数据版本构建 237
18.3 质量、隐私及评估污染 238
18.4 精确去重及近似去重 239
18.5 划分及混合定义训练分布 242
18.6 词元物化、打包及存储 243
18.7 流式执行、发布及删除 245
18.8 数据管线设计 247

第十九章 训练稳定性 **248**
19.1 动量优化 248
19.2 解耦权重衰减 250
19.3 学习率日程及训练时钟 251
19.4 有限精度、损失缩放及梯度裁剪 251
19.5 训练状态闭合性 252
19.6 训练算法及诊断 254

第二十章 监督微调 **256**
20.1 监督微调的学习对象 256
20.2 回答掩码及移位的精确定义 258
20.3 样本权重及长度归一化 260
20.4 多轮对话及模板协议 261
20.5 指令数据的来源及质量 262
20.6 训练批次及参数更新的接口 264
20.7 评估及能力保持 265
20.8 训练契约及后续方法 267

第二十一章 参数高效微调 **268**
21.1 受约束的模型适配 268
21.2 LoRA 的低秩更新 269

21.3 反向传播、初始化及缩放 270
21.4 目标模块及更新预算 272
21.5 合并、组合及数值语义 273
21.6 QLoRA 的量化及梯度边界 274
21.7 其他参数高效方法 275
21.8 训练及制品生命周期 278
21.9 微调框架的职责及组合 280

第二十二章 训练工程优化 **281**
22.1 显存预算及可训练集合 281
22.2 有效监督词元及梯度累积 282
22.3 序列长度及数据装箱 282
22.4 混合精度及激活重计算 282
22.5 分布式微调方案的选用 283
22.6 微调执行评估 283

第二十三章 预训练原理 **285**
23.1 预训练所估计的分布 285
23.2 有效词元上的目标及归一化 286
23.3 序列装箱的条件分布 288
23.4 数据混合及长尾覆盖 289
23.5 不同预训练任务的监督结构 290
23.6 参数、数据及计算的统一预算 291
23.7 经验规模定律的拟合 292
23.8 计算最优分配的推导 294
23.9 规模结论的外推边界 295
23.10 预训练的数据流及状态流 296

第二十四章 强化学习基础 **298**
24.1 决策过程及语言模型 298
24.2 回报及 Bellman 关系 300
24.3 两步决策过程 301
24.4 策略梯度推导 302
24.5 基线、优势及方差 303
24.6 蒙特卡洛、时序差分及广义优势估计 304
24.7 Actor–Critic 及策略分布 305
24.8 重要性采样及 PPO 306
24.9 语言模型的奖励及长度目标 307

24.10 采样、估计及更新的数据流 308
24.11 奖励及验证边界 309

第二十五章 人类反馈学习 311
25.1 人类反馈及偏好数据 311
25.2 奖励建模 312
25.3 基于人类反馈的强化学习 313
25.4 直接偏好优化 314
25.5 在线 RLHF 的模型分工 316
25.6 GRPO 及方法选择 318
25.7 奖励偏差及评估边界 319

第二十六章 可验证推理训练 320
26.1 可验证任务及监督对象 320
26.2 拒绝采样及推理数据 322
26.3 组内相对优势 323
26.4 概率比、裁剪及 KL 325
26.5 长度、探索及奖励投机 328
26.6 测试时计算及采样指标 330
26.7 工具任务中的轨迹及信用分配 333
26.8 数据污染及独立评估 333

第四部分 训练推理系统 335

第二十七章 模型规模及资源预算 336
27.1 计量对象及单位 336
27.2 参数量核算 337
27.3 矩阵计算量及训练近似 338
27.4 推理显存及并发容量 339
27.5 训练状态、激活及分片 340
27.6 带宽、时间及设备数 341
27.7 主机、存储及数据供给预算 344
27.8 敏感性及预算表达 344

第二十八章 算力基础设施 345
28.1 系统边界及计量单位 345
28.2 加速器计算模型 346
28.3 Roofline 性能模型 348

28.4 主机及设备亲和性 350
28.5 互连及集群通信 352
28.6 存储系统及数据通路 354
28.7 训练及推理集群的组织 357
28.8 资源调度、兼容性及隔离 358
28.9 可靠性、供电及散热 360
28.10 系统性能诊断 362

第二十九章 分布式训练 364
29.1 不变的目标及可变的执行方式 364
29.2 梯度累积及全局归一化 365
29.3 混合精度及激活重计算 366
29.4 集合通信及反向计算的依赖 367
29.5 ZeRO 及 FSDP 状态分片 368
29.6 张量并行的代数结构 369
29.7 流水并行及调度 370
29.8 序列、上下文及专家并行 371
29.9 混合并行及故障恢复 372
29.10 扩展效率及效果判断 373
29.11 分布式训练框架的工程落点 374

第三十章 增量解码 375
30.1 预填充及增量解码 375
30.2 键值缓存等价性 376
30.3 复杂度及带宽 378
30.4 MHA、GQA 及 MQA 的缓存 378
30.5 低秩键值及潜变量缓存 380
30.6 逻辑位置及物理分页 381
30.7 前缀共享及缓存生命周期 382
30.8 滑窗、淘汰及位置编码约束 384
30.9 缓存身份及隔离 384
30.10 增量执行过程 385
30.11 缓存计量及服务边界 386

第三十一章 批处理调度 387
31.1 请求、时间及容量的计量边界 387
31.2 连续批处理 388
31.3 分块预填充及多资源预算 389

31.4 排队关系的推导及适用条件 . . . 390
31.5 准入、期限及背压 . . . 392
31.6 公平性及可恢复的抢占 . . . 393
31.7 尾延迟及有效容量 . . . 394

第三十二章 模型服务架构 395
32.1 制品身份及运行配置 . . . 395
32.2 服务拓扑及组件分工 . . . 396
32.3 请求状态及流式协议 . . . 397
32.4 工作进程的生命周期 . . . 400
32.5 预填充及解码分离 . . . 401
32.6 扩缩容及启动提前量 . . . 402
32.7 模型路由及服务降级 . . . 403
32.8 发布及回滚 . . . 406
32.9 遥测及关联 . . . 407
32.10 故障、背压及重试放大 . . . 408
32.11 推理服务基准测试 . . . 409
32.12 推理引擎体系 . . . 409

第三十三章 推理计算优化 411
33.1 计算、搬运及启动成本 . . . 411
33.2 在线 Softmax 的充分统计量 . . . 412
33.3 FlashAttention 及存储层次 . . . 413
33.4 算子融合及执行图 . . . 414
33.5 推测解码的精确性 . . . 415
33.6 多词元验证及缓存事务 . . . 416
33.7 推测收益及正确性边界 . . . 418

第三十四章 模型优化 419
34.1 压缩对象及误差预算 . . . 419
34.2 量化 . . . 420
34.3 剪枝 . . . 431
34.4 低秩分解 . . . 435
34.5 知识蒸馏 . . . 437
34.6 组合压缩及误差传播 . . . 442
34.7 真实硬件收益及统一比较 . . . 444

第五部分　模型架构拓展 **445**

第三十五章　双向语言建模 **446**
35.1 双向编码输入 . . . 446
35.2 损坏通道及掩码语言建模 . . . 448
35.3 词元预测损失 . . . 450
35.4 句对目标及共享主干 . . . 451
35.5 分类、标注及抽取任务 . . . 452
35.6 双向目标同自回归目标的差异 . . . 453

第三十六章　混合专家模型 **454**
36.1 条件计算及专家结构 . . . 454
36.2 路由权重及局部梯度 . . . 456
36.3 词元路由 . . . 458
36.4 负载均衡及路由稳定性 . . . 459
36.5 总参数、激活参数及执行成本 . . . 463
36.6 共享专家、细粒度专家及软混合 . . . 464
36.7 专家并行的通信及训练推理差异 . . . 465
36.8 可执行结构还原 . . . 467

第三十七章　长上下文建模 **468**
37.1 上下文长度的四种含义 . . . 468
37.2 位置外推的分布偏移 . . . 470
37.3 位置插值及相位压缩 . . . 470
37.4 频率调整及短长距离折中 . . . 472
37.5 滑动窗口、稀疏连接及可达性 . . . 474
37.6 长序列训练的长度及任务分布 . . . 476
37.7 长度增长的计算、缓存及系统约束 . . . 477
37.8 上下文压缩、检索及长窗口的分工 . . . 479
37.9 长距离能力的分离评估 . . . 480
37.10 设计取舍及失效定位 . . . 482

第三十八章　开源模型体系 **483**
38.1 模型分析对象 . . . 484
38.2 DeepSeek 模型体系 . . . 487
38.3 Qwen 模型体系 . . . 491
38.4 Llama 及 Mistral 模型体系 . . . 492
38.5 GLM 模型体系 . . . 494

38.6 其他开源模型提供的技术比较 495
38.7 模型结构还原 497
38.8 模型选型条件 499

第三十九章 视觉 Transformer **501**
39.1 图像补丁表示 501
39.2 补丁投影及卷积的等价性 503
39.3 二维位置及序列组装 504
39.4 视觉编码及读出 506
39.5 分辨率及计算预算 507
39.6 归纳偏置及训练边界 508
39.7 视觉模型的评价边界 510

第四十章 多模态建模 **512**
40.1 模态、任务及表示接口 512
40.2 视觉词元化 513
40.3 视觉及文本的对比对齐 514
40.4 多模态连接器 516
40.5 融合位置及计算代价 518
40.6 监督区域及分阶段训练 519
40.7 音频表示及时间压缩 522
40.8 视频采样、同步及流式因果性 523
40.9 模型能力及制品一致性 525

第四十一章 扩散模型的概率原理 **526**
41.1 随机变量及前向链 526
41.2 前向边缘分布的闭式 527
41.3 反向条件后验的配方推导 528
41.4 变分目标 529
41.5 噪声预测及输出参数化 531
41.6 时间条件及训练路径 532
41.7 DDIM 采样 533
41.8 误差边界及参数化一致性 535

第四十二章 条件视觉生成 **537**
42.1 条件生成的对象及符号 537
42.2 时间条件及去噪主干 538
42.3 无分类器引导的推导及边界 539

42.4 潜空间表示及重建限制 540
42.5 流匹配及连续路径 541
42.6 编辑、视频及联合媒体 542
42.7 训练、采样及制品一致性 542

中文术语索引 **544**

英文术语索引 **552**

参考文献 **560**

插图

1.1 训练与应用的两条数据路径。实线表示数据或制品流动，虚线表示评估反馈；反馈进入下一轮修订，不自动改变正在服务的模型。 10

2.1 观测数据与模型通过负对数概率形成经验交叉熵。真实分布提供总体目标，有限样本只提供其估计；二者之间的差异不能由训练损失自动消除。 21

3.1 共享输入的计算图。实线表示前向依赖，虚线表示从乘法节点返回的梯度。 29

3.2 两层网络的前向与反向路径。图中 $G = \overline{U}$，$D = \overline{Z}$；各仿射层还分别产生权重与偏置梯度。 33

4.1 文本编码与模型协议的分层结构。训练过程产生词表及算法参数；运行过程使用固定制品编码。 43

4.2 字符串 aba 的分段格。每条从位置 0 到位置 3 的路径是一种分段；路径权重为边权之积。 48

4.3 填充、因果可见性和损失选择的职责。提示词进入条件上下文，不要求同时成为监督目标。 54

4.4 输入输出权重共享的两条参数路径。图中省略输出偏置；参数存储只有一份，反向贡献有两类。 58

5.1 四个位置的可见性模式，灰格表示允许读取。右图把前两个与后两个位置视为不同文档，每篇内部保持因果性。 65

5.2 单头例题第三个查询的加权汇总。点的位置由给定矩阵计算，表示真实二维值空间中的凸组合，不是训练测量。 67

5.3 两头结构示意。每个头分别归一化查询到键的读取权重，拼接发生在输出特征轴，不发生在序列轴。 68

5.4 自注意力与交叉注意力的信息来源。右侧的源记忆共享位置索引，查询位置和源位置不要求一一对应。 69

6.1 两层 FFN 与门控 FFN. 门控结构新增输入投影，并通过逐元素乘法形成输入相关的交互. 79
6.2 归一化位置改变前向函数和反向路径. Pre-LN 的恒等项绕过本子层归一化，末端归一化仍影响最终损失. 83
6.3 原始 Transformer 的编码器—解码器架构（Post-LN）. 主数据流自下而上；灰色旁路显式表示残差连接，加法后执行 LayerNorm. 蓝色跨列箭头将最终源记忆送入每个解码器层. 依据原始结构绘制 [6]. 85

7.1 二维旋转的几何含义. 图为结构示意，实线向量与虚线圆弧表示旋转关系，不是训练数据的投影. 97

8.1 相同因果网络的两种使用方式. 训练利用所有有效位置的已知目标，生成读取当前前缀末位的分布. 104
8.2 监督索引与因果前缀. 竖直箭头给出各位置的预测对象；横向虚线表示前缀依赖，实际注意力仍读取全部允许历史. 105
8.3 局部贪心错过完整高概率序列的概率树. 节点数值为累计概率；含 c 的前缀下一步以概率一结束. 112

9.1 从任务定义到决策证据. 每一层都保留自己的适用条件. 118

10.1 模型建议到执行能力之间的独立边界. 身份和凭据由控制层提供，外部内容不能授予权限. 137
10.2 事件处理闭环. 证据保全与遏制可并行开展，恢复以修复证据和受限能力为前提. 147

11.1 不同上下文组件进入同一条件预测过程；当前请求不改变模型参数. . 151
11.2 聚合与验证是不同决策分支，可以组合，但应分别记录各自依据. . . 156

12.1 HNSW 的概念层级：高层导航与底层候选探索. 示意连接不代表实际构建结果. 165
12.2 离线语料组织与在线检索链路. 生成回答不属于本图的检索职责. . . 168

13.1 受约束的检索生成流程. 迭代补充的是证据；身份、时间范围和资源上限由外部控制层保持. 178
13.2 知识发布与在线读取的关系. 内容快照保证可复现性，权限撤销仍须及时作用于在线访问. 181

14.1 领域应用以事实、受控执行和业务反馈形成闭环. 190

15.1 策略、外部授权与环境反馈构成的闭环. 拒绝和待批准只形成观测，不产生工具副作用. 194

15.2 模型位于受控执行闭环中. 上下文组织、准入、验证和恢复共同构成 Harness 的主要职责；记录成功返回不等于验证业务目标成立. 196
15.3 动作状态与受控分支. 等待批准不进入执行，已有批准仍须通过当前校验. 198
15.4 监督者与局部任务通过版本化共享状态协作. 202
15.5 工具动作的逻辑时序. 拒绝和待批准分支在执行前结束；图中下半段只描述允许分支. 205

16.1 应用状态、模型建议和外部执行各自承担明确职责. 207
16.2 发送可以重复，逻辑动作身份与业务回执保持稳定. 210
16.3 工单创建的持久执行时序. 当前授权重查失败时终止发送并记录拒绝；没有业务回执时不能走“已创建”答复路径. 216
16.4 以网络与凭证范围隔离模型推理和业务写入. 部署区表示职责边界，不规定必须使用六套集群. 218

17.1 二次例题的陡峭方向：实线振荡收敛，虚线振荡发散. 点由 $v_k = (1-9\eta)^k$ 精确计算，连线仅用于识别离散更新顺序. 223
17.2 学习与评估的信息边界. 虚线表示允许依据验证结果重新训练；最终测试结果不回流到本轮配置选择. 232

18.1 原始数据、语义记录与训练制品分层. 虚线表示控制约束，实线表示内容变换. 238
18.2 近似去重的候选生成与精确确认承担不同职责. 241
18.3 不可拆样本的顺序打包示意. 每格为一个位置，粗线是来源边界；是否允许跨边界注意力仍由掩码策略决定. 244

19.1 检查点保存与恢复的数据流. 发布发生在完整性确认之后；异步写入使用冻结快照. 254

20.1 目标位置与预测位置相差一位. 回答目标从 a 开始，而预测它的输出位于助手前缀末端. 259
20.2 指令数据到监督目标的形成过程. 内容正确性、模板编码与损失边界分别承担不同职责；评估反馈用于下一轮数据与方案选择. 264

21.1 LoRA 只训练低秩旁路，基座仍参与前向计算和输入梯度传播. . . . 270
21.2 QLoRA 的存储、计算和更新是三个不同边界. 虚线仅表示适配器参数梯度. 274
21.3 输入软提示只在入口增加可训练向量；逐层前缀在多个注意力层直接提供状态. 图示比较插入位置，不表示必须同时启用两种方法. . . . 277

23.1 独立文档装箱的两种边界：注意力禁止跨文档读取，损失忽略跨文档预测配对. 实线表示有效相邻监督. 288
23.2 数值例在固定计算下的损失余量. 曲线由本节解析函数计算；纵轴从零起算，曲线最低点对应参数与数据的边际收益平衡. 295
23.3 预训练同时积累模型知识与可恢复状态. 实线表示计算和保存路径，虚线表示恢复或下一轮实验设计；验证结果不直接生成当前训练目标. . 296

24.1 两步 MDP. 所有第一步奖励为零，第二步奖励给出完整回报. 该构造用于精确计算价值与策略梯度. 301
24.2 PPO 单样本裁剪目标的解析分段线. 裁剪限制有利方向上的继续增益，并不是把所有概率比强制限制在区间内. 307
24.3 策略训练的数据流. 行为策略产生数据，价值模型估计回报，当前策略执行更新，参考策略提供可选约束. 更新后须明确何时刷新采样策略. 308

25.1 DPO 的数据与梯度边界. 参考分支提供固定概率基线，不能随策略更新漂移. 316
25.2 在线奖励优化的职责. 参考、奖励、价值与策略是逻辑角色，不要求部署为四个完全独立的服务. 317

26.1 可验证推理训练的数据与策略循环. 独立评估不向训练回流隐藏答案. 329
26.2 测试时计算以剩余预算约束继续生成与验证，输出质量由选择结果而非候选数量决定. 332

27.1 资源供给与状态持久化形成闭环. 任一阶段的持续处理率不足，都可能限制整条训练路径. 344

28.1 加速器存储与执行层次. 图中表示逻辑路径，不表示所有架构具有相同缓存组织. 347
28.2 Roofline 的对数坐标示意. 折线是性能上界，不是测量曲线. 349
28.3 双 NUMA 节点的设备亲和性示意. 相同设备数量可能形成不同的数据路径. 350
28.4 环形全归约的两阶段时间结构. 格数仅用于示意，轮数由参与进程数决定. 353
28.5 分片检查点的数据与提交路径. 未提交分片可以回收，不可直接用于恢复. 355
28.6 集群的逻辑网络与数据通路. 管理、计算、存储和服务流量可以物理分离，也可以在共享网络上实施资源与权限隔离. 357

29.1 状态的逻辑所有权. 图中分片不包含前向或反向期间临时聚合的完整模块. 368

29.2 三个均衡阶段处理四个微批次的前向流水．每个微批次必须等待上一阶段，形成启动与排空空位；反向具有反向依赖． 371
29.3 八设备逻辑网格：两数据副本、两流水阶段、每阶段两张量成员．每个框对应两个不同物理设备，纵向同步匹配同一参数分片． 372

30.1 单层增量计算的数据流．新查询读取全部合法键值，历史查询和历史前馈输出不再重算． 377
30.2 四个查询头下的键值共享关系．查询头数相同，缓存键值头数依次为 4、2、1． 379
30.3 逻辑块映射到不连续物理块．页表解决存储布局，逻辑位置和有效长度决定模型可见的内容． 381
30.4 缓存块生命周期．CoW 边表示创建新的私有副本，原共享块仍保持原状态；取消与正常结束都按所有权释放一次引用．未完成的设备访问禁止复用物理存储． 383

31.1 在固定轮时模型下，连续批处理及时利用完成请求释放的槽位． . . . 388
31.2 每一层均有有限预算，释放信号与容量信号形成反馈． 392

32.1 模型服务的职责拓扑．实线表示请求或制品数据路径，虚线表示控制配置；流式输出沿请求链路返回． 397
32.2 请求状态机．终止状态不重新进入执行；重试必须具有明确尝试身份与既有请求的关联． 398
32.3 P/D 交接采用准备、校验和所有权提交．数据已到达不等于目标节点已经拥有输出权． 402

33.1 分块注意力的数据移动．完整分数矩阵不必写回显存，但查询仍处理全部合法键． 414
33.2 推测解码的提交边界．草稿产生的状态是暂存状态，拒绝后的后缀不能继续参与注意力． 417

34.1 范围内映射到最近格点，范围外映射到端点．端点外的误差随输入距离增长． 423
34.2 通道尺度变换保持原函数，却改变低精度表示的误差分配． 427
34.3 完整恢复与融合恢复的数据路径不同．必须确认实际执行路径，才能解释运行时收益． 430
34.4 前馈中间通道剪枝必须同步改变上下投影，保持外部残差接口． . . . 433
34.5 低秩因子替换原矩阵，实际执行两次投影． 436
34.6 输出分布蒸馏的数据与梯度边界．教师输出为固定目标，硬标签项与软标签项使用各自声明的温度． 440

35.1 共享表示与两类监督. 词元损坏不改变句对构造标签；两个头的梯度在共享主干汇合. 451

36.1 稀疏 MoE 的数据流. 专家批次大小由路由决定；选择多个专家意味着多条返回路径. 458

37.1 长上下文扩展的四项设计及共同评价边界. 实现其中一项，不足以证明完整长距离能力. 469
37.2 位置插值的坐标对应. 图中箭头表示坐标映射，不表示删除或合并词元. 471
37.3 单层直接可见范围：八个位置、局部窗口三. 着色单元表示允许读取；第三幅额外允许读取全局键位置 0，全部连接仍满足 $j \leqslant i$. 该开头位置不能汇聚未来信息. 474
37.4 局部窗口的跨层信息可达性. 三层、窗口四的位置 9 能够间接接收位置 0 的信息，但需要多次表示转换. 475

38.1 潜在内容与位置分量的分工. 图省略当前查询路径和输出投影，突出历史缓存的身份. 488

39.1 ViT 的表示路径. 分类读出压缩空间序列；密集预测可以保留补丁表示. 506

40.1 感知型视觉语言模型的模块边界. 冻结参数不会自动切断对上游可训练输入所需的梯度. 519
40.2 不同模态通过时间区间对应，而非按数组位置一一配对. 图为采样关系示意，不代表实际媒体. 524

42.1 条件 U-Net 的结构关系；跳跃连接传递特征，不绕过去噪任务. . . . 538
42.2 预测网络与求解器反复协作，解码仅在规定终点执行. 543

表格

9.1 主要统计对象. 118
9.2 发布时的代表性评测结果 122

10.1 风险、策略与执行记录使用的主要符号. 序数等级不具有概率单位. . 134
10.2 思考保护与反蒸馏控制的作用边界 144

11.1 本章主要符号与对象. 151
11.2 构造的五候选记录，只列最终结果和验证状态. 158

12.1 主要符号与形状. 161

13.1 检索增强生成的主要符号与形状. 171
13.2 知识服务的最小逻辑数据契约. 具体编码形式可变，字段语义应保持稳定. 181

14.1 领域应用的主要符号与计量对象. 187
14.2 业务缺口、候选机制及其验证边界. 187
14.3 领域应用评估指标及其统计口径. 191

16.1 智能体运行系统的主要符号. 207
16.2 智能体运行系统的核心持久对象与约束. 208
16.3 四种具体实现所展示的架构关注点；表格不表示性能排名或生产验收结果. 212
16.4 操作接口的选择应依据对象语义、可验证性和权限，而非仅依据是否能够点击成功. 215
16.5 不同故障位置的可确认事实与恢复动作. 217

18.1 主要符号与计量对象 235

19.1 训练稳定性的主要符号与计量对象. 248
19.2 训练检查点的状态组成 253

21.1 本章主要符号. 线性层采用列向量约定. 269

21.2 不同适配方法的主要资源与表达边界. 277

25.1 偏好学习的主要符号. 312
25.2 反馈学习方法的监督对象、计算角色与适用边界. 318

29.1 分布式训练的主要符号与逻辑设备量. 365

31.1 批处理调度的主要符号与计量对象. 387

33.1 推理计算优化的主要符号. 411

34.1 主要符号. 矩阵乘法采用列向量输入约定. 420
34.2 给定尺度与零点的手工编码结果，误差定义为 $\widehat{x} - x$. 422

38.1 开放程度的四个可分别取证的维度. 权重可下载不蕴含其余三项. . . 484
38.2 固定版本的结构、目标与资源比较. GB 为 10^9 字节；状态按两字节、单序列、标准键值或潜在缓存计算，未计 MTP 附加状态、分页和工作区. 485
38.3 两条开放权重路线的结构差异. 参数规模采用报告口径，本表不构成能力排序. 493
38.4 三个宽松许可开放权重制品的字段比较. 参数与许可采用发布者口径，配置数值取自各制品的固定提交；表内数值不构成能力比较. 496

39.1 视觉编码的主要符号. 501
39.2 固定图像尺寸下的序列规模. 508

40.1 本章主要符号. 矩阵推导未写批量轴时，对每个样本分别应用. . . . 512
40.2 目标掩码与上下文有效性的区别. 位置 5 的标签由位置 4 的输出预测. 520

42.1 条件视觉生成的主要符号. 537

第一部分

语言模型理论基础

第一章　绪论

语言模型研究的基本问题，是如何根据已有上下文描述后续符号出现的可能性. 这个问题可以用于文本续写，也可以通过适当的输入输出表示扩展到问答、翻译、代码生成与工具选择. 然而，从一个条件概率模型到可实际使用的应用，还必须经过数据构建、参数学习、生成决策、资源配置和效果验证. 只有同时理解这些环节，才能解释一个模型为什么给出某种答案，以及一个应用为什么能够或不能完成任务.

本章首先建立语言建模的研究对象，再沿发展历程解释表示、结构与训练方式的变化，介绍规模定律和涌现能力，进而区分模型、学习算法与应用系统. 这里出现的概率式用于表达问题结构，具体推导将在后续数学与模型章节展开.

本章问题

语言模型学习的是何种分布？技术发展怎样改变表示与学习方式？规模增长能预测什么，任务能力又应怎样验证？一个预测概率怎样转化为生成决策？这些问题决定后续各章的论述层次.

1.1　语言建模的研究对象

1.1.1　文本序列化

文本由字符组成，模型通常处理词元（Token）序列. 词元是编码规则规定的离散单元，可以对应字节、字符、词的一部分或特定控制标记. 给定有限词表 $\mathcal{V}$，一段文本经分词器转换为

$$x_{1:T} = (x_1, x_2, \ldots, x_T), \qquad x_t \in \mathcal{V}. \tag{1.1}$$

T 是编码后的序列长度，不必等于字符数或词数. 词元编号只表示其在词表中的身份；编号相邻不意味着含义接近. 连续向量表示及其学习属于模型的一部分，而不是整数编号本身的性质.

例如，“学习语言模型”可以被某种编码规则分成“学习”“语言”“模型”三个单元，也可以被另一种规则分成更多字符或字节单元. 两种编码可以都正确恢复原文，但词元数量、词表大小和预测任务已经不同. 因此，讨论上下文长度、每词元损失和生成速度时，需要同时知道所使用的编码规则.

语言模型给序列赋予概率. 设参数集合为 θ，模型分布记为 p_θ. 概率链式法则允许将联合概率分解为

$$p_\theta(x_{1:T}) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t}), \qquad x_{<t} = (x_1, \ldots, x_{t-1}). \tag{1.2}$$

第一个位置的条件前缀为空；实际系统也可以用约定的起始标记表示初始条件. 式中每一项回答同一个问题：在当前前缀已知时，下一词元的概率如何分布？模型需要在词表上给出非负且总和为一的概率，而不是只给出一个词元.

式 (1.2) 本身是概率恒等式. 用神经网络计算条件概率、限制可读取的前缀长度、在位置间共享参数，才是具体的建模选择. 区分恒等关系和模型假设，有助于避免把某种网络结构误认为语言建模的唯一形式.

1.1.2 条件生成及任务表示

在问答或翻译中，输入条件 c 与待生成序列 $y_{1:T}$ 具有不同职责. 模型描述的是

$$p_\theta(y_{1:T} \mid c) = \prod_{t=1}^{T} p_\theta(y_t \mid c, y_{<t}). \tag{1.3}$$

c 可以包含问题、对话历史、检索材料或经过编码的图像特征. 条件的表示方式不同，网络接收信息的路径也可能不同. 把图像转为视觉特征后接入语言模型，与把图像描述写成文本，并不构成相同的条件信息.

训练数据决定模型能够从哪些条件与答案的对应关系中学习. 目标函数决定哪些预测受到约束. 即使采用相同的网络，一个只预测回答部分的训练过程，与一个同时预测问题和回答的过程，也可能学习到不同的行为偏好.

序列通常还需要终止规则. 结束词元表达模型认为回答已经结束，最大长度则是运行时的外部限制. 有限词表上每步概率归一化，不足以单独保证模型必然在有限时间内生成结束词元；系统还需要给出明确的终止条件. [1]

1.2 语言模型的发展历程

语言模型的发展贯穿三条相互影响的线索：怎样表示上下文，怎样利用语料学习，以及怎样把预测能力用于任务. 下述时间节点用于定位代表工作；不同路线长期并存，结构的提出时间也不等于它在语言建模中广泛使用的时间.

[1] 固定长度序列的分布与所有有限长度序列的分布是不同对象. 后者若由逐步生成定义，还需要考虑终止概率；截断运行时间则是一项外部策略.

1.2.1　分布式词表示

Shannon 在 1948 年的通信理论工作中讨论了英语的统计近似，为把语言序列看作概率对象提供了早期基础 [1]．传统统计语言模型常采用n **元语言模型**（n-gram Language Model），用最近的 $n-1$ 个词近似完整前缀：

$$p(x_t \mid x_{<t}) \approx p(x_t \mid x_{t-n+1:t-1}). \tag{1.4}$$

这里假定已为序列开头补入适当的起始标记．给定历史 h，可用计数比 $\dfrac{c(h,w)}{c(h)}$ 估计下一词为 w 的概率，其中 $c(h,w)$ 为历史后出现该词的次数，$c(h)$ 为历史出现次数．若历史未出现，分母为零；若某种接续未出现，直接估计又会赋予零概率．平滑、回退与插值据此分配概率质量，并在长短上下文之间共享统计证据．

增加 n 可以利用更多局部信息，却也增加了需要估计的组合数量．有限语料难以覆盖大量长片段，语义相近的词也不会因整数身份不同而自动共享证据．2003 年 Bengio 等人的神经概率语言模型联合学习词的连续表示与条件概率，使相似表示可以参与共享计算 [2]．它仍采用有限窗口，但把部分泛化能力交给了可学习的表示与函数，而不必为每种上下文单独保存一组频数．

1.2.2　循环状态及长程依赖

循环神经网络（Recurrent Neural Network，RNN）将前缀逐步写入隐状态．2010 年 Mikolov 等人的循环语言模型是这一方向的代表工作 [3]．状态可随序列持续更新，因而摆脱了显式固定词窗口；不过，早期信息需要经过多次状态变换才能影响远处预测，学习容易受到梯度衰减或增长的影响．

长短期记忆（Long Short-Term Memory，LSTM）结构早在 1997 年已经提出，通过门控和记忆路径改善跨时间步的信息与梯度传递 [4]．它与后来循环语言模型的应用属于不同的时间节点．门控能够缓解长程学习困难，但固定维度状态仍需压缩历史，训练计算也保留沿时间推进的依赖．由此产生两个后续问题：能否直接访问相关位置，以及能否提高长序列训练的并行程度？

1.2.3　预训练表示及 Transformer

把无标注语料中学到的表示迁移给下游任务，使不同任务可以共享语言知识．静态词向量为同一词型提供固定表示，但一词多义需要结合上下文区分．2018 年的 ELMo 使用预训练双向语言模型的内部状态构造上下文相关表示，再把这些表示交给任务模型 [5]．

2017 年的 Transformer 以注意力组织序列到序列计算，使一个位置可以直接结合其他可见位置的信息，并允许训练时并行计算多个位置 [6]．这种并行性不消除自回

归生成对前一步输出的依赖；密集注意力也引入随序列长度二次增长的位置交互．它改善了信息访问和训练计算的组织方式，同时带来了新的存储与计算约束．

2018 年的 GPT 工作使用生成式预训练，再针对下游任务微调 [7]；同年公开、2019 年正式发表的 BERT 使用掩码预测学习双向表示 [8]．二者体现了预训练与任务适配的不同路径．BERT 的双向掩码目标与 GPT 的因果下一词预测在可见信息和概率含义上有区别，不能因共同使用 Transformer 就视为相同的语言建模目标．

1.2.4 规模化、上下文学习及指令交互

2020 年的 GPT-3 工作系统研究了在提示中给出说明或示例、保持参数不变的任务适应方式 [9]．这种**上下文学习**（In-Context Learning，ICL）改变输入条件，与通过梯度更新参数的微调不同．它拓宽了同一预训练模型的使用方式，但效果仍取决于任务、提示格式、示例选择与训练数据；少样本表现不能直接证明模型掌握了任意任务规则．

规模化还带来资源分配问题：同一计算预算应训练更大的模型，还是让较小模型读取更多数据？2020 年的 Kaplan 等人研究与 2022 年的 Chinchilla 工作，使这一问题可以通过损失曲线和经验拟合讨论 [10, 11]．参数增长、数据供给、优化过程及硬件系统共同影响结果，单独比较参数量不足以解释能力差异．

面向用户的交互还要求模型理解请求、遵循格式并体现偏好．2022 年的 InstructGPT 工作结合示范数据微调与人类反馈训练，展示了对预训练模型行为进行进一步塑造的路径 [12]．语言建模损失、指令完成质量和偏好评价对应不同目标；后训练改善交互行为的证据，仍需按所用评价方式解释．

1.2.5 推理训练、多模态及工具扩展

后续发展沿多个方向展开．2023 年的视觉指令微调工作把视觉表示接入语言交互 [13]；ReAct 将推理文本与环境动作交替组织 [14]；2025 年的 DeepSeek-R1 报告研究了通过强化学习提升推理表现的训练路径 [15]．这些工作分别改变可接收的信息、可执行的动作和训练反馈，不能只用模型规模解释．

在训练完成后，系统还可以为困难问题分配更多生成、搜索或验证计算．额外计算是否改善答案，取决于候选质量、验证信号与选择方式；更长的输出并不自动更可靠．模型接入工具后，实际执行、状态管理与权限控制也需要由应用系统承担．沿这条发展线索阅读后续章节，应分别追问：结构改变了哪条信息路径，训练改变了哪个目标，系统又增加了什么外部条件？

1.3　规模定律及涌现能力

1.3.1　规模定律的适用范围

规模定律（Scaling Law）是对模型、数据、计算与测量结果之间关系的经验描述. 对语言模型，常见研究对象是在可比数据分布上测得的平均验证交叉熵. 一个用于理解参数与数据约束的拟合形式为

$$L(N, D) \approx E + \frac{A}{N^{\alpha}} + \frac{B_s}{D^{\beta}}, \qquad A, B_s, \alpha, \beta > 0. \tag{1.5}$$

其中 N 为参数量，D 为训练词元数，E 为拟合底座，其余系数由实验估计. 式子表示：在适用范围内，扩大参数或数据可以降低相应的损失余量；只扩大其中一项，另一项仍可能限制收益. 拟合底座不应直接解释为已知的真实语言熵.

Kaplan 等人报告了特定实验条件下损失随规模变化的幂律规律；Chinchilla 工作进一步研究固定训练计算下的参数与数据配置，得到其计算最优范围内两者应近似同比例增长的结论 [10, 11]. 这些结果依赖各自的训练设置与拟合范围，不能转化为适用于所有模型的固定词元参数比. 模型结构、语料质量、重复使用数据及推理服务成本变化，都可能改变预算选择.

对常见密集模型的粗略训练预算，可写成 $C \approx \kappa N D$，其中 C 是训练计算量，κ 汇总结构与计算口径相关的系数. 在固定 C 下增加 N，通常需要压缩 D，因此更大模型未必带来更低损失. 第二十三章《预训练原理》进一步推导拟合、计算最优分配和外推验证. 这里需要先掌握的判断是：规模定律用于有条件的预测与规划，任务正确率仍需单独测量.

1.3.2　涌现能力的度量

涌现能力（Emergent Abilities）在 Wei 等人的 2022 年研究中，指一些在较小模型中未观察到、在较大模型中出现，且难以从小规模表现外推的任务能力 [16]. 这一说法描述的是给定模型系列、任务与测量方式下的现象. “小模型未观察到”受到测试集大小、随机基线、提示方式和评价粒度限制，并不直接说明其相关内部表示完全缺失.

2023 年 Schaeffer 等人指出，一些表面上的突跃可以由非线性或不连续的评价指标产生；改变度量方式后，部分能力曲线呈现平滑改善 [17]. 这一结果要求检验度量的影响，也没有证明所有任务变化都能用同一种解释消除. 任务要求完整答案正确时，精确匹配有实际意义；但仅凭该分数的跃升，尚不能推出内部机制发生了突变.

考虑一个用于说明度量效应的构造算例：答案包含 10 个位置，每个位置独立且以相同概率 q 正确，则整串完全正确的概率为 q^{10}. 当 q 从 0.5 增至 0.8 时，逐位置正确率从 50% 升至 80%，整串正确率却从约 0.10% 升至 10.74%. 少量测试题上，前

者可能长期测为零，随后才出现明显非零分数. 此处的独立性只服务于算例，真实生成中的错误通常相关；这些数值不是语言模型实验结果.

因此，研究涌现时应同时报告严格任务指标与更细粒度的诊断指标，并在疑似转折附近增加规模点、重复实验和不确定性估计. 模型之间的数据、训练目标、提示或解码方式若同时变化，观察到的差异也不能全部归因于规模. 数据污染与样本记忆则需要另行排查.

1.3.3 平均损失及任务能力的关系

平均损失汇总大量预测位置，任务评分则可能只关心某个领域或要求多个步骤同时正确. 前者平滑下降时，后者仍可能因阈值和组合要求而快速变化；平均改善也可能掩盖少数任务的退步. 规模定律与涌现研究因此关注不同层次的观测.

训练计算与推理时计算也应分开记录. 增加训练词元会改变参数的学习经历；增加候选采样或搜索预算则改变同一模型的使用过程. 两者可以分别研究收益曲线，但不能把训练损失的幂律直接用于预测搜索成功率，更不能由任何一条曲线保证事实正确性、可靠推理或生产可用性.

1.4 概率预测、学习目标及生成决策

1.4.1 统计估计

模型通常不能直接知道真实语言分布，只能观察有限语料. 设数据集为 $D = \{x^{(1)}, \ldots, x^{(n)}\}$，一种基本学习方式是选择使观测数据似然尽可能大的参数，即最大似然估计：

$$\theta^* \in \arg\min_{\theta} \left[-\sum_{i=1}^{n} \log p_\theta(x^{(i)}) \right]. \tag{1.6}$$

将序列概率分解代入，便得到逐词元负对数概率的求和. 把它写成可微计算后，优化算法可以据此调整参数. 预测、损失与参数更新构成训练计算的主线.

该目标说明模型受到什么监督，却不保证所有任务属性都被显式优化. 训练语料可能包含错误，某些领域可能代表不足，模型也可能无法充分拟合已有数据. 因此，“能生成自然语言”与“所述事实正确”不是可以互相替代的判断. 事实核验、任务约束和系统控制仍需独立处理.

自监督学习从数据自身构造目标，下一词元预测就是其中一种方式. 它不要求人工为每个词元另行标注，但仍依赖数据的采集、筛选、混合与组织. 是否存在人工标签，不决定数据处理是否影响模型行为.

1.4.2　局部决策及序列决策

得到概率后，还需要决定如何选择输出．考虑只生成两个词元的简化任务．首步只有 a、b 两个候选，其概率分别为 0.6 和 0.4．第二步候选为 u、v，条件概率如下．

首词元	首步概率	$p(u \mid$ 首词元$)$	$p(v \mid$ 首词元$)$
a	0.6	0.5	0.5
b	0.4	0.9	0.1

由乘法规则，四条完整序列的概率为

$$p(a,u) = 0.6 \times 0.5 = 0.30, \qquad p(a,v) = 0.30, \tag{1.7}$$

$$p(b,u) = 0.4 \times 0.9 = 0.36, \qquad p(b,v) = 0.04. \tag{1.8}$$

四者之和为一．逐步选择最大条件概率的贪心策略先选择 a，随后无论怎样处理第二步的并列，得到的序列概率都是 0.30；联合概率最大的序列却是 (b,u)，其概率为 0.36．因此，局部最大选择并不保证完整序列的概率最大．

这个例子还没有给出哪条序列是任务的正确答案．概率是模型对数据规律的表达，任务正确性则由任务本身定义．即使能够精确找出最大概率序列，也仍需要检验它是否满足问题要求．采样、搜索和验证分别解决不同环节的问题．

1.5　模型、算法及系统的分工

1.5.1　网络结构及学习过程

网络结构规定输入信息如何转为输出分数，包括表示维数、模块连接、注意力可见范围和非线性变换．训练算法规定数据如何被抽取、损失如何形成、梯度如何计算以及参数如何更新．相同结构经过不同数据和目标训练，可以获得不同能力；相同目标采用不同结构，也可能具有不同的计算和泛化特征．

Transformer 通过注意力交换位置信息，再通过逐位置非线性变换加工表示．编码器、因果解码器和编码器—解码器都可以由这些组件构成，但它们的可见性及输出目标不同．原始 Transformer 将注意力用于序列到序列建模 [6]，不能将其中某个子层直接等同于一个完整语言模型．

模型规模也不是单一维度．参数数量反映所存储的可学习自由度，激活参数量描述某次计算实际使用的参数范围，上下文长度描述可接收的信息范围，而训练词元数描述训练暴露的数据量．这些量相互影响，但不能互相替代．

1.5.2　学习方式及结构坐标

监督学习（Supervised Learning）以给定标签或目标值约束预测；**无监督学习**（Unsupervised Learning）从观测中刻画分组、低维表示或分布结构；**自监督学习**（Self-

Supervised Learning）由数据自身构造目标，例如用文本前缀预测后续词元；**强化学习**（Reinforcement Learning，RL）通过动作、状态转移和奖励优化决策. 它们描述反馈来源和学习问题，并不分别对应一种固定网络. 语言模型预训练的目标由数据自身构造，但优化时仍有明确的监督词元.

结构	核心机制	适用性与限制
线性模型 / 树模型	线性打分 / 递归划分特征空间	可形成表格任务基线；能力取决于特征与划分.
多层感知机（Multilayer Perceptron，MLP）	全连接映射与逐点非线性组合	适合固定维度表示，不自动利用空间邻接.
卷积神经网络（Convolutional Neural Network，CNN）	局部连接和空间共享参数	利用局部结构；跨远距离关系需多层或其他机制.
循环神经网络（Recurrent Neural Network，RNN）	$h_t = f_\theta(h_{t-1}, x_t)$ 的递归状态	顺序更新；历史压缩在状态中，路径长度随时间增长.
Transformer	内容相关注意力与逐位置变换	可按可见性构造不同序列模型，计算代价随连接数增长.
扩散模型（Diffusion Model）	学习逐步去噪或对应连续动力学	在数据或潜空间生成；采样通常需多步计算.

长短期记忆是带门控状态的循环结构，使用不同路径控制记忆保留与更新；它仍保留递归时间依赖. 上述坐标不是质量排名：局部连接、递归状态和注意力分别引入不同的**归纳偏置**（Inductive Bias），即在有限数据下偏向某些函数和信息交互方式. 具体选择还要结合训练目标、数据规模和部署约束.

1.5.3 训练系统及应用系统

训练系统将语料变成参数，应用系统将请求变成结果. 两者共享模型制品，却具有不同的状态和故障模式. 训练关心数据消费、参数更新和恢复一致性；应用关心请求生命周期、响应时间、工具执行与结果有效性.

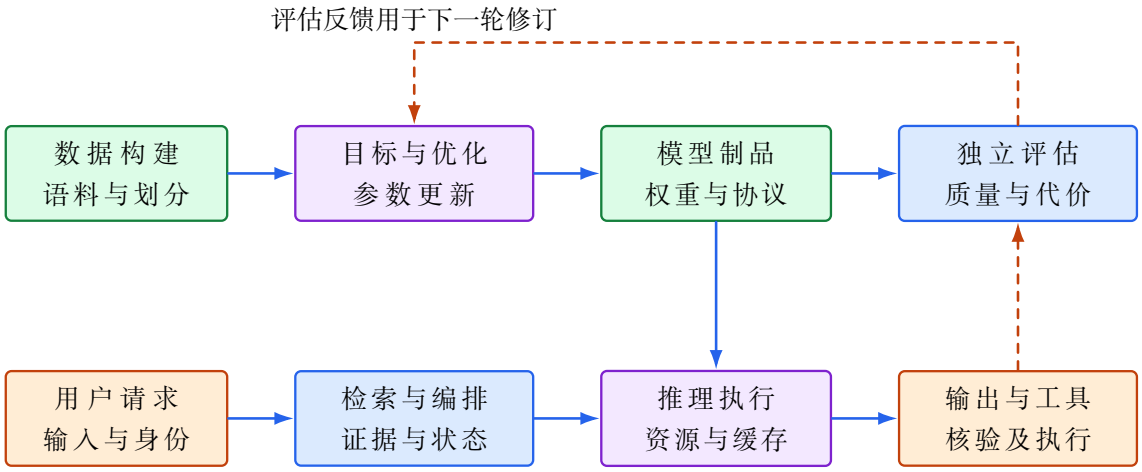


图 1.1: 训练与应用的两条数据路径. 实线表示数据或制品流动，虚线表示评估反馈；反馈进入下一轮修订，不自动改变正在服务的模型.

图 1.1 中的模型制品不仅包含权重，还包括解释权重所需的结构配置、分词器和输入输出协议. 对于适配器或多模态模型，还需要相应基座、处理器及连接模块. 一个能够加载的权重文件，只说明某一层面的兼容性，并不证明输入处理、任务质量或运行性能正确.

GPU 提供计算和显存，主机负责数据处理与运行控制，网络承载通信，存储保存语料、模型和检查点. 任一环节供给不足，都可能使其余资源等待. 因此，算力规划需要考虑端到端数据路径，不能只把设备峰值算力相加.

1.5.4　检索及工具扩展能力的方式

检索增强生成从外部知识库选择材料，再把材料作为回答条件. 工具调用则使应用能够执行查询、计算或业务操作. 前者主要改变模型可见的信息，后者改变系统可以采取的动作. 智能体进一步根据中间观察选择动作顺序，因而引入规划、记忆、状态恢复和终止等问题.

这些能力依赖模型外部的执行机制. 模型产生一个工具名称和参数，不意味着相应动作已经发生；执行器必须检查参数、权限和运行结果. 模型生成了一个引用，也不意味着引用材料确实支持该论断. 系统需要保留这些区别，才能定位错误并采取有效修正.

1.6　知识体系及学习路径

共同基础围绕一次模型请求建立概念链：文本如何成为词元与张量，注意力与网络怎样计算条件分布，生成策略怎样选择与终止输出，以及效果和风险怎样检验. 数

学知识先支撑这些基本对象与运算，再随具体推导加深. 书前阅读说明给出各方向的章节顺序.

应用智能体系统关注任务、证据和执行结果，沿上下文学习、检索与 RAG 进入领域应用，再深入智能体架构和运行系统. Harness 是其中模型之外的执行与控制部分，重点在任务状态、工具权限和失败恢复. 模型训练方法关注数据怎样改变模型行为，从监督微调和参数高效适配进入预训练、强化学习、人类反馈学习；完整梯度推导和训练目标属于这一方向的前置.

训练推理系统关注模型计算的资源与执行，先建立容量预算和硬件瓶颈分析，再分别进入训练系统或推理服务. 训练分支需要理解反向传播和训练状态；推理分支需要理解增量解码、缓存与请求生命周期. 数据、评估、安全和多模态按各方向的对象与任务深入.

选择方向时，以要解释和验证的对象为依据. 同一个应用可以使用已有模型，模型适配可以使用已有训练框架，系统优化也可以在固定模型上开展；各方向在制品、接口和证据处衔接. 学习成果应体现为可解释的机制、可核查的假设和可验证的判断.

第二章　数学基础

本章快速复习后续内容反复使用的数学对象．向量承载特征，矩阵实现批量线性变换，梯度描述标量目标对参数的局部变化；概率则规定语言模型输出的含义．读者只需重新建立符号、形状和概念之间的联系，不需要在此完成一门独立的数学课程．随后将这些对象用于推导**最大似然估计**（Maximum Likelihood Estimation，MLE）、**熵**（Entropy）、**交叉熵**（Cross-Entropy，CE）和**库尔贝克—莱布勒散度**（Kullback–Leibler Divergence，KL）．

假设与适用范围

词表取有限集合；除明确讨论连续情形外，概率均为离散质量．涉及条件概率的条件事件具有正概率；需要有限对数损失时，预测分布在目标支持集上严格为正．样本独立同分布只在相应估计推导中使用，不作为所有文本片段天然满足的事实．

2.1　向量、矩阵及线性变换

2.1.1　向量的方向及尺度

向量可以看作一组有顺序的数，也可以看作空间中的点或方向．设 $x = (x_1, \dots, x_d)^{\mathsf{T}} \in \mathbb{R}^d$，其欧氏长度和两个向量的内积分别为

$$\|x\|_2 = \sqrt{\sum_{i=1}^{d} x_i^2}, \qquad x^{\mathsf{T}} y = \sum_{i=1}^{d} x_i y_i. \tag{2.1}$$

内积同时包含长度与夹角信息．余弦相似度 $\dfrac{x^{\mathsf{T}} y}{\|x\|_2 \|y\|_2}$ 去除了长度影响，但只有在向量非零时才有定义．在语言模型中，一个向量常表示某个词元或位置的特征；坐标本身通常没有独立的固定语义，向量之间的关系才是模型使用的信息．

向量相加要求维数相同．标量乘法改变长度并可能改变方向．逐元素乘法 $x \odot y$ 与内积不同：前者仍是向量，后者将公共索引求和后得到标量．看到一个公式时，先写出输入和输出形状，往往比先代数化简更容易发现错误．

2.1.2 矩阵表示批量数据和线性映射

矩阵 $A \in \mathbb{R}^{m\times n}$ 有 m 行、n 列．矩阵乘法

$$y = Ax, \qquad y_i = \sum_{j=1}^{n} A_{ij}x_j \tag{2.2}$$

把 n 维输入映射到 m 维输出．乘法成立的条件是相邻的公共维数相同；公共维被求和，外侧维被保留．若一批样本按行组成 $X \in \mathbb{R}^{B\times n}$，常将同一变换写为 $Y = XW$，其中 $W \in \mathbb{R}^{n\times m}$．这只是批处理约定不同，并不改变每个样本执行同一映射的事实．

线性变换满足 $f(ax + by) = af(x) + bf(y)$．神经网络常用的“线性层”实际计算

$$z = Wx + b, \tag{2.3}$$

含偏置 b 时严格说是**仿射变换**（Affine Transformation）．连续堆叠多个仿射变换仍可合并为一个仿射变换，因此网络需要激活函数引入非线性．转置 A^{T} 交换行列；逆矩阵只对可逆方阵存在．深度学习中通常直接计算线性方程或矩阵乘积，不应为了形式方便显式求逆．

2.2 导数、梯度及链式法则

一元导数描述函数在某一点附近的变化率．多元标量函数 $f : \mathbb{R}^d \to \mathbb{R}$ 的梯度为

$$\nabla_x f = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_d}\right)^{\mathsf{T}}. \tag{2.4}$$

梯度与输入同形，并指向欧氏空间中函数上升最快的局部方向；负梯度因此是局部下降方向．梯度为零只说明一阶变化消失，不保证该点是全局最小值．

复合函数通过链式法则求导．若 $z = g(x)$、$f = h(z)$，则

$$\frac{\partial f}{\partial x} = \frac{\partial z}{\partial x}^{\mathsf{T}} \frac{\partial f}{\partial z}. \tag{2.5}$$

计算图中的反向传播就是从标量损失出发，沿依赖关系反向应用这条规则．同一变量经多条路径影响损失时，各路径贡献必须相加．矩阵求导、广播和自动微分的具体规则将在第三章《神经网络基础》继续说明．

2.3 概率基础

概率描述不确定事件，而统计推断利用有限样本认识其分布．离散样本空间中，概率质量函数满足

$$p(x) \geqslant 0, \qquad \sum_{x} p(x) = 1. \tag{2.6}$$

条件概率 $p(x \mid y)$ 表示已知 y 后对 x 的重新分配；它与联合概率的关系为 $p(x,y) = p(x \mid y)p(y)$. 语言模型正是反复估计“给定已有文本，下一个词元如何分布”. 期望是按概率加权的平均，方差衡量随机变量围绕期望的离散程度. 后续各节将在这些基本概念上展开最大似然、交叉熵和序列概率.

2.4　随机变量及分布

2.4.1　概率质量函数

随机变量（Random Variable）X 是对不确定结果的数值或符号表示. 大写 X 表示随机变量，小写 x 表示其一个可能取值. 若 X 取值于有限集合 $\mathcal{X}$，**概率质量函数**（Probability Mass Function，PMF）写为

$$p_X(x) = P(X = x), \qquad p_X(x) \geqslant 0, \qquad \sum_{x \in \mathcal{X}} p_X(x) = 1. \tag{2.7}$$

在不会混淆对象时，下标 X 可以省略. 模型中的一个词元编号是一次取值，而词表上的概率向量描述所有候选取值的不确定性. 不能把某个编号的大小解释为该词元的概率.

例如，若 $\mathcal{X} = \{a, b, c\}$ 且概率为 $(0.5, 0.3, 0.2)$，事件 $X \in \{a, c\}$ 的概率是 0.7. 概率具有可加性，是因为这些离散取值对应互斥事件. 两个并非互斥的事件则不能直接将概率相加，否则交集会被重复计算.

本章主要使用有限分布，以便将概念直接连接到词表预测. 连续随机变量用**概率密度函数**（Probability Density Function，PDF）f_X 描述区间概率：

$$P(a \leqslant X \leqslant b) = \int_a^b f_X(x)\,\mathrm{d}x, \qquad \int_{-\infty}^{\infty} f_X(x)\,\mathrm{d}x = 1. \tag{2.8}$$

密度值可以大于一，因为概率由区间积分而非单点密度给出. 对具有密度的变量，单点事件的概率为零；这与离散词元概率不同.

2.4.2　记号及支持集

符号	含义
X, Y；x, y	随机变量及其取值.
$p(x,y)$；$p(y \mid x)$	联合概率及给定 x 后的条件概率.
$p_*(x)$	讨论对象的真实分布，通常未知.
$p_\theta(x)$	参数为 θ 的模型分布.
$\widehat{p}_n(x)$	n 个观测形成的经验频率分布.
$\mathbb{E}_p[g(X)]$	函数 $g(X)$ 在分布 p 下的期望.
$H(p)$；$H(p,q)$	熵；以 p 为目标、q 为预测的交叉熵.
$D_{\mathrm{KL}}(p\|q)$	从目标分布 p 到分布 q 的 KL 散度.

支持集（Support）是分布赋予正概率的取值集合，记为 $\operatorname{supp}(p)=\{x : p(x)>0\}$. 涉及 $\log q(x)$ 的期望若在某个 $p(x)>0$ 的位置遇到 $q(x)=0$，就可能为无穷. 这个边界不是公式上的小细节：一个模型若对真实可能出现的结果给出零概率，该结果一旦发生就会产生无穷大的负对数损失.

除特别说明外，本章使用自然对数，信息量单位为 nat. [1]

2.5　联合概率、条件概率及链式分解

2.5.1　条件分布

若 $p_X(x)>0$，条件概率定义为

$$p(y \mid x)=\frac{p(x,y)}{p_X(x)}, \qquad p_X(x)=\sum_{y} p(x,y). \tag{2.9}$$

条件化把样本空间限制到已知条件，并重新归一化. 它不是删除条件变量，也不是默认条件与结果具有因果关系. 网页内容与某个答案同时出现，可以形成统计相关；这并不证明网页内容导致答案所述事件发生.

将式 (2.9) 两边乘以 $p_X(x)$，得到乘法规则

$$p(x,y)=p_X(x)p(y \mid x). \tag{2.10}$$

交换分解顺序可得贝叶斯公式

$$p(x \mid y)=\frac{p(y \mid x)p_X(x)}{\sum_{x'} p(y \mid x')p_X(x')}, \tag{2.11}$$

要求分母 $p_Y(y)>0$. 分子衡量取值 x 与观测 y 的共同可能性，分母汇总全部候选解释，使结果归一化.

设一个验证器判断生成答案是否正确. 候选答案的先验正确率为 0.2；正确答案有 0.9 的概率通过，错误答案有 0.1 的概率误通过. 记正确事件为 C，通过事件为 A. 联合概率为

$$P(C\cap A)=0.2\times 0.9=0.18, \tag{2.12}$$

$$P(C^{c}\cap A)=0.8\times 0.1=0.08, \tag{2.13}$$

$$P(A)=0.18+0.08=0.26. \tag{2.14}$$

因此

$$P(C \mid A)=\frac{0.18}{0.26}=\frac{9}{13}\approx 0.6923. \tag{2.15}$$

[1] 改用以二为底的对数时，单位为 bit. 两种记法相差常数 $\log 2$；数值比较应保持底数一致. 密度的对数还与坐标及计量单位有关，离散熵的全部性质不能不加条件地移到微分熵.

通过后的正确率高于原来的 0.2，却并非 0.9．$P(A \mid C)$ 表示正确答案被接受的概率，$P(C \mid A)$ 表示接受结果中正确答案的比例，条件方向不同．基础正确率较低时，错误答案数量较多，即使误通过率较低也会形成不可忽略的污染．

这里的数值是用于说明条件概率的设定值，不是验证器性能实测．实际系统必须从合适样本中估计这些概率，并进一步考虑验证器与候选生成器的相关性和分布变化．

2.5.2　序列联合分布的分解

对三个随机变量，连续应用乘法规则有

$$p(x_1, x_2, x_3) = p(x_1, x_2)p(x_3 \mid x_1, x_2) \tag{2.16}$$

$$= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2). \tag{2.17}$$

对一般长度 T，归纳得到

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t \mid x_{<t}). \tag{2.18}$$

这个分解没有假定各位置独立．相反，右端保留了完整前缀条件．只有进一步假定 $p(x_t \mid x_{<t}) = p(x_t)$，才会退化为独立取值乘积．以固定窗口替代完整前缀同样是一项额外限制．

若每个条件分布均归一化，固定长度联合分布也归一化．证明从最后一个位置开始求和：

$$\sum_{x_{1:T}} \prod_{t=1}^{T} p(x_t \mid x_{<t}) = \sum_{x_{1:T-1}} \left[\prod_{t=1}^{T-1} p(x_t \mid x_{<t}) \right] \underbrace{\sum_{x_T} p(x_T \mid x_{<T})}_{1} \tag{2.19}$$

$$= \cdots = \sum_{x_1} p(x_1) = 1. \tag{2.20}$$

这解释了为何逐位置的词表分布能够定义完整序列概率．零概率前缀上的条件分布可以任意补成归一化分布，而不改变联合概率；除法定义只需用于具有正概率的前缀．

2.6　期望、方差及经验分布

2.6.1　期望的概率加权

对有限分布，函数 $g(X)$ 的期望为

$$\mathbb{E}_p[g(X)] = \sum_{x} p(x)g(x). \tag{2.21}$$

期望不要求是变量的一种可能取值. 例如公平二元变量取 0 或 1，期望为 0.5，但一次观测不必出现 0.5. 对任意常数 a, b，期望的线性性给出 $\mathbb{E}[aU + bV] = a\mathbb{E}[U] + b\mathbb{E}[V]$，无须假定 U, V 独立.

记均值为 $\mu = \mathbb{E}[X]$，方差为

$$\operatorname{Var}(X) = \mathbb{E}[(X - \mu)^2] = \mathbb{E}[X^2] - \mu^2. \tag{2.22}$$

最后一个等式由展开平方并使用 $\mathbb{E}[X] = \mu$ 得到. 若 U, V 相关，则

$$\operatorname{Var}(U + V) = \operatorname{Var}(U) + \operatorname{Var}(V) + 2\operatorname{Cov}(U, V), \tag{2.23}$$

其中 $\operatorname{Cov}(U, V) = \mathbb{E}[(U - \mathbb{E}U)(V - \mathbb{E}V)]$. 忽略协方差可能严重低估误差，例如同一文档的多个近重复片段并非新的独立证据.

2.6.2 有限样本估计

设 $X_1, \dots, X_n$ 独立同分布，均值为 μ、有限方差为 σ^2. 样本均值 $\overline{X} = n^{-1} \sum_i X_i$ 满足

$$\mathbb{E}[\overline{X}] = \mu, \qquad \operatorname{Var}(\overline{X}) = \frac{1}{n^2} \sum_i \operatorname{Var}(X_i) = \frac{\sigma^2}{n}. \tag{2.24}$$

第一式来自线性性，第二式还使用了不同样本间协方差为零. 标准差随 $n^{-\frac{1}{2}}$ 缩小，而不是随 n^{-1} 缩小. 将记录复制十次不会产生十倍独立信息.

对离散取值，经验分布为

$$\widehat{p}_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}[X_i = x]. \tag{2.25}$$

于是对任意函数 g，

$$\mathbb{E}_{\widehat{p}_n}[g(X)] = \sum_x \widehat{p}_n(x) g(x) = \frac{1}{n} \sum_{i=1}^{n} g(X_i). \tag{2.26}$$

经验平均可以严格写成经验分布下的期望. 但经验分布并不等于真实分布：未观察到某个取值，不能证明它在总体中的概率为零. 后续的训练目标正是由这个可计算的经验平均构成.

2.7 最大似然及负对数损失

2.7.1 似然及概率的对象不同

假设观测 $x^{(1)}, \dots, x^{(n)}$ 在给定参数后独立，模型给数据的似然为

$$\operatorname{Lik}(\theta; D) = \prod_{i=1}^{n} p_\theta(x^{(i)}). \tag{2.27}$$

数据固定时，这个表达式被视为参数 θ 的函数，称为似然．它一般不在参数空间上积分为一，因此不应直接解释为“参数为真的概率”．参数后验还需要先验和相应归一化．

由于对数严格单调，最大化似然与最大化对数似然具有相同的最优解．再乘以负号，得到最小化问题

$$\theta^* \in \arg\min_{\theta} \mathcal{L}_n(\theta), \qquad \mathcal{L}_n(\theta) = -\frac{1}{n}\sum_{i=1}^{n} \log p_\theta(x^{(i)}). \tag{2.28}$$

对数将连乘转为求和，既便于分析，也避免大量小概率连乘导致的数值下溢．除以固定的 n 不改变最优解，但改变梯度尺度；优化时仍需与学习率等设置共同考虑．

2.7.2　伯努利参数估计

设 $X \in \{0,1\}$，$p_\theta(X=1)=\theta$，$0<\theta<1$．在 n 次独立观测中出现 k 次一，负对数似然为

$$J(\theta) = -k\log\theta - (n-k)\log(1-\theta). \tag{2.29}$$

当 $0<k<n$ 时，求导并令导数为零：

$$J'(\theta) = -\frac{k}{\theta} + \frac{n-k}{1-\theta} = 0 \quad\Longrightarrow\quad \widehat{\theta} = \frac{k}{n}. \tag{2.30}$$

且 $J''(\theta) = \dfrac{k}{\theta^2} + \dfrac{n-k}{(1-\theta)^2} > 0$，所以该驻点是唯一最小值．例如十次中六次为一，估计值为 0.6．当 $k=0$ 或 $k=n$ 时，若参数域包含端点，最优解分别位于 0 或 1；不能在这些情形机械套用内部驻点条件．

端点解也说明经验估计的局限．训练样本没有某个结果时，最大似然可能将其概率设为零．引入先验、平滑或更强的共享结构可以改变这一行为，但这些操作也改变了估计问题，需明确说明．

2.7.3　条件监督、回归及多标签

在分类任务中，通常给定输入 x_i，拟合 $p_\theta(y_i \mid x_i)$．二分类令 $p=\sigma(z)=(1+e^{-z})^{-1}$，则伯努利负对数似然为

$$\ell(z,y) = -y\log p - (1-y)\log(1-p) \tag{2.31}$$

$$= \log(1+e^{z}) - yz \tag{2.32}$$

$$= \max(z,0) - yz + \log(1+e^{-|z|}). \tag{2.33}$$

第二行来自代入 Sigmoid，第三行分别对 $z\geqslant 0$ 与 $z<0$ 提取指数中的较大项．最后一式中指数不大于一，避免了很大的正指数．这是损失稳定计算的数学依据．

例如 $z=2$、$y=1$ 时，$p\approx 0.8808$，损失为 $\log(1+e^{-2})\approx 0.1269$；若标签改为零，损失为 $2+\log(1+e^{-2})\approx 2.1269$. 同一预测对不同真值应受到不同惩罚. [2]

对连续目标，若假定 $Y\mid x\sim\mathcal{N}(f_\theta(x),\sigma^2)$ 且方差固定，则

$$-\log p_\theta(y\mid x)=\frac{1}{2}\log(2\pi\sigma^2)+\frac{(y-f_\theta(x))^2}{2\sigma^2}. \tag{2.34}$$

去掉与参数无关的项，最小化负对数似然等价于最小化平方误差. 若方差也需要学习，第一项便不能删除；离群值和异方差也可能要求不同的分布模型.

互斥多分类用一个总和为一的类别分布，多标签则允许多个标签同时成立. 若将各标签建模为**条件独立**（Conditional Independence）的伯努利变量，其联合似然为各标签似然的乘积，损失为各自二元交叉熵之和. 条件独立是所采用的分布结构，并非多标签数据天然满足的性质.

类别不平衡时，若仅把正类损失乘以 $w>0$，总体条件目标变为 $-wp_*\log q-(1-p_*)\log(1-q)$，其中 $p_*=P(Y=1\mid x)$. 求导并令其为零，得到最优输出 $q^*=\dfrac{wp_*}{wp_*+1-p_*}$. 除 $w=1$ 或退化端点外，q^* 不再等于原始条件概率. 加权目标改变了拟合对象，不能将其输出直接解释为已校准概率；决策阈值也应在独立验证中按错误代价选择.

2.8 熵、交叉熵及 KL 散度

2.8.1 熵衡量分布内部的不确定性

信息熵（Information Entropy）源于对概率不确定性的定量研究 [1]. 以下用有限分布推导其性质，信息不确定性与文本语义价值须区分.

对概率 $p(x)>0$ 的事件，定义自信息为 $I(x)=-\log p(x)$. 较少见的结果具有较大的自信息；两个独立结果联合出现时，其自信息相加，因为 $-\log[p(x)p(y)]=-\log p(x)-\log p(y)$. 按真实出现概率平均，得到熵

$$H(p)=-\sum_x p(x)\log p(x). \tag{2.35}$$

取 $0\log 0=0$，这与 $u\log u$ 在 $u\to 0^+$ 时的极限一致. 离散概率不大于一，故 $H(p)\geqslant 0$；确定性分布的熵为零.

熵描述一个分布自身的不确定性，并不评价模型是否正确. 一个模型把错误答案的概率集中到接近一，也可以具有很低的预测熵. 低熵表示确信，确信是否合理还需要与目标分布或观测结果比较.

[2] 对 $y\in[0,1]$ 的软标签，同一式子仍定义两个类别间的交叉熵；此时 y 不再表示一次伯努利取值，而表示目标概率或平均监督.

2.8.2　交叉熵及散度的分解

设目标分布为 p，模型或编码分布为 q，交叉熵定义为

$$H(p,q) = -\sum_{x} p(x)\log q(x). \tag{2.36}$$

采样权重来自 p，被取对数的预测来自 q，两者不可交换．将 $\log q(x) = \log p(x) - \log[\dfrac{p(x)}{q(x)}]$ 代入，在有限且支持相容时得到

$$H(p,q) = -\sum_{x} p(x)\log p(x) + \sum_{x} p(x)\log\frac{p(x)}{q(x)} \tag{2.37}$$

$$= H(p) + D_{\mathrm{KL}}(p\|q). \tag{2.38}$$

第一项不随模型 q 改变，第二项衡量预测分布偏离目标分布所带来的额外平均对数损失．因此，对固定目标分布最小化交叉熵，等价于最小化这个方向的 KL 散度．

KL 散度的非负性可以由基本不等式 $-\log u \geqslant 1-u$ 推出．假设在 p 的支持上 q 为正，则

$$D_{\mathrm{KL}}(p\|q) = -\sum_{x:p(x)>0} p(x)\log\frac{q(x)}{p(x)} \tag{2.39}$$

$$\geqslant \sum_{x:p(x)>0} [p(x)-q(x)] \tag{2.40}$$

$$= 1 - \sum_{x:p(x)>0} q(x) \geqslant 0. \tag{2.41}$$

等号成立要求支持上的比值均为一，且 q 没有把额外概率放到支持之外，即 $p=q$．若 q 在 p 的某个正概率位置为零，散度为正无穷，非负性仍成立．由此可以进一步证明，在 C 个取值上，$H(p)\leqslant\log C$：令 $u(x)=\dfrac{1}{C}$，则 $D_{\mathrm{KL}}(p\|u)=\log C - H(p)\geqslant 0$．最大熵由均匀分布取得．

2.8.3　KL 散度方向性

令 $p=(0.75,0.25)$，$q=(0.5,0.5)$，逐项计算得到

$$H(p) = -0.75\log 0.75 - 0.25\log 0.25 \approx 0.5623, \tag{2.42}$$

$$H(p,q) = -0.75\log 0.5 - 0.25\log 0.5 \approx 0.6931, \tag{2.43}$$

$$D_{\mathrm{KL}}(p\|q) = 0.75\log 1.5 + 0.25\log 0.5 \approx 0.1308. \tag{2.44}$$

因此 $0.6931 \approx 0.5623 + 0.1308$，与式 (2.38) 一致．反向计算却得到

$$D_{\mathrm{KL}}(q\|p) = 0.5\log\frac{0.5}{0.75} + 0.5\log\frac{0.5}{0.25} \approx 0.1438. \tag{2.45}$$

二者不同，说明 KL 散度一般不对称. 它也不满足距离所需的三角不等式，因此本书不将其称为通常意义的距离. 训练和蒸馏中的 KL 项必须说明哪个分布提供采样权重；改变方向便改变了目标.

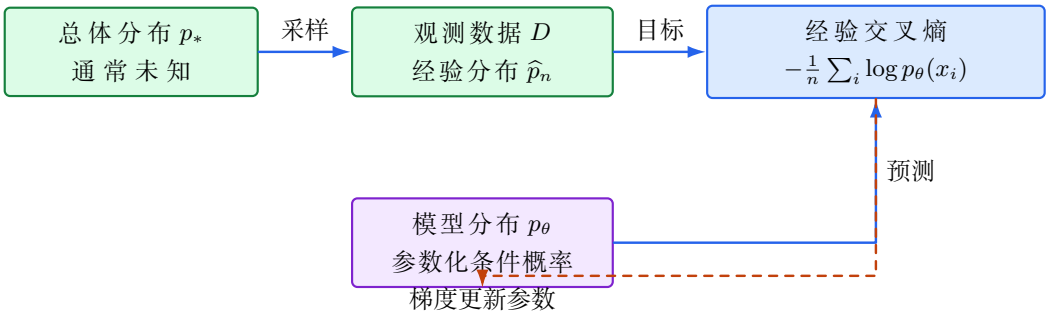


图 2.1: 观测数据与模型通过负对数概率形成经验交叉熵. 真实分布提供总体目标，有限样本只提供其估计；二者之间的差异不能由训练损失自动消除.

2.9 序列损失

2.9.1 独热目标及 Softmax

互斥 C 分类的目标类别为 y，独热向量为 e_y. 对预测分布 q，

$$H(e_y, q) = -\sum_{j=1}^{C} \mathbf{1}[j = y] \log q_j = -\log q_y. \tag{2.46}$$

因此用整数类别编号索引正确类别，与显式构造独热向量再求交叉熵，在数学上给出同一标量. 前者不需要分配整个独热数组. 软标签则有多个非零权重，不能再简化为只读取一个类别.

令网络产生实数分数 $z \in \mathbb{R}^C$，**归一化指数函数**（Softmax）将其映射到概率单纯形：

$$q_j = \frac{e^{z_j}}{\sum_k e^{z_k}}, \qquad \ell(z, y) = -z_y + \log \sum_k e^{z_k}. \tag{2.47}$$

对全部分数减去相同常数不会改变概率. 令 $a = \max_k z_k$，有

$$\log \sum_k e^{z_k} = a + \log \sum_k e^{z_k - a}. \tag{2.48}$$

右端指数都不大于一. 稳定实现应计算这种形式，而不是先计算可能溢出的指数再求对数. 分数的绝对平移不可识别，真正决定概率的是分数差. 具体梯度将在张量与自动微分章推导.

2.9.2　条件生成的最大似然

给定上下文 c，长度为 T 的目标 $y_{1:T}$ 满足

$$p_\theta(y_{1:T} \mid c) = \prod_{t=1}^{T} p_\theta(y_t \mid c, y_{<t}), \qquad -\log p_\theta(y_{1:T} \mid c) = -\sum_{t=1}^{T} \log p_\theta(y_t \mid c, y_{<t}). \tag{2.49}$$

这里先按固定长度讨论；可变长度的完整生成事件还需包含终止词元，其概率处理见第八章《自回归语言建模》．分类是目标仅含一个类别的特例；翻译、问答和指令回答则把类别预测重复到整个目标序列．条件 c 可以进入每一步计算，却不一定属于损失监督区域．仅屏蔽条件位置的损失，并不会删除模型对条件信息的依赖．

2.9.3　序列归一化及样本权重

对文档 i 的有效预测位置集合 $\mathcal{T}_i$，令 $N_i = |\mathcal{T}_i|$．序列链式分解给出损失和

$$S_i = -\sum_{t \in \mathcal{T}_i} \log p_\theta(x_t^{(i)} \mid x_{<t}^{(i)}). \tag{2.50}$$

总词元平均为 $\mathcal{L}_{\text{token}} = \dfrac{\sum_i S_i}{\sum_i N_i}$，文档等权平均则为 $\mathcal{L}_{\text{doc}} = n^{-1} \sum_i \dfrac{S_i}{N_i}$，这里要求被平均的文档均有 $N_i > 0$．二者只有在特定条件下才相等．

例如两篇文档分别含两个和八个有效目标，平均词元损失分别为一和二，则 $S_1 = 2$、$S_2 = 16$．总词元平均是 $\dfrac{18}{10} = 1.8$，文档平均是 $\dfrac{1+2}{2} = 1.5$．后者对短文档的每个词元赋予更大权重．损失归一化并非只影响显示格式，它直接改变不同样本对学习目标的贡献．

填充位置和不参与监督的提示位置应由明确的有效位置集合排除．注意力掩码规定信息可见性，损失掩码规定优化区域，概率公式不能将两者混为一谈．若全部监督位置都被屏蔽，应报告无有效目标，而不能计算除零的平均损失．

2.9.4　困惑度的含义及比较条件

若平均负对数损失使用自然对数，**困惑度**（Perplexity，PPL）定义为

$$\mathrm{PPL} = \exp(\mathcal{L}_{\text{token}}) = \left[\prod_{i,t \in \mathcal{T}_i} \frac{1}{p_\theta(x_t^{(i)} \mid x_{<t}^{(i)})} \right]^{\frac{1}{N}}, \quad N = \sum_i N_i. \tag{2.51}$$

它是正确目标逆概率的几何平均．若模型对每个正确目标都赋予 $\dfrac{1}{K}$，困惑度为 K；一般情况下，它不是每步实际候选数，也不是错误率．指数是单调的，所以在相同目标和归一化下，困惑度与平均负对数损失的排序相同．

跨模型比较时，需要固定语料、分词器、有效位置、上下文和边界处理. 分词更粗的模型可能在更少位置上完成同一文本的编码，直接比较每词元损失会改变计量单位. 改用每字节或每字符指标也必须明确文本的编码与归一化条件，不能据此忽略输入协议差异.

第三章　神经网络基础

神经网络把仿射变换与非线性组合为可训练函数．本章先快速串联神经元、激活函数、损失、Softmax、梯度下降、反向传播和训练循环，再用张量与自动微分说明这些步骤如何可靠地落到批量计算上．一个损失函数即使在统计意义上合理，如果计算时交换了样本轴与特征轴、遗漏了共享参数的梯度，仍然会得到错误的训练过程．

本章采用行向量批处理约定，从线性映射出发建立矩阵微分，再将链式法则组织成计算图．随后推导分类损失、广播和嵌入查表的反向计算，并完成一个可以逐项复核的两层网络例题．这里讨论的是梯度如何正确产生；如何选择更新方向、学习率和正则化，属于第十七章《优化及泛化》中的优化问题．

3.1　前馈神经网络

3.1.1　神经元计算

一个人工神经元先计算输入的加权和，再通过激活函数得到输出：

$$z = w^{\mathsf{T}} x + b, \qquad a = \phi(z). \tag{3.1}$$

w 决定各输入对结果的作用，b 调整激活阈值，ϕ 提供非线性．把多个神经元并排写成矩阵形式，便得到一层网络 $A = \phi(XW + b)$．层数增加并不改变这一基本单元，只是让前一层的输出成为后一层的输入．

常见激活函数包括 ReLU、Sigmoid 和双曲正切：

$$\mathrm{ReLU}(z) = \max(0, z), \qquad \sigma(z) = \frac{1}{1 + e^{-z}}, \qquad \tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}. \tag{3.2}$$

ReLU 计算简单，正区间梯度不衰减；Sigmoid 将数值压到 $(0, 1)$，适合表示二元概率；tanh 输出以零为中心．Sigmoid 和 tanh 在绝对值较大时导数接近零，深层或长序列中容易出现梯度消失．激活函数的选择既改变表达能力，也改变梯度传播．

3.1.2 标量损失目标

网络输出必须通过损失函数与目标比较. 回归常用均方误差

$$\mathcal{L}_{\text{MSE}} = \frac{1}{B}\sum_{i=1}^{B}(\hat{y}_i - y_i)^2. \tag{3.3}$$

互斥多分类先用 Softmax 把分数 u_j 变成概率，再用交叉熵评价目标类别:

$$q_j = \frac{e^{u_j}}{\sum_{k=1}^{C} e^{u_k}}, \qquad \ell = -\sum_{j=1}^{C} y_j \log q_j. \tag{3.4}$$

Softmax 的各输出共享同一个分母，所以类别概率相互耦合. 它适合“只能选一个类别”的任务；多标签任务通常对各标签分别使用 Sigmoid. 实际计算交叉熵时应直接使用原始分数的稳定实现，避免先算极小概率再取对数.

3.2 训练、梯度下降及反向传播

训练的目标是寻找使训练目标较小的参数 θ. 最基本的梯度下降更新为

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta_t), \tag{3.5}$$

其中 η 是学习率. 学习率过小会使训练缓慢，过大可能越过低损失区域而振荡或发散. 小批量训练用一部分样本估计梯度，降低单步成本，也引入抽样噪声.

一次训练迭代包含五个阶段：取出一个批次；执行前向传播；计算标量损失；反向传播得到所有参数的梯度；由优化器统一更新参数. 反向传播只负责求梯度，梯度下降或 Adam 等优化器负责如何更新，两者不能混为一谈. 更新之前还应清楚梯度是按样本求和还是求平均，是否跨多个批次累积.

训练集损失下降说明优化正在拟合训练数据，不等同于模型对新数据有效. 验证集用于选择超参数和训练停止点，测试集用于最后评估；若测试信息参与选择，它就不再是独立测试. 更完整的学习率、初始化、正则化和泛化问题见第十七章《优化及泛化》.

3.3 张量、形状及线性映射

3.3.1 形状表达计算对象的语义

在本书的计算语境中，张量（tensor）指带有若干索引轴的数值数组. 标量没有轴，向量有一个轴，矩阵有两个轴. 三维张量 $X \in \mathbb{R}^{B\times T\times d}$ 可以表示 B 个样本、每个样本 T 个位置、每个位置 d 个特征. X_{itj} 则是第 i 个样本在第 t 个位置的第 j 个特

征. [1] 形状只给出各轴的长度，轴的语义还需要文字约定. 两个形状同为 $B \times B$ 的矩阵，一个可能描述样本之间的相似度，另一个可能只是特征维数碰巧等于批量大小；形状相同不意味着含义相同.

本章主要讨论二维批处理，并使用下列符号. 单个向量在定义梯度和雅可比时视为列向量；批量矩阵中每个样本按行存放. 两种约定之间通过转置明确转换.

符号	含义	形状或范围
B, d, h, C	批量大小、输入宽度、隐藏宽度、类别数	正整数
X	批量输入，每行一个样本	$B \times d$
W_1, b_1	第一层权重和偏置	$d \times h,\ h$
Z, A	隐藏层预激活和激活	$B \times h$
W_2, b_2	第二层权重和偏置	$h \times C,\ C$
U, Q, Y	分类分数、预测概率和目标分布	$B \times C$
$\mathcal{L}$	批量平均损失	标量
$\overline{R}$	损失对中间量 R 的梯度	与 R 同形
$\mathbf{1}_B$	元素全为一的列向量	B

3.3.2　矩阵乘法的公共轴归约

设 $X \in \mathbb{R}^{B \times d}$，$W \in \mathbb{R}^{d \times h}$，则线性映射 $Z = XW$ 的分量形式为

$$Z_{ij} = \sum_{k=1}^{d} X_{ik} W_{kj}, \qquad Z \in \mathbb{R}^{B \times h}. \tag{3.6}$$

这里被求和的 k 是输入特征轴；样本轴 i 与输出特征轴 j 得以保留. 对每个样本使用同一个 W，意味着参数在样本之间共享，而不是为每一行分配一套权重.

若加入偏置 $b \in \mathbb{R}^h$，仿射映射为

$$Z = XW + \mathbf{1}_B b^{\mathsf{T}}, \qquad Z_{ij} = \sum_{k} X_{ik} W_{kj} + b_j. \tag{3.7}$$

偏置平移输出，因此严格说包含偏置的层是仿射层. 习惯上所称的线性层常包含这项平移，推导时应保留它的独立梯度.

矩阵乘法不同于逐元素乘法. $A \odot D$ 要求元素一一对应，或先按明确规则广播；AD 则沿一个公共轴收缩. 将 XW 误写成逐元素运算，可能直接触发形状错误，也可能因特殊尺寸而得到合法却完全不同的函数. 核对分量公式可以识别后一种隐蔽错误.

序列上的共享线性层可以写为

$$Z_{itj} = \sum_{k=1}^{d} X_{itk} W_{kj} + b_j, \qquad Z \in \mathbb{R}^{B \times T \times h}. \tag{3.8}$$

[1] 微分几何中的张量还涉及坐标变换法则. 本章使用深度学习计算中的数组含义，不将数组维数与几何张量的变换性质混为一谈.

它只变换最后一个特征轴，不混合不同位置. 暂时将前两个轴合并为 BT 行，是执行同一计算的一种方式，前提是恢复形状时保留位置与样本的对应关系.

3.3.3 转置、重排及存储布局

转置 X^{T} 交换矩阵的两个索引：$(X^{\mathsf{T}})_{ji} = X_{ij}$. 重塑形状则按约定的元素顺序重新分组. 将一个 2×3 数组直接重塑为 3×2，通常不等于转置；前者不自动把行索引变成列索引. 在多头注意力等运算中，重排轴与合并轴必须分别说明，否则即使元素总数不变，位置之间的关系也可能被破坏.

数学上的形状还应与物理存储区分. 某些重排可由索引步长描述，不必立即复制数据；另一些后续运算会要求重新组织连续存储. 两种实现可能计算同一函数，却具有不同的数据搬运代价. 本章首先保证语义正确，存储带宽和算子效率将在算力与推理部分讨论.

3.4 矩阵梯度

3.4.1 梯度刻画标量函数的一阶变化

对于可微函数 $\mathcal{L}:\mathbb{R}^n\to\mathbb{R}$，在 x 附近有

$$\mathcal{L}(x+\Delta x)=\mathcal{L}(x)+\nabla_x\mathcal{L}^{\mathsf{T}}\Delta x+o(\|\Delta x\|). \tag{3.9}$$

梯度的第 j 个分量是 $\dfrac{\partial\mathcal{L}}{\partial x_j}$. 该展开中的线性项常记为全微分 $\mathrm{d}\mathcal{L}=\nabla_x\mathcal{L}^{\mathsf{T}}\,\mathrm{d}x$. 它描述任意微小扰动的损失变化，而不仅是沿某一个坐标的变化.

对矩阵 X，将各元素的贡献相加，得到

$$\mathrm{d}\mathcal{L}=\sum_{i,j}\overline{X}_{ij}\,\mathrm{d}X_{ij}=\langle\overline{X},\mathrm{d}X\rangle_F=\mathrm{tr}(\overline{X}^{\mathsf{T}}\,\mathrm{d}X). \tag{3.10}$$

其中 $\langle A,B\rangle_F=\sum_{ij}A_{ij}B_{ij}$ 是 Frobenius 内积，tr 表示方阵对角元素之和. 式 (3.10) 定义了与 X 同形的矩阵梯度；不必把矩阵展平后再写一个高阶导数数组.

例如，$\mathcal{L}=\dfrac{1}{2}\|X\|_F^2=\dfrac{1}{2}\sum_{ij}X_{ij}^2$，故 $\mathrm{d}\mathcal{L}=\sum_{ij}X_{ij}\,\mathrm{d}X_{ij}$，于是 $\overline{X}=X$. 若系数 $\dfrac{1}{2}$ 被省略，梯度变成 $2X$. 这类常数不会改变单独最小化该项的极小点，却会改变与其他损失项组合时的相对尺度.

3.4.2 线性层梯度的推导

令 $Z=XW+\mathbf{1}_B b^{\mathsf{T}}$，上游已经给出 $G=\overline{Z}\in\mathbb{R}^{B\times h}$. 在一次微分中，三个输入分别产生

$$\mathrm{d}Z=(\mathrm{d}X)W+X(\mathrm{d}W)+\mathbf{1}_B(\mathrm{d}b)^{\mathsf{T}}. \tag{3.11}$$

代入矩阵内积，并利用迹在维数相容时的循环性质，可得

$$\mathrm{d}\mathcal{L} = \operatorname{tr}\big(G^{\mathsf{T}}(\mathrm{d}X)W\big) + \operatorname{tr}\big(G^{\mathsf{T}}X(\mathrm{d}W)\big) + \operatorname{tr}\big(G^{\mathsf{T}}\mathbf{1}_B(\mathrm{d}b)^{\mathsf{T}}\big) \tag{3.12}$$

$$= \operatorname{tr}\big((GW^{\mathsf{T}})^{\mathsf{T}}\,\mathrm{d}X\big) + \operatorname{tr}\big((X^{\mathsf{T}}G)^{\mathsf{T}}\,\mathrm{d}W\big) + (G^{\mathsf{T}}\mathbf{1}_B)^{\mathsf{T}}\,\mathrm{d}b. \tag{3.13}$$

与梯度的定义逐项比较，得到

$$\boxed{\overline{X} = G W^{\mathsf{T}}, \qquad \overline{W} = X^{\mathsf{T}} G, \qquad \overline{b} = G^{\mathsf{T}}\mathbf{1}_B.} \tag{3.14}$$

这三个式子分别回答：误差如何传回输入、各样本如何共同调整权重，以及共享偏置如何接收整批样本的影响.

也可以用索引检查权重梯度. 由于 $\dfrac{\partial Z_{ij}}{\partial W_{kr}} = X_{ik}\mathbf{1}[j=r]$，链式法则给出

$$\frac{\partial \mathcal{L}}{\partial W_{kr}} = \sum_{i,j} G_{ij} X_{ik} \mathbf{1}[j=r] = \sum_{i} X_{ik} G_{ir} = (X^{\mathsf{T}}G)_{kr}. \tag{3.15}$$

求和不是一种额外的经验处理，而是共享参数影响多个输出所必需的结果. 梯度形状也随之确定：GW^{T} 为 $B \times d$，$X^{\mathsf{T}}G$ 为 $d \times h$，$G^{\mathsf{T}}\mathbf{1}_B$ 为 h.

3.4.3　逐元素非线性及不可微点

若 $A_{ij} = \phi(Z_{ij})$，各输出仅依赖对应输入，故

$$\overline{Z} = \overline{A} \odot \phi'(Z). \tag{3.16}$$

整流线性单元（Rectified Linear Unit，ReLU）为 $\phi(z) = \max(0, z)$. 在 $z > 0$ 时导数为一，在 $z < 0$ 时为零. 因此正区间直接传递梯度，负区间阻断经过该节点的局部梯度；在零点不存在通常意义下的唯一导数. [2]

如果两个仿射层之间没有非线性，则 $(XW_1 + \mathbf{1}_B b_1^{\mathsf{T}})W_2 + \mathbf{1}_B b_2^{\mathsf{T}}$ 仍可合并为一个仿射映射. 非线性使不同输入区域具有不同的有效映射，是两层网络超出单层表达形式的关键. 其反向代价则是梯度要乘以依赖前向状态的局部导数，因而需要保存或重新计算该状态.

3.5　计算图及自动微分

3.5.1　链式法则的依赖结构

设 $z = f(x) \in \mathbb{R}^m$，$x \in \mathbb{R}^n$. 雅可比矩阵（Jacobian matrix）定义为 $J_f \in \mathbb{R}^{m \times n}$，其中 $(J_f)_{ij} = \dfrac{\partial z_i}{\partial x_j}$. 微分满足 $\mathrm{d}z = J_f\,\mathrm{d}x$. 若 $\mathcal{L} = g(z)$，则

$$\mathrm{d}\mathcal{L} = \overline{z}^{\mathsf{T}} J_f\,\mathrm{d}x, \qquad \overline{x} = J_f^{\mathsf{T}}\overline{z}. \tag{3.17}$$

[2] ReLU 是凸函数，其零点次微分为 $[0, 1]$. 计算实现需要约定零点取值，常见约定为零. 这不意味着包含 ReLU 的整个非凸网络拥有相同的凸分析性质.

正向计算把输入映射到输出；反向计算把输出上的损失敏感度映射回输入．后者对应**向量雅可比积**（Vector–Jacobian Product，VJP）：行向量形式为 $\bar{z}^{\mathsf{T}} J_f$，本章使用其转置 $J_f^{\mathsf{T}} \bar{z}$ 表示列梯度．它通常不必显式形成整个雅可比．

计算图把基本运算表示为节点，把依赖表示为有向边．对于无循环的计算图，可按拓扑顺序求出前向值，再按相反顺序应用式 (3.17)．即使模型包含循环结构，有限次展开的一次执行也可以表达为这样的依赖序列．

一个变量若经由两条路径影响损失，两条路径的贡献必须相加．考虑 $a = x^2$、$b = 3x$、$\mathcal{L} = ab$．前向在 $x = 2$ 时得到 $a = 4$、$b = 6$、$\mathcal{L} = 24$；反向先得到 $\bar{a} = b = 6$、$\bar{b} = a = 4$，再得到

$$\bar{x} = \bar{a}(2x) + \bar{b}\,3 = 6 \times 4 + 4 \times 3 = 36. \tag{3.18}$$

直接对 $3x^3$ 求导同样得到 $9x^2 = 36$．若仅保留最后一条路径，结果便会错误地变成 12．权重共享、残差连接和重复嵌入编号都遵循同一条累加原则．

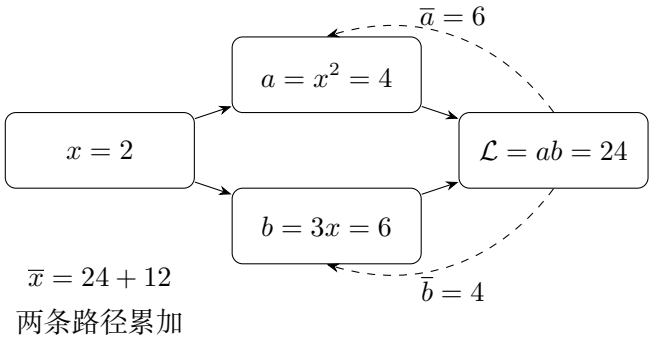


图 3.1: 共享输入的计算图．实线表示前向依赖，虚线表示从乘法节点返回的梯度．

3.5.2 正向模式及反向模式

自动微分按照基本运算的导数规则，对实际执行的组合函数计算导数．它与把表达式化简成一个符号公式不同，也不通过微小扰动近似导数．除基本运算与浮点计算本身的误差外，链式法则不引入有限差分那样的步长截断误差．

正向模式给定输入方向 v，随前向传播计算 $J_f v$；反向模式给定输出权重 w，从后向前计算 $J_f^{\mathsf{T}} w$．如果一个函数具有大量参数而只有一个标量损失，反向模式以输出端种子 $\bar{\mathcal{L}} = 1$ 开始，一次反向遍历即可得到所有参数的梯度．逐坐标使用正向模式则通常需要多次遍历．因此，标量训练目标特别适合反向模式．若输入方向很少、需要多个输出的方向响应，正向模式也有价值．

反向模式并非没有成本．局部梯度往往依赖前向中间量，例如线性层需要 X，ReLU 需要知道 Z 的符号．这些量可以保存在计算过程中，也可以在反向时重新计算．保存占用内存，重算消耗计算，二者的取舍构成后续激活重计算方法的基础．

3.5.3　梯度计算及参数更新

在一次前向和反向之间，参数以及反向所需的中间量必须保持一致．若计算 $\overline{W}_2$ 后立刻更新 W_2，再使用更新后的 W_2 计算隐藏层梯度，那么该梯度已经不再对应原来的前向函数．正确过程是完成当前损失的全部反向计算，再进行参数更新．

对于多个小批量，梯度既可以在每批后清除，也可以有意累加．但累加的归一化必须与目标一致：两个批次分别含 B_1, B_2 个有效样本时，总体平均梯度为

$$g = \frac{B_1 g_1 + B_2 g_2}{B_1 + B_2}, \tag{3.19}$$

其中 g_1, g_2 是各自的平均梯度．简单平均两个批次的平均梯度，仅在两批大小相等或目标确实赋予每批相等权重时成立．

计算图中的切断梯度会保留某个数值却改变对其来源的求导关系．离散索引选择、阈值决策、随机采样也不能不加说明地按光滑函数处理．训练模式和是否记录梯度同样是两个问题：随机失活等机制影响前向函数，梯度记录则决定是否保存求导依赖．关闭梯度记录不会自动把随机训练函数变成确定性的评估函数．

算法 3.1　标量损失的反向自动微分

输入：有限无环计算图、输入与固定参数，各节点的前向运算及局部反向规则．**输出：**损失对各可训练参数的梯度．

1. 按拓扑顺序执行前向计算，保存局部求导所需状态，得到标量损失 $\mathcal{L}$．
2. 将所有节点的伴随量初始化为零，并令 $\overline{\mathcal{L}} = 1$．
3. 按逆拓扑顺序访问每个节点 $v = f(u_1, \ldots, u_k)$．对每条输入边，执行 $\overline{u}_i \leftarrow \overline{u}_i + J_{f,u_i}^{\mathsf{T}} \overline{v}$；重复引用同一节点时仍逐条累加．
4. 返回参数节点的伴随量．完成全部反向计算后，优化器才可更新参数．

不变量：访问某节点时，其所有后继已将经过各自路径的贡献累加到该节点．若单个基本运算的反向成本与前向同阶，总算术成本与计算图执行成本同阶；保存中间状态的内存另计．不可微点采用已声明的求导约定．

3.6　分类损失的导数及数值稳定性

3.6.1　Softmax 的雅可比

设单个样本的分数列向量为 $u \in \mathbb{R}^C$，其第 j 类概率为

$$q_j = \frac{\exp(u_j)}{S}, \qquad S = \sum_{k=1}^{C} \exp(u_k). \tag{3.20}$$

分子在 $j=r$ 时依赖 u_r，分母在所有情况下都依赖 u_r. 同时求导，得到

$$\frac{\partial q_j}{\partial u_r} = \frac{\mathbf{1}[j=r]\exp(u_j)S - \exp(u_j)\exp(u_r)}{S^2} \tag{3.21}$$

$$= q_j\big(\mathbf{1}[j=r] - q_r\big). \tag{3.22}$$

因此对角元素为 $q_j(1-q_j)$，非对角元素为 $-q_jq_r$，矩阵形式为

$$J_{\text{softmax}} = \operatorname{diag}(q) - qq^{\mathsf{T}}. \tag{3.23}$$

非对角项体现类别之间的竞争：一个分数升高时，其他类别概率会因公共分母增大而下降. 只保留对角项会遗漏这种耦合.

给定概率端的梯度 $g=\overline{q}$，式 (3.23) 与 g 的乘积可以写为

$$\overline{u} = q \odot \big(g - (q^{\mathsf{T}}g)\mathbf{1}_C\big). \tag{3.24}$$

显式雅可比需要 C^2 个元素，而这个向量形式只需要长度为 C 的数组及一次内积. 公式相同不意味着必须采用同样的存储方式.

由于 $J_{\text{softmax}}\mathbf{1}_C = 0$，沿所有分数同时增加相同常数的方向，概率不变. 这也可由 $\operatorname{softmax}(u + c\mathbf{1}_C) = \operatorname{softmax}(u)$ 直接验证. 对归一化交叉熵，其分数梯度各分量之和为零，正是同一不变性的另一种表现.

3.6.2 交叉熵梯度化简

设目标分布 $y_j \geqslant 0$ 且 $\sum_j y_j = 1$，交叉熵为 $\ell = -\sum_j y_j \log q_j$. 先对概率求导，再代入 Softmax 雅可比：

$$\frac{\partial \ell}{\partial u_r} = \sum_j \left(-\frac{y_j}{q_j}\right) q_j\big(\mathbf{1}[j=r] - q_r\big) \tag{3.25}$$

$$= -y_r + q_r \sum_j y_j = q_r - y_r. \tag{3.26}$$

因此，对 B 个样本取平均并沿类别轴归一化时，$\overline{U} = \dfrac{Q-Y}{B}$. 独热标签是 Y 的特例；软标签只要保持非负且行和为一，也满足相同公式.

若目标权重之和为 $s \neq 1$，梯度实际为 $sq - y$，不能继续机械使用 $q-y$. 例如多标签向量可以同时含多个一，其语义是多个二元事件，通常不应强行放入互斥类别的归一化模型. 这说明损失的张量形状相同，也不保证任务定义相同.

3.6.3 分数空间损失

对独热目标类别 k，交叉熵可以直接写为

$$\ell(u,k) = -u_k + \log \sum_j e^{u_j}. \tag{3.27}$$

令 $m=\max_j u_j$，利用指数的公共因子得到

$$\log\sum_j e^{u_j}=m+\log\sum_j e^{u_j-m}. \tag{3.28}$$

右侧指数的自变量均不大于零，且至少一个等于零，从而避免直接计算巨大正分数的指数．对于有限分数，数学上的 Softmax 概率严格为正；浮点计算却可能将极小概率舍入为零．若先形成概率再取对数，就可能得到无穷损失；直接使用式 (3.28) 可以避免这条不稳定路径．

例如 $u=(1000,999)$，直接计算 e^{1000} 可能溢出，减去最大值后只需计算 $(1,e^{-1})$．两种写法在实数算术中等价，数值可靠性却不同．稳定变换也不能修复输入本身已经包含的非有限值；对异常输入仍应追溯其上游来源．

3.6.4　二分类及多标签的对应关系

二分类用一个分数 z 表示正类相对于负类的对数几率，通过 $\sigma(z)=\dfrac{1}{1+e^{-z}}$ 得到正类概率．对目标 $y\in[0,1]$，二元交叉熵为

$$\ell(z,y)=-y\log\sigma(z)-(1-y)\log(1-\sigma(z)) \tag{3.29}$$

$$=\log(1+e^{z})-yz \tag{3.30}$$

$$=\max(z,0)-yz+\log(1+e^{-|z|}). \tag{3.31}$$

最后一式给出稳定的计算形式．由中间一式求导可得

$$\frac{\partial\ell}{\partial z}=\frac{e^{z}}{1+e^{z}}-y=\sigma(z)-y. \tag{3.32}$$

把两类 Softmax 分数设为 $(0,z)$，恰好得到相同的概率和损失，因此二者的梯度结果是一致的．多标签任务则对每个类别分别构造这样的二元损失，不要求各类别概率之和为一．

模型输出原始分数之后，应将其交给包含该稳定变换的损失计算．若先进行一次 Sigmoid，再把所得概率当成分数传入同类损失，相当于优化另一个复合函数，既改变预测范围，也改变梯度．是否已经执行概率变换应由公式判断，而不是仅凭变量名称判断．

考虑两层感知机（Multilayer Perceptron，MLP）：

$$Z=XW_1+\mathbf{1}_B b_1^{\mathsf{T}},\quad A=\mathrm{ReLU}(Z),\quad U=AW_2+\mathbf{1}_B b_2^{\mathsf{T}},\quad Q=\mathrm{softmax}(U). \tag{3.33}$$

取 $B=d=h=C=2$，给定

$$X=\begin{bmatrix}1&2\\-2&1\end{bmatrix},\quad W_1=\begin{bmatrix}1&-1\\1&1\end{bmatrix},\quad W_2=\begin{bmatrix}\frac{1}{2}&-\frac{1}{2}\\-\frac{1}{2}&\frac{1}{2}\end{bmatrix},\quad b_1=b_2=\begin{bmatrix}0\\0\end{bmatrix}. \tag{3.34}$$

第一个样本属于第 1 类，第二个样本属于第 2 类，故目标矩阵 $Y = I_2$．这里的参数特意取为便于手算的数值，例题检验的是计算过程，不代表经过训练的模型．

逐步相乘得到

$$Z = \begin{bmatrix} 1+2 & -1+2 \\ -2+1 & 2+1 \end{bmatrix} = \begin{bmatrix} 3 & 1 \\ -1 & 3 \end{bmatrix}, \tag{3.35}$$

$$A = \begin{bmatrix} 3 & 1 \\ 0 & 3 \end{bmatrix}, \qquad U = \begin{bmatrix} 1 & -1 \\ -\dfrac{3}{2} & \dfrac{3}{2} \end{bmatrix}. \tag{3.36}$$

第二个样本的第一隐藏单元处在 ReLU 负区间，因此输出为零．定义 $a = (1+e^{2})^{-1} \approx 0.119203$，$b = (1+e^{3})^{-1} \approx 0.047426$，则

$$Q = \begin{bmatrix} 1-a & a \\ b & 1-b \end{bmatrix} \approx \begin{bmatrix} 0.880797 & 0.119203 \\ 0.047426 & 0.952574 \end{bmatrix}. \tag{3.37}$$

取批量平均交叉熵，有

$$\mathcal{L} = -\frac{1}{2}\big(\log(1-a) + \log(1-b)\big) = \frac{1}{2}\big(\log(1+e^{-2}) + \log(1+e^{-3})\big) \approx 0.087758. \tag{3.38}$$

至此，损失值、所有中间量和参数均来自同一次前向计算，反向过程应保持这组数值不变．

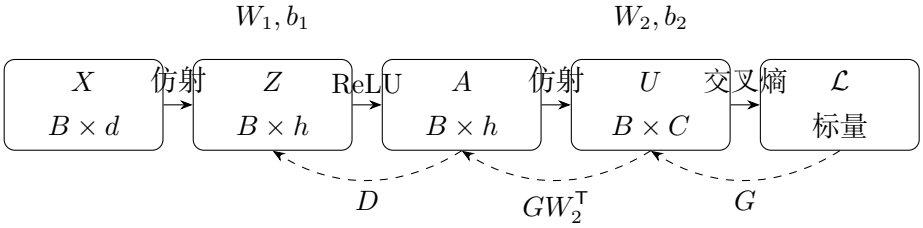


图 3.2: 两层网络的前向与反向路径．图中 $G = \overline{U}$，$D = \overline{Z}$；各仿射层还分别产生权重与偏置梯度．

由交叉熵的推导，首先得到

$$G = \overline{U} = \frac{Q - Y}{2} = \frac{1}{2}\begin{bmatrix} -a & a \\ b & -b \end{bmatrix}. \tag{3.39}$$

输出层参数梯度为

$$\overline{W}_2 = A^{\mathsf{T}}G = \begin{bmatrix} -\dfrac{3a}{2} & \dfrac{3a}{2} \\ \dfrac{-a+3b}{2} & \dfrac{a-3b}{2} \end{bmatrix} \approx \begin{bmatrix} -0.178804 & 0.178804 \\ 0.011537 & -0.011537 \end{bmatrix}, \tag{3.40}$$

$$\overline{b}_2 = G^{\mathsf{T}}\mathbf{1}_2 = \begin{bmatrix} \dfrac{b-a}{2} \\ \dfrac{a-b}{2} \end{bmatrix} \approx \begin{bmatrix} -0.035889 \\ 0.035889 \end{bmatrix}. \tag{3.41}$$

在计算隐藏激活的梯度时仍使用原来的 W_2：

$$\overline{A} = GW_2^{\mathsf{T}} = \frac{1}{2}\begin{bmatrix} -a & a \\ b & -b \end{bmatrix}. \tag{3.42}$$

这里 $\overline{A}$ 恰好与 G 数值相同，是本例所选权重的结果，不是一般恒等式．ReLU 的局部导数矩阵为 $\mathbf{1}[Z>0] = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$，故

$$D = \overline{Z} = \overline{A} \odot \mathbf{1}[Z>0] = \frac{1}{2}\begin{bmatrix} -a & a \\ 0 & -b \end{bmatrix}. \tag{3.43}$$

第二个样本的第一隐藏单元反向被阻断，这与其前向处于负区间相对应．

继续使用式 (3.14) 可得

$$\overline{W}_1 = X^{\mathsf{T}}D = \begin{bmatrix} -\frac{a}{2} & \frac{a}{2}+b \\ -a & a-\frac{b}{2} \end{bmatrix} \approx \begin{bmatrix} -0.059601 & 0.107027 \\ -0.119203 & 0.095490 \end{bmatrix}, \tag{3.44}$$

$$\bar{b}_1 = D^{\mathsf{T}}\mathbf{1}_2 = \begin{bmatrix} -\frac{a}{2} \\ \frac{a-b}{2} \end{bmatrix} \approx \begin{bmatrix} -0.059601 \\ 0.035889 \end{bmatrix}, \tag{3.45}$$

$$\overline{X} = DW_1^{\mathsf{T}} = \begin{bmatrix} -a & 0 \\ \frac{b}{2} & -\frac{b}{2} \end{bmatrix}. \tag{3.46}$$

输入梯度虽然不是这里需要更新的参数，却仍有明确含义：它描述当前损失对输入特征的局部敏感度．若 X 来自更早的可训练模块，这个梯度正是该模块收到的上游信息．

以 $W_{1,11}$ 为例，其梯度为 $-\frac{a}{2} \approx -0.059601$．仅对这个坐标施加正扰动 δ 时，一阶损失变化约为 -0.059601δ．如果所有参数沿负梯度作足够小的共同更新，则一阶变化为 $-\eta\|\nabla_\theta \mathcal{L}\|^2$．这只是在可微邻域中的局部判断；过大的步长可能跨过激活边界或使高阶项占据主导，因此不能从梯度符号直接推断任意步长都会降低损失．

本例完成了从输入、两次仿射变换、非线性和概率归一化，到损失以及全部参数梯度的闭合计算．每个矩阵均可用形状、逐元素求导和数值扰动交叉核对，三种证据互相补充．

3.7 广播、归约及嵌入查表

3.7.1 广播运算的反向归约

广播（broadcasting）把长度为一或缺失的轴按规则扩展，使同一个参数参与多个位置的运算. 其本质是参数共享，不必理解成实际复制数据. 对 $Z_{ij} = X_{ij} + b_j$，有

$$\overline{b}_j = \sum_i \overline{Z}_{ij}, \qquad \overline{X}_{ij} = \overline{Z}_{ij}. \tag{3.47}$$

若 $Z_{itj} = X_{itj} + b_j$，则 b_j 同时沿样本轴和位置轴广播，故

$$\overline{b}_j = \sum_{i=1}^{B} \sum_{t=1}^{T} \overline{Z}_{itj}. \tag{3.48}$$

反向后应恢复偏置原来的形状，而不是保留扩展后的每个位置一份梯度.

例如 $B = T = 2$，某一通道的上游梯度在四个位置分别为 $1, 2, 3, 4$，共享偏置收到的梯度是 10. 如果前向还有一个在四个位置上取平均的损失，其 $\dfrac{1}{4}$ 系数应已通过上游梯度传来；不能在偏置反向时再无条件除以四.

反过来，归约求和 $s_j = \sum_i X_{ij}$ 的反向将 $\overline{s}_j$ 广播回每一行；均值 $s_j = B^{-1} \sum_i X_{ij}$ 的反向还要乘以 $\dfrac{1}{B}$. 广播与归约之所以互为对应，是因为它们分别表达共享和聚合.

对带掩码的损失，设 $M_{it} \in \{0, 1\}$ 表示有效位置，$N = \sum_{it} M_{it} > 0$，则

$$\mathcal{L} = \frac{1}{N} \sum_{it} M_{it} \ell_{it}, \qquad \overline{U}_{itj} = \frac{M_{it}}{N} (Q_{itj} - Y_{itj}). \tag{3.49}$$

分母是有效位置数，而不一定是填充后张量的元素数. 掩码固定时此式成立；若权重本身由可训练模块生成，分母和权重的求导需要另行处理.

3.7.2 嵌入查表的稀疏梯度

设词表大小为 V，嵌入矩阵为 $E \in \mathbb{R}^{V \times d}$. 编号 k 的独热向量 e_k 满足 $(e_k)_j = \mathbf{1}[j = k]$，因此

$$e_k^{\mathsf{T}} E = E_{k,:}. \tag{3.50}$$

对一批编号 $k_1, \ldots, k_B$，令选择矩阵 $S \in \{0, 1\}^{B \times V}$ 的第 i 行为 $e_{k_i}^{\mathsf{T}}$，则嵌入输出为 $H = SE$. 根据线性层的求导结果，

$$\overline{E} = S^{\mathsf{T}} \overline{H}, \qquad \overline{E}_{j,:} = \sum_{i : k_i = j} \overline{H}_{i,:}. \tag{3.51}$$

选择矩阵仅用于论证；实际计算可以直接按编号访问参数行，避免构造巨大的独热数组.

设编号为 $(2,1,2)$，按本章从一开始的行编号，三个输出的梯度分别为 $(1,2)$、$(3,4)$、$(5,6)$. 则第 1 行的梯度为 $(3,4)$，第 2 行为 $(6,8)$，未被选中的其他行在这条查表路径上的梯度为零. 第 2 行必须把两次出现的贡献相加，而不是用最后一次覆盖前一次. [3]

查表不提供对整数编号本身的通常导数；它对被选取的连续参数可微. 若嵌入表还被其他路径使用，例如同时作为输出投影的权重，那么那些路径的梯度也要累加. 即使某一行的数据梯度为零，动量状态或权重衰减仍可能使优化步骤改变该行，因此不能把“本批未出现”直接等同于“参数绝不变化”.

对长度为 T 的序列，显式独热表示需要 BTV 个元素，编号只需要 BT 个整数；查表后的连续表示需要 BTd 个元素. 这种计算表示的变化不改变数学含义，却决定了大词表输入是否具有可行的存储成本.

3.8　有限差分及梯度核验

3.8.1　中心差分的误差来源

将全部参数按固定顺序看作向量 θ，选取单位方向 v，定义单变量函数 $f(t)=\mathcal{L}(\theta+tv)$. 若该方向邻域内具有足够光滑的三阶导数，则 Taylor 展开给出

$$\frac{\mathcal{L}(\theta+\varepsilon v)-\mathcal{L}(\theta-\varepsilon v)}{2\varepsilon}=v^{\mathsf{T}}\nabla_{\theta}\mathcal{L}+O(\varepsilon^{2}). \tag{3.52}$$

中心差分消去了展开中的偶次项，截断误差为二阶. 但当 ε 很小时，两个接近的损失值相减会放大舍入误差；若损失求值的绝对误差量级为 uM，差商中的该项约为 $O(\frac{uM}{\varepsilon})$，其中 u 为机器精度尺度，M 为求值的相关尺度.

因此步长不是越小越好. 应使用足够精细的数值精度，在一组合理的步长上观察误差先下降、随后受舍入影响的区间. 根据上述理想误差模型平衡两项时，步长呈 $u^{\frac{1}{3}}$ 的尺度关系，但实际常数还取决于参数尺度、损失尺度和三阶导数；不能将它作为对所有模型固定适用的数值.

只扰动上述两层网络中的 $W_{1,11}$，将扰动记为 t. 当 $|t|<\dfrac{1}{2}$ 时，四个隐藏预激活仍保持原来的正负状态，因此 ReLU 分支不变. 此时两个正确类别的分数差分别为 $2+t$ 与 3，损失可以直接化为

$$f(t)=\frac{1}{2}\log(1+e^{-2-t})+\frac{1}{2}\log(1+e^{-3}). \tag{3.53}$$

由这个独立的标量表达式可得

$$f'(0)=-\frac{1}{2(1+e^{2})}=-0.059601461\ldots, \tag{3.54}$$

[3]程序中的词元编号常从零开始，书中矩阵行号从一开始. 转换时应整体调整编号约定，不应只修改一处索引.

与矩阵反向结果一致. 取 $\varepsilon = 10^{-4}$, 式 (3.52) 同样给出约 -0.059601461. 这一核对有意义, 是因为它绕过了矩阵反向程序, 直接从扰动后的损失计算变化.

3.8.2 梯度一致性的判定边界

梯度校验应固定输入、标签、参数及随机选择, 并避免扰动跨越 ReLU 折点或离散分支. 若原点本身位于不可微处, 中心差分可能给出左右变化率的平均值, 它并不等于实现选定的次梯度. 随机失活若在两次求值时使用不同掩码, 差商还会混入函数本身变化造成的噪声.

比较时应同时报告绝对误差和相对尺度. 例如令 a 为解析方向导数, n 为数值差商, 可考察 $|a-n|$, 并辅以 $\dfrac{|a-n|}{\max(\delta, |a|+|n|)}$, 其中 $\delta > 0$ 避免近零分母. 只比较相对误差会对接近零的导数过度敏感, 只比较绝对误差又可能掩盖大尺度量上的错误.

通过校验说明当前输入和参数附近的计算图与导数相容, 不能证明类别标签正确、样本划分合理, 或损失符合业务目标. 尤其是错误地沿样本轴执行 Softmax, 仍然定义了一个可微函数, 自动微分可以正确求出这个错误函数的梯度. 因此, 可靠检查应先确认轴与任务语义, 再检查分量公式和数值导数, 最后才讨论训练效果.

第四章　文本表示

语言模型不能直接对一段可见文字执行矩阵乘法. 原始字节必须先被解释为有边界的文本记录，再按任务规则形成样本，经规范化、分词、特殊词元插入和长度处理，最终成为整数张量. 只有先说明这些对象，后续的嵌入、注意力和训练目标才有明确输入.

本章是全书数据主线的第一部分，处理理解模型计算所必需的输入语义：文本长什么样、样本边界在哪里、词元编号如何产生，以及变长序列如何组成批次. 第十八章《训练数据工程基础》是第二部分，再讨论大规模采集、治理、去重、混合、分片和可恢复生产. 前一部分回答“模型接收什么”，后一部分回答“这些输入如何被可靠地生产出来”.

文本编码决定序列长度、稀有字符串的表示方式、模型输出空间以及对话边界. 本章区分字符、字节和词元，再讨论基于合并规则、最长匹配和概率分段的三类方法；同时保留从 N-gram、词向量到循环网络的必要背景，用来说明序列表示为何逐步转向注意力. 所有手算例均为明示的小型构造，用于检验数学过程，不代表任何现有模型的实际词表或性能.

4.1　模型输入构造

4.1.1　文本边界

磁盘中的文本首先是按某种字符编码保存的字节，但训练器通常不直接把整个文件当作一个样本. 数据还需要区分至少四个层次：**原始资料**是抓取文件、文档或消息快照；**逻辑记录**保留一篇文章、一轮对话或一组问答的语义结构；**训练样本**是按特定任务从记录中选择或切出的消费单位；**模型序列**则是加入边界符号、完成截断或填充后的词元编号. 一个对象在上一层完整存在，不意味着它在下一层仍保持相同边界.

例如，同一段包含标题、两个段落和来源字段的 JSONL 记录，可以作为一条完整文档参加预训练，也可以按段落切成两条样本；用于监督微调时，它还可能被转换成带角色边界的消息序列. 三种处理读取了相同文字，却定义了不同的条件上下文、结束位置和有效标签. 换行是否保留、空文档是否丢弃、相邻记录之间是否插入 EOS，都不是分词器能够自行决定的事实.

层次	典型对象	本层必须明确的决定
存储层	UTF-8 文件、JSONL、Parquet	字节编码、字段与记录边界
语义层	文档、消息序列、问答对	字段含义、顺序与分组关系
样本层	文本窗口、提示—回答、句对	任务边界、截断与监督区域
模型层	词元编号、特殊词元、有效性掩码	词表版本、长度、填充及位置来源

这里先固定一个重要边界：本章只建立单条记录如何成为模型输入的最小语义，不展开来源许可、海量去重、混合比例、分片发布和删除传播. 那些问题不影响读者理解一次分词和一次注意力计算，却会决定真实训练集的分布与可追溯性，因而集中留到第十八章《训练数据工程基础》.

4.1.2 最小输入管线及张量形状

设第 b 条原始文本为 $s^{(b)}$，规范化函数为 N，固定分词器为 T. 编码结果是长度可变的编号序列

$$s^{(b)} \xrightarrow{N} \widetilde{s}^{(b)} \xrightarrow{T} x^{(b)}_{1:L_b}, \qquad x^{(b)}_t \in \{0, \dots, V-1\}. \tag{4.1}$$

任务模板可以在编码前组织结构，也可以由后处理器插入 BOS、EOS、SEP 等特殊词元. 随后按明确策略截断，并将同一批次补齐到 $T_{\max}$，得到编号张量 $K \in \{0, \dots, V-1\}^{B \times T_{\max}}$ 和有效性掩码 $M \in \{0,1\}^{B \times T_{\max}}$. $M_{bt} = 1$ 表示该位置来自样本协议中的真实输入，零表示为组成矩形张量而添加的填充位置.

本章后文将通过嵌入矩阵把 K 转换为连续张量 $X \in \mathbb{R}^{B \times T_{\max} \times d}$；第五章《注意力机制》再在允许的位置之间以注意力机制读取这些向量. 因此，从字符串到注意力输入的主线是

$$\text{文本记录} \longrightarrow \text{训练样本} \longrightarrow \text{词元编号与掩码} \longrightarrow \text{连续向量} \longrightarrow \text{上下文交互}. \tag{4.2}$$

若记录边界、特殊词元或填充位置在前两步已经错误，后续张量即使形状合法，也只会精确计算一个定义错误的任务.

两层数据问题

输入表示层关心一条样本如何变成可解释的编号、掩码和向量，是理解模型结构的前提；训练数据工程层关心大量异构来源如何经过治理与生产形成可复现的数据版本，是理解训练分布的前提. 二者共享“样本边界”和“词元计量”等接口，但不应混成同一章的同一抽象层.

4.2　神经语言模型

4.2.1　N-gram 用有限上下文估计序列概率

自然语言处理（Natural Language Processing，NLP）研究如何让计算系统表示、理解和生成语言．早期统计语言模型常作有限记忆假设，用最近 $n-1$ 个词预测下一个词：

$$p(w_1,\dots,w_T)\approx\prod_{t=1}^{T}p(w_t\mid w_{t-n+1},\dots,w_{t-1}). \tag{4.3}$$

这类模型称为**N-gram 语言模型**（N-gram Language Model）．二元模型只看前一个词，三元模型看前两个词．条件概率可由计数估计，但未见组合会得到零概率，因此需要加法平滑、回退或插值．n 增大能保留更长的局部模式，同时使组合数量迅速增加，数据稀疏和存储成本也随之加重．

N-gram 的价值不只在历史意义．它明确展示了语言模型的核心任务：根据上下文分配下一词的概率；也展示了有限上下文、稀疏计数和泛化之间的矛盾．神经语言模型改变了概率参数化方式，但没有改变条件预测这一目标．

4.2.2　离散符号的连续嵌入

独热向量能区分词的身份，却不能直接表达词之间的相似性．词向量为每个词分配一个低维可学习向量．早期方法可以从词与上下文的共现统计中分解矩阵，也可以通过预测上下文或中心词学习表示．训练完成后，语义或句法关系可能表现为向量距离和方向上的规律．

词向量的含义来自训练目标和语料，不能把几何接近直接解释为事实相同．传统静态词向量为同一个词提供固定表示，无法随句子消解一词多义；后来的上下文表示会根据整段输入产生不同向量．本章后文将从嵌入查表继续讨论离散身份的连续表示，第五章《注意力机制》和第七章《位置表示》再分别建立注意力计算与位置表示．

4.3　神经序列模型

4.3.1　前馈网络依赖固定宽度输入

前馈神经网络（Feedforward Neural Network，FNN）中的信息只从输入流向输出，不保存跨步状态．用于语言建模时，可以把固定窗口内的若干词向量拼接或汇总，再预测下一个词：

$$h=\phi(W_x[x_{t-k};\dots;x_{t-1}]+b),\qquad p(w_t\mid w_{t-k:t-1})=\mathrm{softmax}(W_o h+c). \tag{4.4}$$

与 N-gram 相比，不同上下文通过共享词向量和网络参数获得统计共享，未见过的完整组合也可能得到合理预测．但窗口长度仍需预先固定，超过窗口的信息无法进入当前预测．

4.3.2 RNN 用隐藏状态汇总历史

循环神经网络（Recurrent Neural Network，RNN）在每一步更新隐藏状态：

$$h_t = \phi(W_x x_t + W_h h_{t-1} + b), \qquad y_t = W_o h_t + c. \tag{4.5}$$

h_t 是对截至位置 t 的历史摘要，同一组参数在所有时间步共享，因此模型可以处理变长序列．训练时把循环按时间展开，再通过时间反向传播计算梯度．虽然依赖路径原则上可以很长，反复乘以状态雅可比会使梯度指数衰减或增长，造成梯度消失或爆炸；隐藏状态容量也限制了远距离信息的保留．

4.3.3 LSTM 用门控制记忆读写

长短期记忆网络（Long Short-Term Memory，LSTM）在隐藏状态之外维护记忆单元 c_t，并用门控制遗忘、写入和输出：

$$f_t = \sigma(W_f[x_t; h_{t-1}] + b_f), \qquad i_t = \sigma(W_i[x_t; h_{t-1}] + b_i), \tag{4.6}$$

$$\widetilde{c}_t = \tanh(W_c[x_t; h_{t-1}] + b_c), \qquad c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{c}_t, \tag{4.7}$$

$$o_t = \sigma(W_o[x_t; h_{t-1}] + b_o), \qquad h_t = o_t \odot \tanh(c_t). \tag{4.8}$$

门向量的元素位于 $(0, 1)$，可逐维决定保留多少旧记忆、写入多少新信息．记忆单元中的加性路径缓解了普通 RNN 的长期梯度问题，但不能保证任意长依赖都被保存．RNN 和 LSTM 还必须按时间顺序计算，难以像 Transformer 那样同时处理所有位置．后续的注意力机制通过显式读取多个位置，改变了信息交互和并行计算方式．

4.4 文本表示的层次

4.4.1 字符、码点、字节及词元

日常所说的“一个字符”有时指一个可见字形，有时指一个 Unicode 码点，两者并不总是一一对应．例如，带重音的字母可以由单一码点表示，也可以由基本字母与组合标记共同表示．字素簇描述更接近读者感知的文本单位；码点则是字符编码标准中的编号．讨论字符串长度时必须说明采用哪一种计数方式．[1]

[1] 一个表情符号也可能由多个码点组成．光标移动、字符高亮和模型分词分别服务不同目的，不能假定它们具有相同边界．

字节是八位存储单元，具有 256 种可能取值. UTF-8 将 Unicode 标量值编码为一个到四个字节. 例如，汉字“中”的码点是 `U+4E2D`，UTF-8 表示为十六进制字节 `E4 B8 AD`. 这一个汉字在字节级模型中最初对应三个基本单元；经过子词合并后，可能对应一个、两个或三个词元.

词元（Token）是模型词表中的离散单位，可以对应完整单词、子词、单个字符、字节片段或者结构控制符号. 子词 (Subword) 强调单位可能小于通常意义上的词，不保证每一项都是语言学上的词根或词缀. 中文不使用空格分隔全部词语，程序代码、数字和混合语言文本也具有不同结构，因而不能将“按空格切分”视为普遍适用的文本表示.

设普通词表为 $\mathcal{V}_{\text{text}}$，特殊词元集合为 $\mathcal{V}_{\text{special}}$，完整词表为它们的不交并 $\mathcal{V}$，大小为 V. 每个词元对应唯一编号，通常选自 $\{0,\dots,V-1\}$. 编号只指定参数矩阵的行，不表达数量大小或语义距离. 编号 12 与 13 相邻，不意味着对应词元比编号 12 与 900 更相似.

4.4.2　变长映射及符号约定

对合法文本集合 $\mathcal{S}$，确定性编码器写为

$$T:\mathcal{S}\longrightarrow\bigcup_{L\geqslant 0}\{0,\dots,V-1\}^{L},\qquad T(s)=(x_1,\dots,x_{L(s)}).\tag{4.9}$$

这里 $L(s)$ 随输入变化，不能在定义域之外将它视为固定长度. 解码器 D 将词元序列恢复为文本或字节. 为了讨论原文可逆性，本章默认普通文本编码时不额外插入控制词元；加入控制词元后的序列需要按明确协议解释.

符号	含义
s；$N(s)$	原始文本；规范化后的文本.
$\mathcal{V},V$	完整词表及其大小，包含实际可用的新增词元.
$x_{1:L}$	词元编号序列，L 为词元数.
$b(v)$	普通词元 v 所代表的字节串.
$r(a,b)$	已训练 BPE 规则中符号对 (a,b) 的合并优先级.
$\mathcal{Z}(s)$	能够完整拼接为文本 s 的候选分段集合.
$q(v)$	Unigram 分词模型为词元 v 指定的概率参数.
$B,T_{\max},d$	批大小、填充后长度及模型隐藏维数.

若嵌入矩阵为 $E\in\mathbb{R}^{V\times d}$，则词元序列的连续表示为 $H_i=E[x_i]$，形状为 $[L,d]$. 将词元编号写成独热行向量 $e_{x_i}^{\mathsf{T}}$，同一操作可写为 $H_i=e_{x_i}^{\mathsf{T}}E$. 查表实现避免显式构造稀疏独热矩阵；其数学语义仍然是选择对应的行. 批处理时输入编号为 $[B,T_{\max}]$，嵌入输出为 $[B,T_{\max},d]$.

4.4.3　编码流程

一个分词系统通常包含规范化、预分词、词元模型、后处理和解码五类职责．规范化决定哪些文本差异被保留；预分词决定哪些位置允许发生子词组合；词元模型选择切分；后处理加入任务边界；解码将普通词元还原为文本．不同实现可以合并其中若干步骤，但不能省略对这些语义的说明．

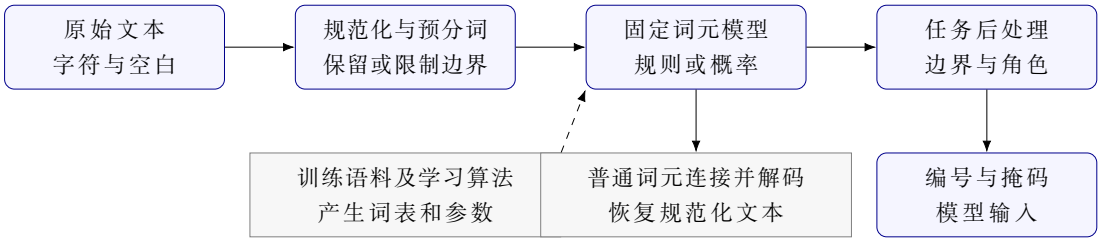


图 4.1: 文本编码与模型协议的分层结构．训练过程产生词表及算法参数；运行过程使用固定制品编码．

例如，空格可以作为预分词边界，也可以被保留为词元内容或边界标记．若预处理只保留切出的非空片段而丢弃所有间隔，连续空格、换行与制表符便无法恢复．规范化规则和分词算法的选择共同决定系统行为，算法名称相同不代表整数序列相同．

4.5　字节对编码

4.5.1　训练目标及计数对象

字节对编码（Byte Pair Encoding，BPE）通过逐次合并相邻基本符号构造词表．将这一思想用于子词表示，是处理未登录词的一种经典方法 [18]．现代字节级变体以字节为基础符号；字符级变体则以字符为基础．算法名称中的“字节”不能代替对初始字母表的实际说明．

设训练语料经预处理后包含片段 $s^{(1)},\dots,s^{(m)}$，片段权重为正整数 $w_1,\dots,w_m$，表示出现次数．在第 k 轮，片段的当前表示为符号序列 $s^{(j,k)}$．相邻对计数为

$$c_k(a,b)=\sum_{j=1}^{m} w_j \sum_{i=1}^{|s^{(j,k)}|-1} \mathbf{1}[s_i^{(j,k)}=a,\ s_{i+1}^{(j,k)}=b]. \tag{4.10}$$

内层遍历一个片段内部的相邻位置，外层累加全语料权重．预分词边界两侧的符号不参与同一次计数．选出最大计数的候选对后，将其非重叠出现替换为新符号，更新语料表示，再重新统计．词表达到目标大小，或不存在合法合并时，训练停止．

这里的目标是基于局部频率逐步缩短序列，并非直接最大化下游语言模型的似然，也不保证得到全局最优压缩词表．训练后的规则必须保留先后顺序．若第 k 轮学

习 $(a,b)\mapsto ab$，可记其优先级为 $r(a,b)=k$；数值越小表示越早执行．词元编号与规则优先级是不同概念，有些格式将它们结合存储，有些格式分别存储．

算法 4.1　BPE 词表训练

输入：预分词后的加权片段、基础符号集合、目标词表大小及确定的平局规则．**输出：**词表、编号映射与按学习顺序排列的合并规则．

1. 将每个片段展开为基础符号序列，保留片段边界；初始化规则表为空．
2. 若已达到目标词表大小，停止．否则根据式 (4.10) 统计当前片段内全部合法相邻对．
3. 若没有合法候选，停止．按频数及平局规则选择一对，登记其合并规则和新符号．
4. 在每个片段内从左到右替换该对的非重叠出现，保留权重，返回第 2 步．

合并不改变片段所代表的字节串．基础字节覆盖不因某轮未出现而删除．朴素重计数每轮扫描当前语料，若初始总符号数为 N、执行 M 次合并，扫描成本上界为 $O(MN)$；局部计数索引可降低重复扫描，但必须保持相同规则．

取三个训练片段：`abab` 出现两次，`abac` 出现一次，`ac` 出现一次．以 a,b,c 为初始符号，不允许跨片段合并．初始加权总长度为

$$N_0=2\times 4+1\times 4+1\times 2=14. \tag{4.11}$$

初始相邻对中，(a,b) 在 `abab` 内出现两次，考虑权重后贡献四次；在 `abac` 中再贡献一次，因此总计为五．相似地，(b,a) 计数为三，(a,c) 计数为二．第一轮合并 $(a,b)\mapsto ab$，得到

$$(ab,ab)\times 2,\qquad (ab,a,c)\times 1,\qquad (a,c)\times 1. \tag{4.12}$$

总长度变为 $N_1=2\times 2+3+2=9$，减少五，与本轮五次非重叠替换一致．

第二轮中，(ab,ab) 与 (a,c) 的计数都为二，(ab,a) 的计数为一．为使过程确定，规定比较两个符号所代表的字节串，按有序对的字典序处理平局；由此本例选择 (a,c)，得到

$$(ab,ab)\times 2,\qquad (ab,ac)\times 1,\qquad (ac)\times 1. \tag{4.13}$$

总长度变为 $N_2=7$．第三轮选择计数为二的 (ab,ab)，得到 $(abab)\times 2$、$(ab,ac)\times 1$、$(ac)\times 1$，于是 $N_3=5$．

轮次	选中规则	频数	合并后加权词元数
1	$(a,b)\mapsto ab$	5	9
2	$(a,c)\mapsto ac$	2	7
3	$(ab,ab)\mapsto abab$	2	5

本例只为三个基础符号增加三个合并词元，并未覆盖全部字节．如果改为完整字节级初始词表，未在训练样例出现的其他基础字节仍需保留．这一区别决定未见字符能否表示，不能从一个英文手算例推断完整输入覆盖性．

4.5.2 分词器推理规则

对新片段 `ababac`，训练好的规则按既定优先级执行：

$$(a, b, a, b, a, c) \longrightarrow (ab, ab, a, c) \tag{4.14}$$

$$\longrightarrow (ab, ab, ac) \tag{4.15}$$

$$\longrightarrow (abab, ac). \tag{4.16}$$

第一步应用优先级一，第二步应用优先级二，第三步应用优先级三．编码时不重新比较新文本中的频率，也不添加未训练的合并规则．否则相同字符串在不同批次或不同上下文中可能获得不同编号，模型参数所对应的表示协议随之改变．

BPE 编码也不是简单地从最终词表选择最长前缀．设另一套有效规则依次为 $(b, c) \mapsto bc$、$(a, b) \mapsto ab$．对输入 `abc`，BPE 先得到 (a, bc)，不能再合并；若只看词表并优先选最长前缀，则会得到 (ab, c)．两种结果都能还原原文，却给模型提供不同的输入．因此，无损解码不足以证明两种编码实现等价．

4.5.3 分词合并不变量

对 (a, a, a)，相邻对 (a, a) 的位置计数为二，但两次出现共享中间符号，无法同时替换．规定从左向右处理后，一轮替换结果为 (aa, a)，长度只减少一．由此可见，频数与实际压缩量并非在所有情况下相等．对于自重叠候选，应明确计数方式与非重叠替换规则，不能将式 (4.10) 直接当作精确压缩收益．

训练平局需要稳定的规则；编码平局还涉及同一合并在不同位置重叠的处理顺序．实现可以采用高效数据结构维护局部候选，但优化不能改变这两种语义．最直接的正确性不变量是

$$b(v_1^{(k)}) \| \cdots \| b(v_{L_k}^{(k)}) = \mathrm{UTF8Encode}(s), \qquad L_{k+1} < L_k \tag{4.17}$$

对每一次实际合并均成立，其中 $\|$ 表示字节串连接．它证明每轮只删除边界而不改变内容．频率降低、语义改善或下游模型性能提升都不是这一不变量的结论．

4.6 WordPiece 分词模型

4.6.1　词首及续接片段

WordPiece 同样使用子词词表，但常见 BERT 编码器按从左到右的最长匹配执行切分 [19]．词首片段和词内续接片段可使用不同词表项，续接项常以 `##` 标记．这个标记是制品约定，不是原文中的两个井号．

设词表包含 `play`、`player`、`##ing`、`##s`．对预分词后的 `playing`，从词首尝试能够匹配的最长片段，找到 `play`，剩余 `ing` 使用续接词元 `##ing`．对 `players`，词首直接选择更长的 `player`，再选择 `##s`．已经选定的前缀不因后续概率更高而自动回退．

这个最后的性质容易被忽略．取词表 $\{a, ab, \texttt{\#\#bc}\}$，输入 `abc`．最长匹配首先选择 ab，但剩余 c 没有续接项；原始 BERT 的这一路径会把整个词替换为未知词元，而不会回退到 $(a, \texttt{\#\#bc})$．尽管后一条分段存在，它不是该贪心算法自动搜索的结果．超过实现规定的最大词长也可能触发未知词元，属于应记录的实现边界．

4.6.2　WordPiece 训练边界

词表训练决定保留哪些片段，最长匹配决定给定词表后怎样编码．这是两个不同问题．WordPiece 的历史训练方法与似然改进有关，但不同实现的候选构造、打分和停止规则并不完全相同．仅凭 BERT 发布的词表和编码源文件，无法还原生成该词表的全部训练过程．

一些教学实现使用类似

$$\frac{c(a,b)}{c(a)c(b)} \tag{4.18}$$

的关联分数选择合并，它倾向于突出相对于边缘频率更紧密的组合．然而，这一表达式并不是所有 WordPiece 实现统一遵守的训练定义，也不能被直接写成某个既有词表的原始训练算法．本章因此分别陈述可核实的编码规则与训练问题，不以一个便于演示的分数替代未公开的细节．

BPE 保存有序合并规则，WordPiece 的常见编码路径依赖最终词表及最长匹配约定．二者都可以产生类似的子词，但排序语义不同．重新训练任一分词器都可能改变词元编号；不能因为算法仍叫 BPE 或 WordPiece，就沿用原模型嵌入矩阵而不检查对应关系．

4.7　Unigram 分词模型

4.7.1　候选集合构造

Unigram 分词模型为候选词元赋予概率，并在能够拼接出原文本的分段中选择或采样 [20]．与逐次合并不同，它允许同一文本同时存在多条有权重的分段路径．下面用独立构造的有限例子推导这一计算过程．

设普通词表中的每个非空词元 v 具有参数 $q(v) > 0$，且 $\sum_{v} q(v) = 1$. 对分段 $z = (z_1, \ldots, z_K) \in \mathcal{Z}(s)$，定义权重

$$w_q(z) = \prod_{i=1}^{K} q(z_i), \qquad Z_q(s) = \sum_{z \in \mathcal{Z}(s)} w_q(z). \tag{4.19}$$

给定字符串后的分段分布为

$$P_q(z \mid s) = \frac{w_q(z)}{Z_q(s)}. \tag{4.20}$$

分母只汇总能够重构同一字符串的路径. 词元概率之积体现一元独立假设，但约束“必须拼出 s”使候选位置之间并非可以任意独立选择. [2]

4.7.2 分段及数值比较

取词表 $\{a, b, ab, ba\}$，分别赋值 $0.2, 0.2, 0.4, 0.2$. 对字符串 `aba`，全部候选分段如下：

分段	权重计算	给定字符串的概率
(ab, a)	$0.4 \times 0.2 = 0.08$	$\dfrac{0.08}{0.128} = 0.625$
(a, ba)	$0.2 \times 0.2 = 0.04$	$\dfrac{0.04}{0.128} = 0.3125$
(a, b, a)	$0.2 \times 0.2 \times 0.2 = 0.008$	$\dfrac{0.008}{0.128} = 0.0625$

最可能分段是 (ab, a). 若将参数改为 $q(a) = q(b) = 0.3$、$q(ab) = 0.1$、$q(ba) = 0.3$，则三条路径权重分别变为 $0.03, 0.09, 0.027$，最优分段成为 (a, ba). 词表没有改变，概率变化却改变了分段结果；只保存词表不能完整描述 Unigram 编码器.

4.7.3 动态规划及 Viterbi 路径

直接枚举全部分段可能随长度指数增长. 利用每条路径最后一个词元的位置，可以建立动态规划. 令 $s_{1:n}$ 为长度 n 的基本字符序列，α_j 为前 j 个字符所有分段的总权重，初始化 $\alpha_0 = 1$. 则

$$\alpha_j = \sum_{\substack{0 \le i < j \\ s_{i+1:j} \in \mathcal{V}_{\text{text}}}} \alpha_i q(s_{i+1:j}), \qquad Z_q(s) = \alpha_n. \tag{4.21}$$

每一项对应以 $s_{i+1:j}$ 为最后一个词元的全部前缀路径. 不同最后词元的起点划分了候选集合，因此求和既不重复也不遗漏.

[2] 在本章，$Z_q(s)$ 用作分段边缘权重. 若要将其解释为所有任意长度文本上的正规概率，还需补充终止或长度生成机制；仅有 $\sum_{v} q(v) = 1$ 不足以规范化所有有限词元序列.

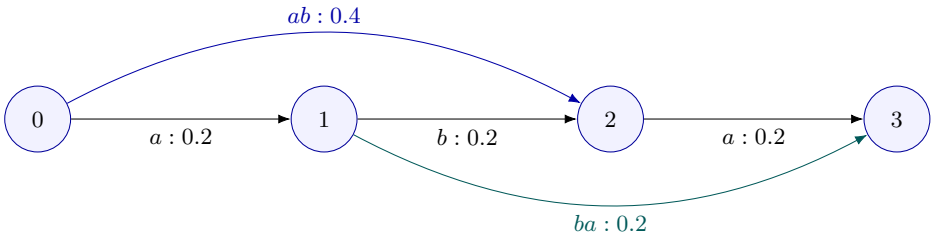
在 aba 的第一组参数下，

$$\alpha_1 = q(a) = 0.2, \tag{4.22}$$

$$\alpha_2 = \alpha_1 q(b) + \alpha_0 q(ab) = 0.04 + 0.4 = 0.44, \tag{4.23}$$

$$\alpha_3 = \alpha_2 q(a) + \alpha_1 q(ba) = 0.088 + 0.04 = 0.128. \tag{4.24}$$

这与逐项枚举一致．若最大词元长度为 $K_{\max}$，且候选匹配可以有效检索，则每个位置只需检查至多 $K_{\max}$ 个起点；动态规划的组合部分为 $O(nK_{\max})$，实际成本还取决于字符串匹配结构．



前向求和：$\alpha_3 = 0.128$　　最优路径：(ab, a)，权重 0.08

图 4.2: 字符串 aba 的分段格．每条从位置 0 到位置 3 的路径是一种分段；路径权重为边权之积．

求最优路径时，将求和改为取最大值，并在对数空间计算：

$$\delta_0 = 0, \qquad \delta_j = \max_{\substack{0 \leqslant i < j \\ s_{i+1:j} \in \mathcal{V}_{\text{text}}}} [\delta_i + \log q(s_{i+1:j})]. \tag{4.25}$$

同时保存取得最大值的起点，最后反向追溯即可恢复分段，这就是此处的 Viterbi 计算．前向求和在长序列上也应使用对数和指数技巧，避免很小的概率乘积下溢．最大路径权重与全部路径总权重不可混淆：前者用于确定性最优编码，后者用于边缘化与概率采样．

4.7.4　参数估计及词表裁剪

固定词表后，可最大化语料的分段边缘对数目标

$$\ell(q) = \sum_{j=1}^{m} \log Z_q(s^{(j)}). \tag{4.26}$$

分段是隐变量．期望最大化（Expectation–Maximization，EM）交替计算分段后验与词元的期望出现次数．令 $n_v(z)$ 表示词元 v 在路径 z 中出现的次数，则期望计数为

$$C_v = \sum_{j} \sum_{z \in \mathcal{Z}(s^{(j)})} P_q(z \mid s^{(j)}) n_v(z). \tag{4.27}$$

在无额外先验或平滑项的基本模型下，最大化 $\sum_v C_v \log q(v)$ 并满足 $\sum_v q(v) = 1$. 引入拉格朗日乘子并求导，得到

$$\frac{C_v}{q(v)} - \lambda = 0 \quad \Longrightarrow \quad q_{\text{new}}(v) = \frac{C_v}{\sum_u C_u}. \tag{4.28}$$

这一步是由带约束的最大化推导而来，实际训练器还可能加入先验、剪枝近似和保底符号限制.

只用前述 `aba` 一条样例，其三条后验概率为 $0.625, 0.3125, 0.0625$. 因此

$$C_a = 0.625 + 0.3125 + 2 \times 0.0625 = 1.0625, \tag{4.29}$$

$$C_b = 0.0625, \qquad C_{ab} = 0.625, \qquad C_{ba} = 0.3125. \tag{4.30}$$

期望总词元数为 2.0625，归一化后得到 $q_{\text{new}} = \dfrac{17, 1, 10, 5}{33}$. 这个例子同时表明 EM 计数是后验加权的分数计数，不是先选一条最佳分段再统计整数次数.

词表学习还需要决定保留哪些候选. 常见流程从较大的候选集合出发，估计概率，再评估移除词元带来的目标下降，逐步删除影响较小的项 [20]. 不能简单删除当前概率最小的词元：一个低频基础字符可能是某些输入唯一可用的表示，删除后这些输入的候选集合变为空集. 保留基础覆盖或指定回退机制，是剪枝约束的一部分.

可用同一例子理解删除代价. 暂时固定其他权重，只屏蔽含 ab 的路径，剩余总权重从 0.128 降为 0.048，对数目标下降 $\log(\dfrac{0.128}{0.048}) \approx 0.9808$；屏蔽 ba 时只降为 0.088，下降约 0.3747. 这只是固定参数下的路径敏感性，不是删除后重新归一化、重新估计参数的完整训练结果. 它解释了词元重要性与“为多少替代路径提供支持”有关.

4.7.5 概率分段及训练一致性

概率分词可以在语言模型训练时为同一文本提供不同表示，这种子词正则化利用分段变化增加输入多样性. 若服务端要求稳定编码，则通常采用固定最优路径，并明确训练与推理各自的策略. 采样种子、温度及候选限制会改变分段分布，不能在缓存键只记录原文时悄然改变这些设置.

SentencePiece 是可承载不同子词模型的工具系统，支持 BPE 和 Unigram 等方法，并包含规范化和空白表示规则 [21]. 因此，“采用 SentencePiece”没有完整回答分词算法是什么，还应说明模型类型与预处理配置；也不能由采用该工具就推断原文一定逐字节可逆.

4.8　可逆性、偏移及语言覆盖

4.8.1　字节覆盖边界

若普通词表包含所有 256 个基础字节，所有合并只连接字节而不删除内容，且前处理无损，则对合法 UTF-8 文本有

$$D(T(s)) = \mathrm{UTF8Decode}\big(b(x_1)\| \cdots \|b(x_L)\big) = s. \tag{4.31}$$

未见词或未见字符仍能分解为基础字节，因而不必因为词表缺少整词而使用未知词元. 但是，能够表示一种语言与善于表示这种语言不同：一个文本若大量退回单字节，序列可能明显增长，参数中也未必学习到充分的组合知识.

对“中”，若编码得到分别对应 `E4` 和 `B8 AD` 的两个词元，两个词元都不是独立完整的 UTF-8 字符. 对每个词元分别进行严格字符解码会失败；若使用替代字符继续解码，则原信息可能被破坏. 正确做法是先连接完整字节流，再解码. 流式显示同样需要保留尚未组成完整字符的尾部字节.

4.8.2　规范化的不可逆性

设 N 将大小写折叠，则 $N(\texttt{A}) = N(\texttt{a})$. 两种输入变成同一规范形式后，无论后续编码多么精确，都不能仅由编码结果判断原文是哪一种. 一般地，若存在 $s_1 \neq s_2$ 且 $N(s_1) = N(s_2)$，那么不保留额外信息的系统无法同时满足对两者的原文恢复.

因此规范化后可讨论的保证通常是

$$D(T(s)) = N(s), \tag{4.32}$$

而非式 (4.31). 统一兼容字符、折叠空白和清除控制字符各自具有用途，也各自删除不同信息. 代码缩进、数学符号、文件路径和专有名词可能依赖这些差异，规范化策略应由任务语义决定.

4.8.3　词元偏移及原文对齐

抽取式问答、命名实体识别和搜索高亮需要把词元位置映射回原文区间. 偏移应明确单位是字节、码点还是其他字符索引. 一个中文词元可能只含部分字节，却在用户界面对应同一个可见字符；规范化又可能让一个原字符映射到多个输出字符，或反过来. 因此偏移映射不一定是逐词元一一、不重叠的区间划分.

特殊词元没有原文来源位置，应使用约定的空区间或独立标志表示，不能把占位偏移当作真正指向文本开头. 解码输出中的空白清理也可能改变文本，故原文恢复检验必须记录清理和跳过特殊词元的选项. 可读的词表字符串只是检查视图，不能代替原始字节与位置协议.

4.9 词表规模、序列长度及计算代价

4.9.1 参数及计算的两种方向

扩大词表可能使高频片段由多个词元合为一个，减少序列长度；同时增加输入嵌入和输出投影的参数. 设隐藏维数为 d，不计输出偏置，输入和输出矩阵独立时需要约 $2Vd$ 个参数，权重共享时需要约 Vd 个. 全词表输出投影每个位置的乘加规模仍与 Vd 成正比，权重共享不会消除该计算.

例如，把词表从 32 000 增至 64 000，在 $d = 4096$ 且权重共享时，额外参数为

$$\Delta P = 32\,000 \times 4096 = 131\,072\,000. \tag{4.33}$$

若仅按每参数两字节存储，新增权重占用 262 144 000 字节，即 250 MiB. 训练还需梯度、优化器状态等额外内存，不能把这个权重存储数当作训练总显存.

另一方面，长度为 L 的密集注意力具有随 $L^2 d$ 增长的主要交互计算. 若同一文本的长度由 1000 降至 750，这部分规模的比例为 $(\frac{750}{1000})^2 = 0.5625$. 但若同时将词表翻倍，全词表输出投影的 LVd 规模比例为 $0.75 \times 2 = 1.5$. 这两个方向相反的结果说明，不能用序列压缩率直接预测端到端加速比.

4.9.2 训练分布及多语言公平比较

分词器训练语料的语言配比、领域配比与重复样本会影响词表容量分配. 高频模板可能消耗大量合并机会；数据量较少的语言可能被切得更碎. 相同词元预算下，不同语言能够放入上下文的原文长度因此不同. 数字按单个数字、连续数字块或其他边界切分，也会改变算术任务的表示难度，不能单凭某一种切分更短就判定其更优.

分词器训练应与留出评估分开，按文档或其他适当分组避免相近文本跨越边界. 评估至少分别观察各语言、领域和长度区间的词元数、字节回退、未知词元及长尾分布. 平均压缩良好却在关键少数语言上严重碎片化，仍可能不符合模型用途.

跨分词器比较每词元困惑度缺乏共同单位. 若在相同原始文本上统计负对数似然，可进一步按原始字节数归一化：

$$\mathrm{BPB} = \frac{-\sum_{t=1}^{L} \log p_\theta(x_t \mid x_{<t})}{N_{\mathrm{bytes}} \log 2}. \tag{4.34}$$

这里 BPB 表示每字节比特数，N_{bytes} 为指定文本编码下的原始字节数. 该量仍需统一文本、规范化、终止词元和上下文边界；若多个词元序列可解码为同一字符串，单一路径分数也未必等于字符串全部概率质量. 归一化提供了共同计量单位，并没有自动解决所有模型可比性问题.

4.10　特殊词元及序列协议

4.10.1　角色、字面形式、编号和行为

特殊词元（Special Token）承担控制职责，但其语义由训练和执行协议共同建立. 应分别识别四个对象：词元的角色、配置中的字面形式、实际编号以及运行时行为. 登记一个字符串为 EOS 不会自动教会未训练的模型何时结束，设置 BOS 属性也不一定意味着编码器会自动插入它.

角色	含义	需要区分的边界
BOS	Beginning of Sequence，序列开始	不一定存在，也不一定由后处理自动加入.
EOS	End of Sequence，序列结束	训练监督终点与生成停止条件需共同配置.
PAD	Padding，填充	补齐张量，不是自然语言内容.
UNK	Unknown，未知词元	回退表示可能丢失原始字符串.
SEP	Separator，分隔符	结构分界不必等同于生成终止.
CLS	Classification，分类位置	常用于汇聚分类表示，不是通用 BOS.
MASK	Mask，输入扰动符号	与注意力掩码、损失掩码是不同对象.
角色与哨兵	对话角色、工具边界、缺失片段标识	必须与训练数据及模板一致.

BERT 类句对输入可以采用 `[CLS] A [SEP] B [SEP]`；某些自回归模型采用起始符号、提示、回答和终止符号的序列；编码器—解码器则分别构建源序列与目标序列. 具体模型可能采用不同变体，应读取制品协议，不能只从架构名称推断固定编号或固定模板. 句段编号和位置编号是附加输入，不是特殊词元.

4.10.2　对话模板的条件语义

对话模板将结构化消息转换为词元序列，包括角色边界、轮次结束、工具调用和待生成回答的前缀. 同样的消息内容，经不同模板编码后得到不同条件，因此模型看到的任务也不同. 模板已经插入特殊词元时，再由后处理器重复加入起始符号，可能产生训练中未见的格式.

用户正文中出现与控制符号相同的字面字符串，应由输入策略明确处理为普通文本、转义文本或被允许的控制符号. 不能不加区分地让正文自行改变角色边界；反过来，也不能假定控制词元能独立消除提示注入. 结构解析和模型对内容的遵循属于不同层次.

截断同样需要理解结构. 截去回答尾部会删除真实结束位置；截去消息头部可能留下没有角色的内容；截去工具调用的括号或字段会破坏结构. 序列长度预算应在模板展开后计算，再按任务规则保留完整片段和必要边界. 特殊词元的数量本身也占用上下文容量.

4.11 填充、因果可见性及损失选择

4.11.1 掩码功能分类

长度分别为 T_b 的 B 条序列补齐为 $[B, T_{\max}]$ 张量. 令 $m_{bj} \in \{0,1\}$ 表示位置 j 是否是真实输入，填充位置为零. 注意力读取位置 j 时应排除填充键. 对有效查询位置 i，自回归模型还要求键位置不晚于查询位置，因此允许关系为

$$A_{bij} = m_{bj}\mathbf{1}[j \leqslant i]. \tag{4.35}$$

实现可以把不允许的位置映射为注意力分数上的负无穷偏置，也可以使用等价的稀疏或内核接口. 填充查询位置通常不承担输出任务；若额外将其整行全部屏蔽，应处理全屏蔽行的数值边界.

损失掩码 $\ell_{bt} \in \{0,1\}$ 则决定目标位置是否参与训练：

$$\mathcal{L} = -\frac{1}{\sum_{b,t} \ell_{bt}} \sum_{b,t} \ell_{bt} \log p_\theta(y_{bt} \mid \text{该位置可见输入}), \qquad \sum_{b,t} \ell_{bt} > 0. \tag{4.36}$$

提示词可以作为可见条件而不计入损失，真实 EOS 可以同时可见且接受监督，PAD 则通常既不是有效键也不接受监督. 因果掩码控制未来信息，填充掩码控制虚构位置，损失掩码控制学习目标，三者不可互相替代.

4.11.2 下一词元对齐

设一个自回归训练样本由提示词元 p、回答词元 a 与真实 EOS 构成；批处理需要再补齐两个位置. 以下表格将输入与目标显式错开一位：相对于目标，解码器输入在开头加入 BOS 并右移；等价地，相对于完整词元串，目标向左错开一位. 输入位置 t 的输出预测下一词元 y_t：

预测位置 t	1	2	3	4	5
输入	BOS	p	a	EOS	PAD
目标 y_t	p	a	EOS	PAD	PAD
输入有效性 m_t	1	1	1	1	0
回答监督 ℓ_t	0	1	1	0	0

第一个输出预测提示，因此此处不计损失；第二个输出以 BOS 和 p 为条件预测回答；第三个输出学习回答何时结束. 对于接受同一完整序列作为输入与标签的训练接口，损失内部通常将位置 t 的分数与位置 $t+1$ 的标签配对. 若已使用表中的输入—目标对齐，就不能再执行一次这种错位，否则监督会错开两个位置.

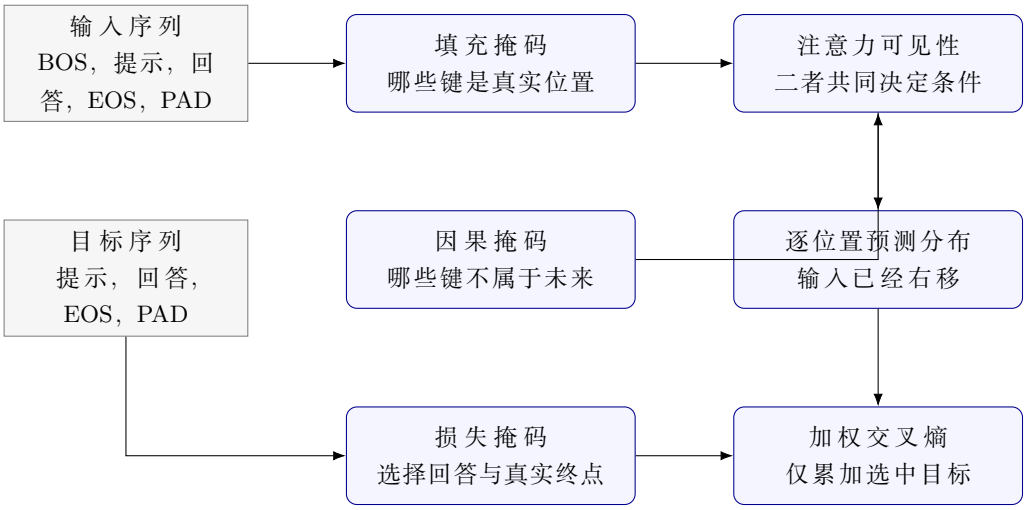


图 4.3: 填充、因果可见性和损失选择的职责．提示词进入条件上下文，不要求同时成为监督目标.

若 PAD 与 EOS 共享编号，表中真实 EOS 与两个填充目标在数值上相同，按“编号等于 EOS 就忽略”处理将误删第三个监督位置．正确区分依据是样本长度与位置来源．标签忽略值如 -100 是某些损失接口的约定，不属于词表，不能送入嵌入层.

4.11.3　编码器—解码器及批处理边界

条件生成还可将源文本交给编码器，将目标 $y_1, \ldots, y_T$, EOS 交给解码器监督．对应解码器输入为起始词元、$y_1, \ldots, y_T$．起始词元由模型配置决定，不要求等于分词器中的 BOS．源端填充掩码、目标端因果掩码及目标损失掩码分别构造；不能把自回归提示拼接的处理规则直接照搬到这一结构.

动态填充按批次最长长度补齐，可减少无效位置；长度分桶还能降低批内差异．在通常取最后位置预测的解码接口中，左右填充的选择会影响最后有效位置、位置编号和缓存索引，必须与接口行为配合．填充方向本身不能决定正确性，正确性来自有效位置被一致处理.

将多条独立样本打包为一条长序列也不是普通填充的反操作．若任务要求样本之间互不可见，仅插入 EOS 不会自动阻断注意力，还需限制跨样本读取．若预训练明确允许跨文档条件，则属于另一种训练定义，必须在数据目标中说明.

4.12　分词器及模型制品的一致性

4.12.1　分词器制品构成

完整制品至少应能恢复词表和编号、BPE 规则或 Unigram 概率、规范化、预分词、后处理、解码、特殊词元映射及对话模板．调用层还会设置截断、填充、最大长

度和返回张量形式．底层算法资产与上层调用配置共同决定输入整数序列．

标准分词接口通常可分为三层理解：加载工厂依据配置选择实现；包装层处理批处理和任务选项；后端执行具体编码流程．分层有助于定位差异，例如整数序列相同而特殊词元位置不同，应先检查模板和后处理，而非重新训练词表．批输出除了编号，还可能包含有效性掩码、句段编号、特殊位置标记和原文偏移．

注册新增词元以后，输入嵌入和输出投影必须包含其对应行．对于稠密连续编号，参数行数应与包含新增词元的完整词表大小一致；如果制品使用非连续编号，还需检查最大编号，不能只数词表项．新增行的随机初始化不等于获得语义，模型还需要相应训练．已有词元的编号置换更危险：张量形状可能完全合法，语义却已错位．

4.12.2　分词器语义一致性

对两套实现 T_1, T_2，若目标是兼容同一模型，核心要求是在指定输入集合上逐项比较 $T_1(s) = T_2(s)$，并分别核对特殊位置、偏移和批处理行为．仅验证 $D_1(T_1(s)) = D_2(T_2(s)) = s$ 只能证明两者可还原同一文本，不能证明模型输入相同．

检验集合应覆盖普通文本、重复空白、不同规范形式、混合语言、未见字符、特殊符号字面量、空文本、长输入以及长度差异显著的批次．重载应跨越实际序列化边界，避免原对象内存中的参数掩盖缺失文件．确定性编码与概率采样分别检验，不能用随机不一致否定本来就声明随机的算法，也不能让随机行为掩盖发布配置差异．

制品身份可由配置内容、训练来源摘要和文件哈希共同记录．数据来源、许可和分词器训练划分属于可追溯元数据，而不是所有加载库都必需识别的字段．分词器升级还会改变预编码训练数据、检索切块、服务计费单位和缓存键，故应与模型权重及上游预处理协同发布．序列可读性、算法等价性和模型任务效果分别需要不同证据，不能用其中一项替代其余两项．

本章前文已经把文本记录整理为带边界、特殊词元和有效性掩码的离散编号，但编号本身没有能够用于矩阵计算的语义几何．编号为 100 的词元既不必比编号为 10 的词元更重要，两者的数值差也不代表语言距离．模型必须先建立从离散身份到连续向量的映射，才能让后续网络对词元表示执行矩阵运算．

本节从查表的数学形式解释嵌入参数如何学习，以及输入输出权重共享为何改变梯度来源．词元嵌入表示离散身份，却不单独提供序列中的绝对或相对位置；第五章《注意力机制》将在形状为 $[B, T, d]$ 的连续表示之上建立注意力计算，第七章《位置表示》再讨论顺序信息如何进入模型．

4.13　嵌入表示

4.13.1　嵌入矩阵及张量形状

设词表大小为 V_{vocab}，模型宽度为 d. **嵌入**（Embedding）参数构成矩阵 $E \in \mathbb{R}^{V_{\text{vocab}} \times d}$,其中每一行对应词表中的一个编号.对批量编号张量 $K \in \{1, \dots, V_{\text{vocab}}\}^{B \times T}$，输入表示定义为

$$X_{bt,:} = E_{K_{bt},:}, \qquad X \in \mathbb{R}^{B \times T \times d}. \tag{4.37}$$

这里 b 是样本索引，t 是词元位置，最后一轴是连续特征. 嵌入矩阵是模型参数；编号则是用于选择参数行的离散输入. [3]

符号	含义	形状或范围
B, T, d	批量大小、序列长度、模型宽度	正整数
V_{vocab}	词表大小，区别于注意力值矩阵	正整数
E	词元嵌入参数	$V_{\text{vocab}} \times d$
X	单条序列的嵌入，省略批量轴	$T \times d$

4.13.2　独热乘法及查表的等价性

令 $e_k \in \{0,1\}^{V_{\text{vocab}}}$ 为第 k 个独热列向量，即 $(e_k)_j = \mathbf{1}[j = k]$. 它与嵌入矩阵相乘时，只有一行参与求和：

$$(e_k^{\mathsf{T}} E)_r = \sum_{j=1}^{V_{\text{vocab}}} \mathbf{1}[j = k] E_{jr} = E_{kr}. \tag{4.38}$$

因此查表与独热矩阵乘法表示同一个函数. 前者直接选择参数行，后者将选择操作写成了线性代数形式. 第三章《神经网络基础》已经用这一等价关系解释反向传播；在词表规模较大时，它还决定输入表示的资源代价.

显式独热张量需要 BTV_{vocab} 个元素，整数编号需要 BT 个元素，查表输出需要 BTd 个元素. 即使独热张量绝大多数元素为零，若按稠密数组保存仍须承担完整存储量. 稀疏选择无需形成这份中间数组，但不会消除嵌入参数本身的 $V_{\text{vocab}}d$ 个元素.

三词元嵌入查表　设一个三词元词表的参数为

$$E = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 1 & 1 \end{bmatrix}, \qquad (k_0, k_1, k_2) = (3, 1, 3). \tag{4.39}$$

[3] 书中矩阵行号通常从一开始，位置编码的时间索引从零开始. 程序采用从零开始的词表编号时，应在词表映射与矩阵行之间使用一致的转换；位置索引与词元编号是两个独立对象.

选择矩阵与输出分别为

$$S=\begin{bmatrix}0&0&1\\1&0&0\\0&0&1\end{bmatrix},\qquad X=SE=\begin{bmatrix}1&1\\1&0\\1&1\end{bmatrix}. \tag{4.40}$$

第 0 与第 2 个位置读取了同一参数行，因此在加入上下文和位置之前，两个输出完全相同. 词元在句中承担的不同作用，需要由后续计算建立，不能由固定查表独立完成.

4.13.3 重复编号及梯度累加

设上游梯度为 $G_X=\dfrac{\partial \mathcal{L}}{\partial X}$. 由 $X=SE$ 得到

$$\frac{\partial \mathcal{L}}{\partial E}=S^{\mathsf{T}}G_X,\qquad \frac{\partial \mathcal{L}}{\partial E_{j,:}}=\sum_{t:k_t=j}(G_X)_{t,:}. \tag{4.41}$$

对于上述例子，若三个位置的上游梯度为 $(1,2)$、$(3,4)$、$(5,6)$，则三行参数梯度依次为 $(3,4)$、$(0,0)$、$(6,8)$. 重复词元并不是两份独立参数，因此其梯度必须合并. 对批量输入，还要同时沿样本轴累加.

未被读取的行在这一条计算路径上的梯度为零，但这不等于该参数在整个训练步骤中保持不变. 其他使用该参数的计算路径、动量以及权重衰减都可能带来变化. 嵌入查表也不对整数编号提供通常的连续导数：可学习的是连续表项，而不是编号与编号之间的算术关系.

4.14 输入输出权重共享及语义几何

4.14.1 词表输出投影

设上下文模块产生隐藏状态 $h_t\in\mathbb{R}^d$，未共享时的输出投影为 $W_{\text{out}}\in\mathbb{R}^{d\times V_{\text{vocab}}}$. 词元 v 的分数为

$$u_{tv}=h_t^{\mathsf{T}}(W_{\text{out}})_{:,v}+c_v. \tag{4.42}$$

输出矩阵的每一列给出该词元在预测空间中的方向. 若输入与输出宽度相同，可以令 $W_{\text{out}}=E^{\mathsf{T}}$，得到

$$u_{tv}=h_t^{\mathsf{T}}E_{v,:}^{\mathsf{T}}+c_v. \tag{4.43}$$

这一约束称为**权重绑定**（Weight Tying），也称权重共享，输入查表与输出预测复用同一组词表参数 [22]. 两个独立的 $V_{\text{vocab}}d$ 参数块因此合并为一个. 若隐藏宽度不同，则还需要相容的投影，不能仅依靠转置完成共享.

共享规定的是参数一致性，而不是两个任务的含义完全相同. 输入侧回答词元如何参与上下文计算，输出侧回答当前状态与候选词元是否匹配. 约束这两个角色可以

减少自由参数，也可能限制模型可表达的组合．因此不能只依据参数节省推断所有数据和模型配置下的质量变化．

4.14.2　同一参数收到两条梯度路径

将所有参与计算的位置合并为 N 行，写作

$$X = \alpha SE, \qquad H = F_\theta(X), \qquad U = HE^{\mathsf{T}} + \mathbf{1}_N c^{\mathsf{T}}, \tag{4.44}$$

其中 α 是固定的输入缩放系数，F_θ 表示上下文模块．令 $G_U = \dfrac{\partial \mathcal{L}}{\partial U}$，$G_X = \dfrac{\partial \mathcal{L}}{\partial X}$．总微分包含输入侧和输出侧两个显式出现的 E：

$$\mathrm{d}\mathcal{L} = \langle G_X, \alpha S\,\mathrm{d}E\rangle_F + \langle G_U, H(\mathrm{d}E)^{\mathsf{T}}\rangle_F \tag{4.45}$$

$$= \left\langle \alpha S^{\mathsf{T}} G_X + G_U^{\mathsf{T}} H, \mathrm{d}E \right\rangle_F . \tag{4.46}$$

所以总梯度为

$$\boxed{\frac{\partial \mathcal{L}}{\partial E} = \alpha S^{\mathsf{T}} G_X + G_U^{\mathsf{T}} H.} \tag{4.47}$$

第一项沿被读取的编号累加，第二项来自候选词元的输出竞争．对于完整词表 Softmax，未作为输入出现的词元也可能通过第二项获得非零梯度；特殊数值或跨位置抵消可以使某些行恰好为零，因此“每行必然改变”并不是一般结论．

填充词元需要特别区分两项机制．查表时屏蔽填充行的梯度，只约束输入侧的局部操作；注意力掩码决定填充位置是否参与读取，损失掩码决定填充目标是否参与优化．若填充行还被共享输出投影使用，它仍可能收到输出路径梯度．将某一行初始化为零，既不能替代掩码，也不能自动保证共享训练后仍为零．

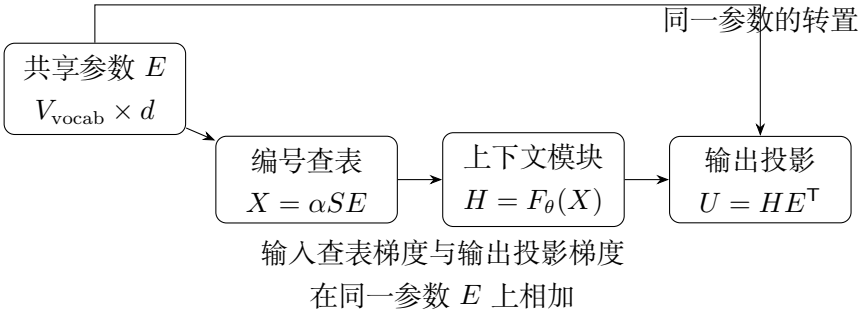


图 4.4: 输入输出权重共享的两条参数路径．图中省略输出偏置；参数存储只有一份，反向贡献有两类．

4.14.3　距离的含义依赖训练目标

连续向量允许定义长度、内积和夹角．对两个非零嵌入向量 u, v，**余弦相似度**（Cosine Similarity）为

$$\cos(u, v) = \frac{u^{\mathsf{T}} v}{\|u\|_2 \|v\|_2}, \tag{4.48}$$

其值位于 $[-1,1]$. 欧氏距离同时受方向和长度影响；余弦相似度消除了各自的正比例缩放，但对零向量没有定义. 使用哪种度量，应由模型的训练目标和后续使用方式决定.

嵌入能够在训练中承载有关共现与预测的规律，但向量接近不必表示同义. 反义词可能出现在相似句法环境中，词元片段也未必对应完整概念. 固定查表将一个编号对应到同一向量；上下文表示 h_t 则同时依赖其他词元和位置. 前者的邻近关系与后者的语境相似性不能直接互换，二维投影中看似清晰的簇也不能证明原空间具有同样分离程度.

还应注意表示的坐标并不天然可解释. 对相邻的两个线性映射 EW，任取可逆矩阵 A，有

$$(EA)(A^{-1}W) = EW. \tag{4.49}$$

在这个组合中，改变中间坐标并相应改变下游权重，输出保持不变，嵌入距离却可能变化. 正交变换保持内积，一般可逆变换则不保持. 这说明不能仅凭某一坐标轴命名语义属性. 上述等价仅针对所写的线性组合；含归一化、非线性或权重共享约束的完整网络，不一定允许任意这样的重参数化.

第五章　注意力机制

第四章《文本表示》先把文本记录变为词元编号和有效性掩码，再在第四章《文本表示》中的“从离散身份到连续表示”一节把离散符号变为形状明确的连续向量. 至此，每个批次已经形成 $X \in \mathbb{R}^{B\times T\times d}$，但仅有逐位置的映射仍不能完成上下文交互，也不会单独提供顺序坐标. 对于句子中同一个词元，其含义可能取决于前面的修饰语、远处的主语或另一段输入中的证据. **注意力机制**（Attention）通过输入相关的权重，在一组可访问位置之间汇总信息，使每个查询位置能够得到面向当前上下文的表示. 顺序信息如何进入这一计算将在第七章《位置表示》讨论.

注意力并不是“选出最重要的词”的同义词. 它首先是一个可微的向量到向量运算：计算匹配分数、在允许位置上归一化、对内容向量加权求和. 只有把这三个步骤及其梯度写清楚，才能正确理解多头、因果掩码和键值共享，也才能区分数学结构与执行内核的优化. 本章以原始 Transformer 的缩放点积形式为基础 [6]，逐项建立这一计算过程.

5.1　查询键值机制

5.1.1　位置相关信息汇总

设候选内容向量为 $v_1,\dots,v_S \in \mathbb{R}^{d_v}$. 对第 i 个查询位置，先确定非负权重 a_{ij}，且 $\sum_j a_{ij} = 1$，再得到

$$o_i = \sum_{j=1}^{S} a_{ij} v_j. \tag{5.1}$$

若所有权重相等，这只是平均汇总，不能根据当前查询区别候选内容；若只保留一个权重，则成为离散选择，通常无法直接通过所选索引传播梯度. 注意力以连续权重实现内容相关的读取，同时保留端到端求导的可能.

查询（Query，Q）表达用于匹配的当前状态，**键**（Key，K）表达候选位置的匹配特征，**值**（Value，V）表达该位置被读取的内容. 键和值拥有相同的位置索引，但承担不同的计算职责. 某个位置能否匹配与匹配后提供什么信息，是可以分别学习的

两个问题. [1]

5.1.2 输入来源及投影形状

先省略批量轴，采用每个位置占一行的约定. 令查询侧输入为 $X_q \in \mathbb{R}^{T\times d_q}$，键值侧输入为 $X_m \in \mathbb{R}^{S\times d_m}$. 两侧特征宽度可以不同. 单头投影定义为

$$Q = X_q W_Q, \qquad K = X_m W_K, \qquad V = X_m W_V, \tag{5.2}$$

其中 $W_Q \in \mathbb{R}^{d_q\times d_k}$，$W_K \in \mathbb{R}^{d_m\times d_k}$，$W_V \in \mathbb{R}^{d_m\times d_v}$. 查询和键必须投影到相同的 d_k 维空间才能计算点积；值的宽度 d_v 可以与 d_k 不同. 为集中讨论主线，本章公式暂省略偏置；加入偏置时按位置广播，反向梯度对位置求和.

先给出本章随后逐项建立的完整单头运算. 未使用掩码时，**缩放点积注意力**（Scaled Dot-Product Attention）定义为

$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^{\mathsf{T}}}{\sqrt{d_k}}\right)V. \tag{5.3}$$

Softmax 对每个查询沿键位置逐行计算. 式 (5.3) 依次包含匹配分数、尺度控制、权重归一化和内容汇总；下面将分别说明每一步的含义、前提与形状. 加入可见性掩码后，只需在归一化前向分数矩阵加入相应的加性掩码.

符号	含义	形状或范围
B, T, S	批量、查询长度、键值长度	正整数
d_q, d_m	查询与记忆侧特征宽度	正整数
d_k, d_v	单头匹配维数与内容维数	正整数
Q, K, V	投影后的查询、键与值	$T\times d_k, S\times d_k, S\times d_v$
U, M, A	分数、加性掩码、注意力权重	$T\times S$
O	单头输出	$T\times d_v$
H_q, H_{kv}	查询头数、键值头数	正整数

符号 V 在本章始终表示值矩阵；需要词表大小时另记为 V_{vocab}. 恢复批量轴后，各样本独立执行上述运算，输入通常写为 $[B,T,d_q]$ 和 $[B,S,d_m]$.

自注意力（Self-Attention）令两侧输入来自同一序列，即 $X_q = X_m = X$，因而 $T = S$；这不意味着投影后的 $Q = K = V$. 只要 W_Q, W_K, W_V 不同，三组向量就不同. **交叉注意力**（Cross-Attention）则允许输入来源不同，例如查询取自目标序列，键和值取自源序列的编码结果. 算法名称描述信息来源，单头的计算公式保持相同.

[1] 查询、键和值的名称借用了检索的表达方式；普通注意力不要求执行精确键匹配，也不保证某个键唯一对应一个位置. 它是连续相似度与加权汇总，而不是字典查找.

5.2　缩放点积及归一化

5.2.1　点积同时包含方向及长度

对查询行向量 q_i 和键行向量 k_j，点积分数为 $q_i k_j^{\mathsf{T}}$. 非零向量满足

$$q_i k_j^{\mathsf{T}} = \|q_i\|_2 \|k_j\|_2 \cos \angle(q_i, k_j). \tag{5.4}$$

因此点积不只是角度相似度，向量范数也影响分数. 将两者分别归一化后计算余弦相似度，是另一种分数定义，不能不加说明地替换. 学习到的投影可以调整匹配方向和尺度，并不要求向量各维预先具备可命名的语义.

5.2.2　点积分数缩放

平方根缩放的矩条件

查询和键的分量均值为零、二阶矩有限；两向量独立，各自坐标方差分别为 σ_q^2, σ_k^2，不同坐标的乘积互不相关. 该假设用于解释初始尺度，不视为训练后始终成立的不变量.

在这些条件下，对 $s = \sum_{r=1}^{d_k} q_r k_r$，有

$$\mathbb{E}[s] = \sum_r \mathbb{E}[q_r]\mathbb{E}[k_r] = 0, \tag{5.5}$$

$$\mathrm{Var}(s) = \sum_r \mathrm{Var}(q_r k_r) = \sum_r \mathbb{E}[q_r^2]\mathbb{E}[k_r^2] = d_k \sigma_q^2 \sigma_k^2. \tag{5.6}$$

故缩放分数 $u = \dfrac{s}{\sqrt{d_k}}$ 满足

$$\mathrm{Var}(u) = \sigma_q^2 \sigma_k^2. \tag{5.7}$$

特别地，两种分量方差均为一时，未经缩放的分数标准差为 $\sqrt{d_k}$，缩放后为一. 该设计抑制维数增长带来的尺度变化，并未把每个实际分数都限制在 $[-1, 1]$ 内.

独立性是推导的条件，而非自注意力训练后的不变量. 同一输入经过不同投影，可能产生相关的查询与键. 即使 q, k 是独立零均值向量，若坐标之间相关、协方差为 C_q, C_k，更一般的结果也是

$$\mathrm{Var}(qk^{\mathsf{T}}) = \mathbb{E}_q[qC_k q^{\mathsf{T}}] = \mathrm{tr}(C_q C_k), \tag{5.8}$$

只有在适当的各向同性条件下才退化为 $d_k \sigma_q^2 \sigma_k^2$. 位置旋转、归一化和训练中的参数变化都可能影响实际分数分布，应将平方根缩放理解为有明确假设的尺度设计.

5.2.3 Softmax 的行归一化

归一化指数函数（Softmax）把实数分数转成非负、总和为一的权重. 对分数矩阵 $U = \dfrac{QK^{\mathsf{T}}}{\sqrt{d_k}}$，每个查询独立归一化其键轴. 没有屏蔽时，

$$A_{ij} = \frac{\exp U_{ij}}{\sum_{r=1}^{S} \exp U_{ir}}, \qquad O = AV. \tag{5.9}$$

行和为一表示一个查询的读取权重在所有键之间分配；列和一般不为一，同一个键可以同时被很多查询读取. 因此注意力矩阵通常不是双随机矩阵，也不要求对称. 即使 $Q = K$ 使点积分数对称，各行归一化分母不同也可能破坏权重矩阵的对称性.

Softmax 对同一行的常数平移不变：对任意 c_i，以 $U_{ij} - c_i$ 代替 U_{ij}，分子和分母同时乘以 e^{-c_i}，结果不变. 取 $c_i = \max_j U_{ij}$，指数项不超过一，可避免大正分数的指数溢出. 这个稳定形式保留了相同的数学函数.

如果某个分数比其他分数大很多，权重便接近独热. Softmax 的**雅可比矩阵**（Jacobian Matrix）汇总其全部一阶偏导数，其对角项为 $a_j(1 - a_j)$、非对角项为 $-a_j a_r$，在这种极度集中的情形下，多数分数梯度会变小. 缩放有助于避免初始化附近无意造成的过早集中，但不能保证所有头都维持分散的权重，也不能把所有小梯度问题归因于 Softmax.

5.2.4 凸组合的范围及例外

在未使用**随机失活**（Dropout）的单头权重运算中，o_i 位于 $\{v_j\}$ 的凸包内. 任意线性投影 P 满足

$$P(o_i) = \sum_j a_{ij} P(v_j), \tag{5.10}$$

所以二维投影可以保留加权组合关系，适合展示汇总机制；二维投影不能保留的距离和方向，则不能用于证明原空间中的语义关系.

若在权重上施加保留概率为 q 的 Dropout，实际使用 $\widetilde{A}_{ij} = \dfrac{m_{ij} A_{ij}}{q}$，其行和在一次随机采样中一般不再等于一，甚至可能为零. 期望满足 $\mathbb{E}_m \widetilde{A} = A$，但每次输出不再必然处于同一个凸包. 检查“概率行和为一”应针对 Dropout 之前的归一化权重；不能把这一性质要求施加到所有训练模式输出上.

5.3　掩码规定可见性

5.3.1　在允许集合上定义 Softmax

为第 i 个查询定义允许读取的索引集合 $\mathcal{J}_i$. 当其非空时，最清楚的定义是

$$A_{ij} = \begin{cases} \dfrac{\exp(U_{ij} - c_i)}{\sum_{r\in\mathcal{J}_i} \exp(U_{ir} - c_i)}, & j \in \mathcal{J}_i, \\ 0, & j \notin \mathcal{J}_i, \end{cases} \qquad c_i = \max_{r\in\mathcal{J}_i} U_{ir}. \tag{5.11}$$

用加性掩码表示时，$M_{ij} = 0$ 对应允许，$M_{ij} = -\infty$ 对应禁止，可以简写为 $A = \mathrm{softmax}(U + M)$. 数学上 $e^{-\infty} = 0$，但具体浮点程序仍须避免全屏蔽行或非有限输入导致的不定运算.

布尔掩码中的真值没有跨所有接口统一的含义：有的接口使用真表示允许，有的使用真表示禁止. 正文用集合和加性形式消除这种歧义；实现时应在接口边界将外部约定转换为单一内部约定. 不能依靠张量类型或名称猜测方向. [2]

5.3.2　填充掩码、因果掩码及文档边界

填充掩码屏蔽没有真实内容的键值位置. 批量中的有效长度不同时，键掩码可由 $[B, S]$ 扩展到所有头和查询行. 屏蔽填充键并不保证填充查询的输出为零：填充查询仍可能读取有效键. 若这些查询应被忽略，需在损失或后续输出处理上明确实现；把嵌入的某一行固定为零不能替代注意力掩码.

因果注意力（Causal Attention）是在自注意力中采用因果可见性规则的注意力算子；**因果掩码**（Causal Mask）则是编码该规则的掩码. 对于等长的自回归训练，因果允许集合为 $\mathcal{J}_i = \{j : j \leqslant i\}$. 因此权重矩阵呈下三角结构，严格上三角被因果掩码屏蔽；因果注意力就是在这个允许集合上归一化并汇总值向量. 这里“因果”指遵守生成顺序的信息限制，不是统计学中的干预因果推断. 目标序列还必须正确右移：若位置 i 的输入已经含有它要预测的答案，那么允许读取自身也会泄漏目标，单靠三角掩码无法修复.

多种硬约束共同存在时，应取允许集合的交集. 例如位置既需要满足 $j \leqslant i$，又必须是有效词元、来自允许的文档. 等价地，各种禁止条件取并集. 将多篇文档拼接成一个序列时，分隔符本身不禁止跨文档读取；是否隔离仍取决于掩码.

[2] 硬掩码改变允许集合；位置偏置等有限加性项只是改变分数. 二者都可写成加法，但有限负偏置通常不等价于严格禁止读取.

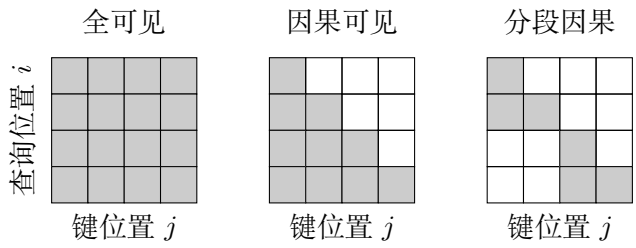


图 5.1: 四个位置的可见性模式，灰格表示允许读取. 右图把前两个与后两个位置视为不同文档，每篇内部保持因果性.

5.3.3 全屏蔽行的退化

若 $\mathcal{J}_i = \varnothing$，式 (5.11) 的分母为零. 将所有分数设为 $-\infty$ 后再减行最大值，还会遇到 $-\infty - (-\infty)$. 把所有分数换成同一个有限负数也不能解决语义问题：减去最大值后所有位置相等，Softmax 返回均匀分布，反而读取全部禁止内容.

应首先区分无效查询与合法查询. 对于必须产生有效表示的合法查询，全屏蔽行通常表示数据或掩码构造错误，应使其显式失败. 对于仅为批量补齐而存在的无效查询，可以约定该行输出恒为零，并在归一化之前分流处理；对应路径不对分数和值产生梯度. 这个零输出是额外定义的算子扩展，不是空集合上的概率分布.

左填充的因果批量尤其容易产生早期无效查询行：它们不能访问未来真实词元，而自身及过去位置又全是填充. 设置不同的 BOS 与 PAD 身份可以避免把真实起始位置误判为填充，但仍必须检查所有无效行. 不能先产生 NaN，再指望与零相乘恢复为有限结果.

算法 5.1 具有严格可见性的单头注意力

输入： $Q \in \mathbb{R}^{T\times d_k}$、$K \in \mathbb{R}^{S\times d_k}$、$V \in \mathbb{R}^{S\times d_v}$，每行允许集合 $\mathcal{J}_i$，查询有效标记 b_i. **输出：** $O \in \mathbb{R}^{T\times d_v}$；需要诊断或显式反向时同时保留权重 A. 本算法关闭权重 Dropout.

1. 检查匹配宽度及键值位置数一致，所有数值输入有限；创建全零的 O，需要保留权重时创建全零的 A.
2. 对查询 $i = 1, \ldots, T$：若 $b_i = 0$，保持该行输出为零并进入下一行；若 $b_i = 1$ 但 $\mathcal{J}_i$ 为空，报告非法可见性并停止.
3. 对每个 $j \in \mathcal{J}_i$ 计算 $u_j = \dfrac{q_i k_j^{\mathsf{T}}}{\sqrt{d_k}}$. 若点积出现非有限值，报告数值异常；否则取 $c = \max\limits_{j\in\mathcal{J}_i} u_j$.
4. 计算 $e_j = \exp(u_j - c)$ 及 $z = \sum\limits_{j\in\mathcal{J}_i} e_j$. 此时至少一项指数为一，分母严格为正.

5. 对允许位置令 $a_j = \dfrac{e_j}{z}$，累加 $o_i \leftarrow o_i + a_j v_j$；禁止位置不参加归一化，也不参加汇总．需要权重时写入 $A_{ij} = a_j$.

6. 全部查询完成后返回结果．有效行的权重非负、行和为一；无效行保持零，禁止位置的权重保持零.

显式逐行形式用于陈述完整定义，批量矩阵或融合实现可以更改执行次序，但必须保持允许集合、无效行约定与相同输入下的计算语义.

5.3.4　矩形因果注意力及前缀偏移

增量读取时，查询可能只包含最新的一个或几个位置，而键值包含全部已缓存前缀，此时 $T \neq S$．因果关系应由绝对位置给出

$$j_{\mathrm{abs}} \leqslant i_{\mathrm{abs}}, \tag{5.12}$$

而不是盲目在 $T \times S$ 矩阵上从左上角画下三角．例如键绝对位置为 $0, 1, 2, 3$，唯一查询的位置为 3，它应能读取四个键；若把该查询当作行号零，只允许键零，就改变了函数．缓存的存储与调度在推理篇展开，本章保留的是偏移不能丢失这一数学条件.

5.4　因果单头注意力

5.4.1　查询键值投影

取一个包含三个位置的自注意力问题，$d_q = d_m = d_k = d_v = 2$，无偏置、无 Dropout．给定

$$X = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}, \quad W_Q = W_K = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad W_V = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}. \tag{5.13}$$

故 $Q = K = X$，而

$$V = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 1 & 2 \end{bmatrix}, \qquad U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}. \tag{5.14}$$

这里特意选择简单矩阵使每步可手算；真实投影一般不是单位矩阵．加入因果掩码，记 $c = \dfrac{1}{\sqrt{2}} \approx 0.707107$，得到

$$U + M = \begin{bmatrix} c & -\infty & -\infty \\ 0 & c & -\infty \\ c & c & 2c \end{bmatrix}. \tag{5.15}$$

5.4.2 逐行归一化及汇总

第一行只有一个允许位置，所以权重为 $(1,0,0)$. 第二行除以 $1+e^{c}$，第三行先减去 c 再归一化，可得到精确形式

$$A=\begin{bmatrix} 1 & 0 & 0 \\ \dfrac{1}{1+e^{c}} & \dfrac{e^{c}}{1+e^{c}} & 0 \\ \dfrac{1}{2+e^{c}} & \dfrac{1}{2+e^{c}} & \dfrac{e^{c}}{2+e^{c}} \end{bmatrix} \approx \begin{bmatrix} 1 & 0 & 0 \\ 0.330238 & 0.669762 & 0 \\ 0.248255 & 0.248255 & 0.503490 \end{bmatrix}. \tag{5.16}$$

每行沿键轴归一化. 将它乘以同一组 V，得到

$$O=AV\approx \begin{bmatrix} 1 & 0 \\ 0.330238 & 1.339523 \\ 0.751745 & 1.503490 \end{bmatrix}. \tag{5.17}$$

第二行输出为 $0.330238(1,0)+0.669762(0,2)$；第三行则位于三个值向量构成的三角形内. 若误把 X 而不是 V 用于汇总，第二坐标会减半；这说明只把权重矩阵算对，仍不足以完成注意力计算.

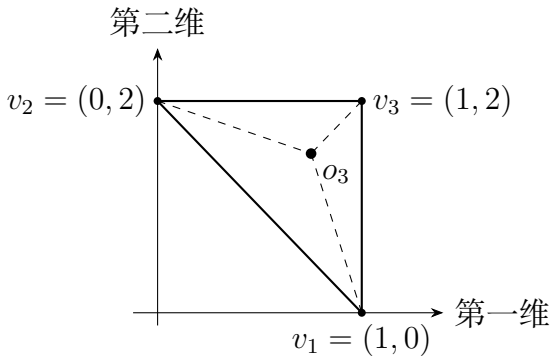


图 5.2: 单头例题第三个查询的加权汇总. 点的位置由给定矩阵计算，表示真实二维值空间中的凸组合，不是训练测量.

5.4.3 改变未来位置及检查信息边界

改变第三行输入，只会改变第三个键值及第三个查询. 在因果掩码下，前两个查询对第三个键的权重严格为零，所以 o_1,o_2 保持不变；o_3 则一般改变. 若去掉掩码，第一行原本对第三个键分配正权重，未来输入的变化便可以影响早期输出.

这一结论不仅针对给定数值成立，还可推广到多层网络：假设上一层位置 j 仅依赖输入前缀 $x_{\leqslant j}$，本层位置 i 只读取 $j\leqslant i$ 的状态，而其他运算均逐位置进行，那么本层位置 i 也只依赖 $x_{\leqslant i}$. 由层数归纳即可证明整个堆叠的前缀隔离. 位置编码必须使用相同位置定义，验证时还须固定随机运算，否则随机性本身会造成输出差异.

5.5　多头注意力及输出投影

5.5.1　独立的匹配空间

多头注意力（Multi-Head Attention，MHA）对同一组输入使用 H_q 组投影．标准 MHA 中每个查询头有自己的键值头，即 $H_{kv} = H_q = H$：

$$Q^{(h)} = X_q W_Q^{(h)}, \quad K^{(h)} = X_m W_K^{(h)}, \quad V^{(h)} = X_m W_V^{(h)}, \tag{5.18}$$

$$O^{(h)} = \mathrm{softmax}\left(\frac{Q^{(h)} K^{(h)\mathsf{T}}}{\sqrt{d_k}} + M^{(h)}\right) V^{(h)}, \tag{5.19}$$

$$C = \mathrm{Concat}_{\mathrm{feature}}(O^{(1)}, \ldots, O^{(H)}), \qquad Y = C W_O. \tag{5.20}$$

$C \in \mathbb{R}^{T \times H d_v}$，$W_O \in \mathbb{R}^{H d_v \times d_{\mathrm{out}}}$，所以最终输出为 $T \times d_{\mathrm{out}}$．用于残差连接时通常取 $d_{\mathrm{out}} = d_q = d_m = d$．常见配置为 $d_k = d_v = \dfrac{d}{H}$；这是一个方便的结构选择，注意力定义本身不强制每个宽度都相同．

把 W_O 按行分成 H 个块 $W_O^{(h)} \in \mathbb{R}^{d_v \times d_{\mathrm{out}}}$，可将拼接投影改写为

$$Y = \sum_{h=1}^{H} O^{(h)} W_O^{(h)}. \tag{5.21}$$

每个头先按自己的权重读取内容，再将读出结果投影到共同输出空间并求和．这里一般不能先平均注意力矩阵，再乘一个共同值矩阵，因为各头的 $V^{(h)}$ 与输出块可能不同．展示“平均注意力图”会隐藏头间差异，不等于网络实际执行了这个平均算子．

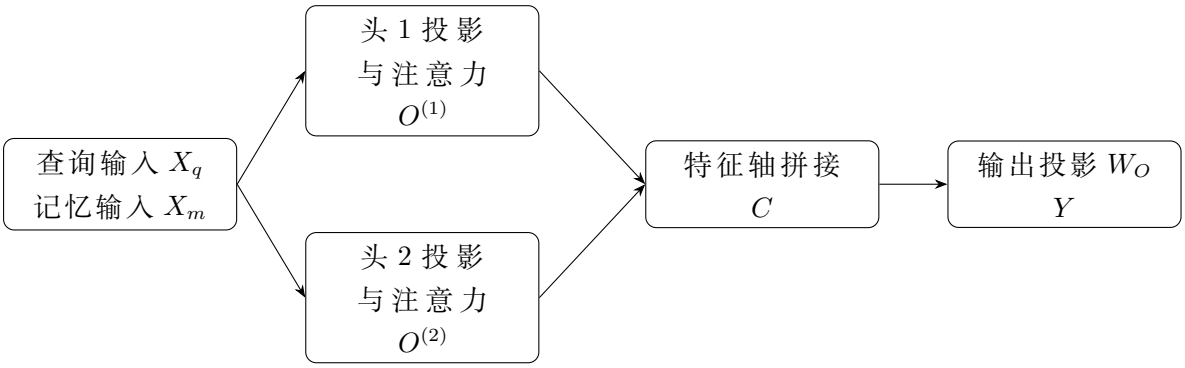


图 5.3: 两头结构示意．每个头分别归一化查询到键的读取权重，拼接发生在输出特征轴，不发生在序列轴．

5.5.2　多头特征空间

恢复批量后，投影输出 $[B, T, H d_k]$ 可以重排为 $[B, T, H, d_k]$，再交换中间两轴得到 $[B, H, T, d_k]$．键和值同理．头维度与序列维度承担不同含义：分头划分投影特征，通常每个头仍能读取全部允许位置；它不是把前半句交给一个头、后半句交给另一个头．

不同头可以学习不同的读取模式，但结构没有要求它们自动承担固定的语法、指代或推理分工. 随机初始化也会产生不同权重图，这只说明参数不同；如果要讨论某个头的具体功能，还需要训练后的干预、消融和跨样本证据.

5.5.3 投影打包及参数共享

自注意力中，三个投影读取同一个 X，可以将参数沿输出特征轴拼接为

$$W_{QKV} = [W_Q\ W_K\ W_V], \qquad XW_{QKV} = [XW_Q\ XW_K\ XW_V]. \tag{5.22}$$

在 $d_k = d_v = \dfrac{d}{H}$ 的标准 MHA 中，整体投影矩阵宽度为 $3d$. 这只是矩阵布局的合并，三个参数区间仍相互独立，不会减少参数量，也不产生 $Q = K = V$. 若框架线性层把权重存为“输出宽度乘输入宽度”，对应打包轴与正文约定相反，迁移时应先核对矩阵乘法方向.

对交叉注意力，Q 与 K, V 来自不同输入，不能使用一次 XW_{QKV} 代替全部投影. 键和值可以在记忆侧打包，查询则在查询侧单独计算. 参数文件可以打包存储，并不意味着执行时必须共用一个输入.

5.6 交叉注意力及结构关系

5.6.1 查询长度及记忆长度

在源到目标的条件生成中，源表示 $X_m \in \mathbb{R}^{S\times d_m}$ 已由编码器得到，目标侧状态 $X_q \in \mathbb{R}^{T\times d_q}$ 发出查询. 交叉权重为 $T \times S$，输出长度保持为 T. 因此它把源信息注入每个目标位置，不会把输出序列长度改成 S.

目标查询需要遵守自身生成顺序，但它通常可以读取全部已经可用的源序列. 不能因为目标侧自注意力使用因果掩码，就在交叉权重矩阵上照搬目标三角形. 若源信息也以流式方式到达，则应根据真实可用时点构造额外掩码；这是任务的信息约束，而非交叉注意力默认包含的性质.

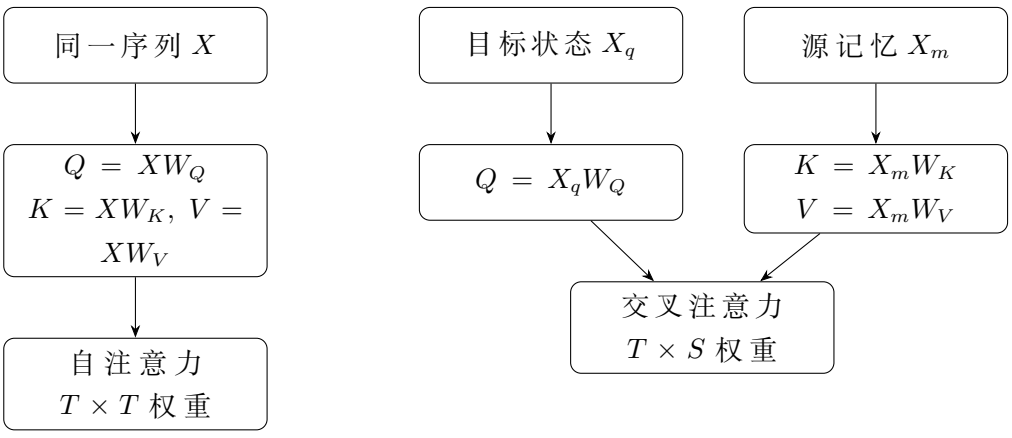


图 5.4: 自注意力与交叉注意力的信息来源. 右侧的源记忆共享位置索引，查询位置和源位置不要求一一对应.

5.6.2　单头交叉注意力

设源侧只有两个值 $v_1 = (2, 0)$、$v_2 = (0, 4)$，目标侧有三个查询，其缩放分数分别为 $(0, 0)$、$(\log 3, 0)$、$(0, \log 3)$. 逐行 Softmax 后的权重是

$$A = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{3}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{3}{4} \end{bmatrix}, \qquad O = \begin{bmatrix} 1 & 2 \\ \frac{3}{2} & 1 \\ \frac{1}{2} & 3 \end{bmatrix}. \tag{5.23}$$

所有目标位置读取同一组源值，却依据不同查询得到不同汇总. 输出仍有三行；第二行没有因行号超过源长度而失去定义. 若第二个源位置是填充，屏蔽该列后，每一行都只能读取 v_1.

编码器结构通常采用双向自注意力，解码器自回归结构采用因果自注意力，编码器–解码器结构则额外采用交叉注意力. 可见性与模块组成是两个维度，不能简单等同. 完整 Transformer 还需逐位置前馈变换、残差和归一化，并通过输出头与训练目标定义预测问题；这些组成在第六章《Transformer 网络结构》中建立.

5.7　注意力的反向传播

5.7.1　Softmax 雅可比的推导

固定一行分数 u，记 $Z = \sum_r e^{u_r}$、$a_j = \dfrac{e^{u_j}}{Z}$. 根据商法则，

$$\begin{aligned} \frac{\partial a_j}{\partial u_r} &= \frac{\mathbf{1}[j = r] e^{u_j} Z - e^{u_j} e^{u_r}}{Z^2} \\ &= a_j \big(\mathbf{1}[j = r] - a_r \big). \end{aligned} \tag{5.24}$$

因此雅可比为 $J = \operatorname{diag}(a) - a a^{\mathsf{T}}$. 若上游梯度为 $r = \dfrac{\partial \mathcal{L}}{\partial a}$，则

$$\frac{\partial \mathcal{L}}{\partial u_j} = a_j \left(r_j - \sum_s a_s r_s \right). \tag{5.25}$$

式中的 $\sum_s a_s r_s = a^{\mathsf{T}} r$ 是上游梯度按当前权重的平均. 分数梯度行和为零，因为对整行加常数不改变 Softmax. 这个零和性质是检查反向传播的有用不变量.

5.7.2 匹配分数梯度

令 $G = \dfrac{\partial \mathcal{L}}{\partial O} \in \mathbb{R}^{T\times d_v}$. 由 $O = AV$，逐元素有 $O_{ir} = \sum_j A_{ij}V_{jr}$，于是

$$\overline{V} = A^{\mathsf{T}}G \in \mathbb{R}^{S\times d_v}, \qquad R = GV^{\mathsf{T}} \in \mathbb{R}^{T\times S}, \tag{5.26}$$

其中上横线表示对应变量的损失梯度. 逐行应用式 (5.25)，得到

$$D_{ij} = \overline{U}_{ij} = A_{ij}\left(R_{ij} - \sum_r A_{ir}R_{ir}\right). \tag{5.27}$$

硬掩码允许集合固定时，禁止位置的权重为零，其分数梯度也为零. 对全屏蔽且定义为零输出的无效行，应沿定义的常量分支返回零梯度，而不是求一个不存在的空集合 Softmax 雅可比.

进一步对 $U = \dfrac{QK^{\mathsf{T}}}{\sqrt{d_k}}$ 求导，得到

$$\overline{Q} = \frac{DK}{\sqrt{d_k}} \in \mathbb{R}^{T\times d_k}, \qquad \overline{K} = \frac{D^{\mathsf{T}}Q}{\sqrt{d_k}} \in \mathbb{R}^{S\times d_k}. \tag{5.28}$$

值路径直接改变被读取的内容，查询和键路径则改变读取权重. 若一行只有一个可见键，其权重恒为一，该行所有分数梯度均为零，但对应值梯度可以非零. 这并非梯度实现错误，而是该行没有可学习的匹配选择.

5.7.3 返回投影矩阵及输入

由式 (5.2)，

$$\overline{W}_Q = X_q^{\mathsf{T}}\overline{Q}, \qquad \overline{W}_K = X_m^{\mathsf{T}}\overline{K}, \qquad \overline{W}_V = X_m^{\mathsf{T}}\overline{V}, \tag{5.29}$$

$$\overline{X}_q = \overline{Q}W_Q^{\mathsf{T}}, \qquad \overline{X}_m = \overline{K}W_K^{\mathsf{T}} + \overline{V}W_V^{\mathsf{T}}. \tag{5.30}$$

自注意力中 $X_q = X_m = X$，三个路径必须相加，不能把它们当作彼此独立的输入：

$$\overline{X} = \overline{Q}W_Q^{\mathsf{T}} + \overline{K}W_K^{\mathsf{T}} + \overline{V}W_V^{\mathsf{T}}. \tag{5.31}$$

多头输出 $Y = CW_O$ 还需先计算 $\overline{W}_O = C^{\mathsf{T}}\overline{Y}$ 与 $\overline{C} = \overline{Y}W_O^{\mathsf{T}}$，再按特征轴拆成各头上游梯度. 如果启用了权重 Dropout，则在经过 Softmax 反向前，还要乘以同一次前向使用的 $\dfrac{m}{q}$；不能重新随机采样掩码.

5.7.4 注意力局部梯度

仍取因果例题第二个查询，令损失 $\mathcal{L} = o_{2,1}$，只取该行输出第一坐标. 在其两个可见位置上，$a = (\alpha, 1-\alpha)$，其中 $\alpha = \dfrac{1}{1+e^{c}} \approx 0.330238$. 上游 $G_2 = (1, 0)$，故

$R_2 = (1, 0)$. 分数梯度为

$$D_{2,1} = \alpha(1-\alpha) \approx 0.221181, \qquad D_{2,2} = -\alpha(1-\alpha), \qquad D_{2,3} = 0. \tag{5.32}$$

值梯度分别在第一坐标得到 α 和 $1-\alpha$. 把 $K_1 = (1, 0)$、$K_2 = (0, 1)$ 代入式 (5.28), 得到

$$\overline{Q}_2 = \frac{\alpha(1-\alpha)}{\sqrt{2}}(1, -1) \approx (0.156399, -0.156399). \tag{5.33}$$

例如沿 $q_{2,1}$ 增大，会提高第一个键的分数，增加第一内容坐标；其导数应为正. 导数符号、行和为零与有限差分数值共同形成比“程序能够反向运行”更强的局部证据.

5.8　计算规模及键值共享

5.8.1　投影成本及位置交互成本

令标准 MHA 的输入输出宽度均为 d、H 个头且 $d_k = d_v = \dfrac{d}{H}$. 若一次乘加计作两次浮点运算，不计偏置和低阶逐元素操作，查询投影约需 $2BTd^2$ 次运算，两个键值投影合计约需 $4BSd^2$，输出投影约需 $2BTd^2$. 位置交互包括

$$QK^{\mathsf{T}}:\ 2BHTSd_k, \qquad AV:\ 2BHTSd_v. \tag{5.34}$$

因此自注意力 $T = S$ 时，线性投影为 $O(BTd^2)$，交互为 $O(BT^2d)$. 说“注意力是二次复杂度”指的是随序列长度增长的位置交互项，并不表示全部成本只有这一项. 短序列下投影和其他网络层仍可能占据主要开销.

显式保存注意力矩阵需要 $BHTS$ 个元素. 以 $B = 2, H = 16, T = S = 4096$、每元素两字节为例，仅一份权重矩阵即需

$$2 \cdot 16 \cdot 4096^2 \cdot 2 = 1\,073\,741\,824 \text{ 字节} = 1 \text{ GiB}. \tag{5.35}$$

这不含分数副本、反向中间量、参数或其他层. 因果掩码令一半左右的条目数学上为零，但若仍分配方形稠密张量，存储并不会自动减半. 避免完整物化权重矩阵的分块与融合方法将在高效推理章节讨论；它们改变内存访问和计算次序，不应被误写为一定改变注意力定义.

5.8.2　MHA、MQA 及 GQA

标准 MHA 为每个查询头配置独立键值投影. **多查询注意力**（Multi-Query Attention，MQA）让所有查询头共享一个键值头 [23]. **分组查询注意力**（Grouped-Query Attention，GQA）则使用介于一与查询头数之间的键值头数，每组查询共享一组键值 [24].

设 H_q 个查询头、H_{kv} 个键值头，且 H_q 可被 H_{kv} 整除. 令每组查询数为 $r = \dfrac{H_q}{H_{kv}}$，查询头从零编号，映射 $g(h) = \lfloor \dfrac{h}{r} \rfloor$，则

$$O^{(h)} = \mathrm{softmax}\left(\frac{Q^{(h)} K^{(g(h))\mathsf{T}}}{\sqrt{d_k}} + M^{(h)} \right) V^{(g(h))}. \tag{5.36}$$

$H_{kv} = H_q$ 对应 MHA，$H_{kv} = 1$ 对应 MQA，中间值对应分组结构. 共享键和值并不使组内各头输出相同，因为查询投影仍不同，Softmax 权重一般也不同. [3]

结构	查询头	键值头	KV 相对规模	结构变化
MHA	H_q	H_q	1	每头独立匹配和内容投影.
GQA	H_q	H_{kv}	$\dfrac{H_{kv}}{H_q}$	多个查询共用一组键值投影.
MQA	H_q	1	$\dfrac{1}{H_q}$	所有查询共用一组键值投影.

5.8.3 键值共享的计算边界

假设键和值的单头宽度均为 d_h，一个样本的 S 个历史位置需要存储 $2SH_{kv}d_h$ 个键值元素. 对于 $H_q = 32, H_{kv} = 8$，相同长度和精度下的键值存储为标准 MHA 的四分之一；MQA 则为三十二分之一. 输入宽度为 d 时，键值投影的参数数目由 $2dH_qd_h$ 变为 $2dH_{kv}d_h$.

但是查询头仍有 H_q 个，每头仍需对允许位置计算权重. 显式分数的形状仍是 $[B, H_q, T, S]$，核心匹配与汇总乘加数量不会仅因共享键值就按相同比例减少. 性能收益取决于重复读取是否真正复用、内核布局和硬件瓶颈，不能把键值存储比例直接当作端到端加速比.

MHA 到 GQA 的改变也不是把张量重排一下：不同键值投影被约束为共享，会改变可表达的函数族. 对既有权重进行合并后通常需要继续适配和重新评估，不能仅凭形状兼容声称输出保持不变. 分组数、单头宽度和投影布局均属于模型结构的一部分，须与权重共同保存.

5.9 注意力结构性质

5.9.1 排列等变性

没有位置表示、掩码也同步变换时，自注意力具有排列等变性. 令 P 为位置置换矩阵，每行每列恰有一个一，并满足 $P^{\mathsf{T}}P = I$. 考虑自注意力输入 X 变为 PX 的情形.

[3] 一些文献把 MHA 与 MQA 视为 GQA 的两个端点；讨论具体结构时，明确给出查询头数与键值头数比仅写缩写更可靠.

排列等变性证明的前提

各位置共享投影参数；不含显式位置输入；Softmax 逐行沿键轴计算；无随机算子．若存在掩码，可见性关系必须与内容同步置换，且每个查询至少有一个允许位置．

首先，线性投影与行置换可交换：

$$Q' = (PX)W_Q = PQ, \qquad K' = PK, \qquad V' = PV. \tag{5.37}$$

记缩放分数为 $U = \dfrac{QK^{\mathsf{T}}}{\sqrt{d_k}}$，则

$$U' = \frac{(PQ)(PK)^{\mathsf{T}}}{\sqrt{d_k}} = PUP^{\mathsf{T}}. \tag{5.38}$$

设置换后的第 i 行对应原第 $\pi(i)$ 行．同步置换键轴只改变一行分数的排列，不改变归一化分母，因此

$$[\mathrm{softmax}(U')]_{ij} = \frac{\exp(U_{\pi(i),\pi(j)})}{\sum_r \exp(U_{\pi(i),\pi(r)})} = [P\,\mathrm{softmax}(U)P^{\mathsf{T}}]_{ij}. \tag{5.39}$$

代入值矩阵并使用 $P^{\mathsf{T}}P = I$，得到

$$O' = P\,\mathrm{softmax}(U)P^{\mathsf{T}}PV = PO. \tag{5.40}$$

输出随输入一起重排，而不是保持不变，所以这里的性质是排列等变性而非排列不变性．

若将带加性掩码 M 的自注意力记为 $F(X; M)$，相应关系是

$$F(PX; PMP^{\mathsf{T}}) = PF(X; M). \tag{5.41}$$

固定下三角因果掩码通常不满足 $PMP^{\mathsf{T}} = M$；只置换词元却保留原掩码时，上述证明不再适用．位置表示与可见性共同使顺序具有作用，不能把“没有显式位置编码”直接等同于“网络不利用顺序”．

5.9.2　注意力权重的语义边界

同一个注意力权重乘以不同值向量，会产生不同输出；输出还经过投影、多头求和、残差和后续非线性变换．因此较大的某个 A_{ij} 只说明当前头在这一步汇总中给该值的系数较大，不能单独推出它对最终预测的贡献最大．

更直接地，若所有 v_j 都相同，那么无论注意力权重怎样改变，只要保持行和为一，输出就不变．这时权重图可以非常集中，也可以非常分散，而网络从此头得到的向量完全相同．若要研究因果影响，应明确干预对象和比较方式，并考虑干预后其他计算路径的变化．

5.9.3 局部性质及整体行为

一组有意义的结构验证应同时覆盖投影形状、允许集合、归一化行和、输出矩阵乘法及梯度. 因果结构还应核对未来扰动不改变早期输出；共享键值结构需核对各查询头映射到正确的组. 对框架实现做数值比较，应统一参数、偏置、缩放、位置表示、掩码语义和 Dropout 状态，随机初始化的两个模块没有逐值相等的理由.

浮点运算顺序不同可能使结果出现舍入差异，合理的相对与绝对误差容限应随精度、规模和数值范围确定. 局部公式一致只能证明对应算子的正确性，不能证明模型已学会翻译、检索或推理. 训练目标、数据与完整网络将进一步决定这些能力.

注意力计算的三个独立约束

投影决定匹配与内容的表示空间，掩码决定合法的信息来源，归一化决定允许位置之间的读取比例. 输出和梯度必须同时遵守这三项约束. 算法 5.1 落实单头定义；图 5.3 增加并行表示空间；共享键值则改变投影的共享关系. 三者均不能由一张权重热力图完整证明.

习题参考结果 第 1 题依次为 5×7、5×7、5×3. 第 2 题方差为 $16d_k$ 与 16. 第 5 题在把 Q,K,V 当作独立输入求偏导时，$\overline{K}_1=(0,\dfrac{\alpha(1-\alpha)}{\sqrt{2}})$，$\overline{K}_2$ 为其相反数. 第 7 题共 3 145 728 字节，即 3 MiB，为 MHA 的四分之一. 第 9 题方差为 $(1-q)\sum_j \dfrac{A_{ij}^2}{q}$；该结论要求固定归一化权重并对独立掩码取期望.

第六章　Transformer 网络结构

注意力算子解决位置之间如何交换信息，位置表示则使计算能够区分绝对或相对顺序，但二者尚未构成完整的语言模型．每个位置的表示还需要非线性变换，多层网络需要可训练的梯度路径，输入和输出需要与词元空间连接．Transformer 将这些职责组织为重复的网络层，并通过可见性规则定义信息能够沿什么方向传播．

本章从逐位置前馈网络、残差与归一化出发，组装编码器、解码器及编码器—解码器结构，最后完成从输入特征到预测概率的逐步计算．原始 Transformer 的后置归一化与整流激活配置，以及采用前置归一化、平滑激活或门控前馈网络的变体，将分别定义．共享组件不意味着函数相同，更不意味着不同配置的权重可以直接互换．注意力算子的推导见第五章《注意力机制》，位置进入计算的不同方式见第七章《位置表示》；本章着重讨论这些机制怎样组成可训练网络．

6.1　网络层的职责及张量约定

6.1.1　位置交互及特征变换

设输入为 $X \in \mathbb{R}^{B\times T\times d}$，其中 B 为批大小，T 为序列长度，d 为隐藏维数．**多头注意力**（Multi-Head Attention，MHA）在自注意力模式下将不同位置的信息汇入当前表示；逐位置前馈网络只沿特征轴计算，所有位置共享同一组参数．两者交替堆叠，使跨位置交互得到的特征能够被非线性重组，再影响下一层的读取．

残差流保持形状 $[B,T,d]$，成为各子层共同读写的表示空间．注意力的输出投影和前馈网络的下投影都必须回到维数 d，才能与输入逐元素相加．残差连接既不是通道拼接，也不是将不同层的特征简单平均．

符号	含义与形状
B, S, T	批大小、源序列长度、目标序列长度.
d, h, d_h	隐藏维数、注意力头数、单头维数；标准等宽结构满足 $d = hd_h$.
f	前馈网络中间维数，不必固定为 $4d$.
N_e, N_d	编码器层数、解码器层数.
X, C	目标状态 $[B, T, d]$、源端记忆 $[B, S, d]$.
V_{vocab}	输出词表大小，与注意力值矩阵 V 区分.
γ, β	归一化增益与偏置，均为 d 维向量.
J_F	将张量展平后，函数 F 相对于输入的雅可比.

结构推导的共同假设

所有子层在有效位置保持隐藏维数；标准多头结构满足 $d = hd_h$；注意力每个有效查询至少有一个可见键；LayerNorm 与 RMSNorm 沿特征轴计算，除显式手算例外均使用 $\epsilon > 0$. 掩码与输入位置均视为给定的非训练状态.

矩阵公式使用行向量约定，即 XW；求导时为方便讨论，将单个位置视为列向量，雅可比按“输出分量对输入分量”排列. 二者只是记号选择，参数存储布局必须在实现中对应. 除明确说明外，结构公式省略 Dropout，其影响将在训练部分单独讨论.

6.1.2 注意力子层的接口

令查询侧输入为 $X_q \in \mathbb{R}^{T\times d}$，键值侧输入为 $X_k \in \mathbb{R}^{S\times d}$. 第 r 个头计算

$$Q_r = X_q W_{Q,r}, \qquad K_r = X_k W_{K,r}, \qquad V_r = X_k W_{V,r}, \tag{6.1}$$

$$A_r = \operatorname{softmax}\left(\frac{Q_r K_r^{\mathsf{T}}}{\sqrt{d_h}} + M\right), \qquad O_r = A_r V_r. \tag{6.2}$$

其中三个投影矩阵为 $d \times d_h$，M 为 $T \times S$ 加性掩码，Softmax 沿键轴归一化. 拼接各头再乘 $W_O \in \mathbb{R}^{d\times d}$，得到 $T \times d$ 输出. 自注意力令两侧输入相同；交叉注意力则允许两侧来自不同网络、具有不同长度.

“自注意力中 $Q = K = V$”通常只是对接口输入来源的简写，投影后的三个矩阵一般并不相等. 三个投影可以打包为一次矩阵乘法，但仍具有独立参数区间，并未发生权重共享. 交叉注意力的查询与键值来自不同张量，不能用一个输入无条件完成全部三组投影.

6.2　前馈网络、激活及门控

6.2.1　两层前馈网络的非线性作用

逐位置前馈网络（Feed-Forward Network，FFN）定义为

$$F(X) = \phi(XW_1 + b_1)W_2 + b_2, \quad W_1 \in \mathbb{R}^{d\times f}, \quad W_2 \in \mathbb{R}^{f\times d}. \tag{6.3}$$

$b_1 \in \mathbb{R}^f, b_2 \in \mathbb{R}^d$ 按位置广播，张量路径为 $[B,T,d] \to [B,T,f] \to [B,T,d]$. 隐藏宽度 f 控制可用中间特征数量；它是结构超参数，不由注意力头数决定.

如果去掉激活函数，两个仿射变换可以合成一个仿射变换：

$$(XW_1 + b_1)W_2 + b_2 = X(W_1W_2) + (b_1W_2 + b_2). \tag{6.4}$$

因此仅增加中间宽度并不能提供同样的非线性表达能力. 激活使不同输入采用不同的有效特征组合. 例如 ReLU 把负预激活置零，其局部雅可比相当于在两个投影之间插入与输入有关的对角选择矩阵.

把无偏置 FFN 写为逐中间特征的和，可得到

$$F(x) = \sum_{j=1}^{f} \phi(xw_{1,j})w_{2,j}^{\mathsf{T}}, \tag{6.5}$$

$w_{1,j}$ 为上投影第 j 列，$w_{2,j}^{\mathsf{T}}$ 为下投影第 j 行. 上投影检测输入方向，下投影组合输出方向，激活控制该项的系数. 这是一种代数解释，不能由此把每个隐藏单元强行命名为某一事实或知识条目.

6.2.2　ReLU 及 GELU

原始 Transformer 使用**整流线性单元**（Rectified Linear Unit，ReLU）$\mathrm{ReLU}(u) = \max(0,u)$[6]. 其正半轴导数为一，负半轴导数为零，零点不可微，实现需采用约定的次梯度. ReLU 的输出非负，但经过下投影与残差相加后，最终表示仍可具有任意符号.

高斯误差线性单元（Gaussian Error Linear Unit，GELU）使用标准正态分布函数 Φ[25]：

$$\mathrm{GELU}(u) = u\Phi(u), \qquad \Phi(u) = \int_{-\infty}^{u} \frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}}\,\mathrm{d}t. \tag{6.6}$$

其导数由乘积规则得到

$$\mathrm{GELU}'(u) = \Phi(u) + u\frac{e^{-\frac{u^2}{2}}}{\sqrt{2\pi}}. \tag{6.7}$$

与 ReLU 在负区间直接截断不同，GELU 对负输入仍可能输出较小负值. 例如 $u=-1$ 时输出约为 -0.1587，$u=1$ 时约为 0.8413，零点导数为 0.5. 这描述了平滑变换的形状，并不证明采用 GELU 的模型在所有任务上都优于 ReLU.

一种常用近似为

$$\mathrm{GELU}(u) \approx \frac{u}{2}\left[1+\tanh\left(\sqrt{\frac{2}{\pi}}(u+0.044715u^3)\right)\right]. \tag{6.8}$$

精确形式与近似形式并非逐位等价. 迁移检查点或核对数值时，需要匹配所采用的定义，而不能仅检查配置中是否出现“GELU”这一名称.

6.2.3 GLU、GEGLU 及 SwiGLU

门控线性单元（Gated Linear Unit，GLU）将两条投影分支逐元素相乘. 省略偏置时，可以统一写为

$$F_{\mathrm{gate}}(X) = [g(XW_g) \odot (XW_u)]W_d, \quad W_g, W_u \in \mathbb{R}^{d\times f}, \quad W_d \in \mathbb{R}^{f\times d}. \tag{6.9}$$

经典 GLU 的门函数为 Sigmoid；**高斯误差线性门控单元**（GELU-Gated Linear Unit，GEGLU）使用 GELU；**Swish 门控线性单元**（Swish-Gated Linear Unit，SwiGLU）采用 Swish 门，本文取其参数为一，即**Sigmoid 线性单元**（Sigmoid Linear Unit，SiLU），写为 $\mathrm{SiLU}(u) = u\sigma(u)$，其中 $\sigma(u) = \dfrac{1}{1+e^{-u}}$[26]. 门控分支与内容分支都由当前输入决定，最后仍通过下投影回到残差宽度.

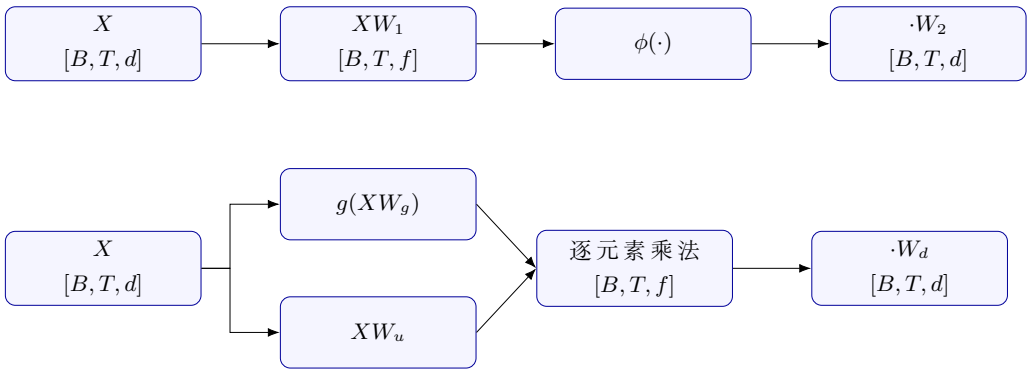


图 6.1: 两层 FFN 与门控 FFN. 门控结构新增输入投影，并通过逐元素乘法形成输入相关的交互.

图 6.1 中的两条分支在同一中间维数上对齐. 对单个中间分量，若 $a = xw_g, b = xw_u$，输出乘积为 $g(a)b$，其微分为

$$\mathrm{d}[g(a)b] = b\,g'(a)\,\mathrm{d}a + g(a)\,\mathrm{d}b. \tag{6.10}$$

这表明两条分支相互调制梯度. Sigmoid 门取值在零与一之间，而 SiLU 门可以为负且没有有限正上界，因而 SwiGLU 的“门”不能简单解释为开启概率，也不会自动形成稀疏专家路由.

无偏置两层 FFN 的参数量为 $2df$，同宽门控 FFN 为 $3df$．公平比较时应区分保持中间维数与保持参数预算：若原 FFN 宽度为 $4d$，等参数门控宽度约为 $\frac{8d}{3}$，因为 $3d(\frac{8d}{3}) = 2d(4d)$．实际维数还可能为硬件对齐取整．门控 FFN 不能仅靠更换式 (6.3) 中的激活名称实现，因为其线性投影数量已经改变．

6.3　残差连接及梯度传播

6.3.1　低秩增量更新

将任一保持形状的子层写为 F，残差更新为

$$x_{\ell+1} = x_\ell + F_\ell(x_\ell). \tag{6.11}$$

当 F_ℓ 接近零时，该层接近恒等映射．网络不必让一个全新的复合变换从随机初始化中学习如何复制所有输入信息，而可以围绕已有表示学习修正．

对输入求导得到

$$J_\ell = I + J_{F_\ell}, \qquad \frac{\partial \mathcal{L}}{\partial x_\ell} = (I + J_{F_\ell})^{\mathsf{T}} \frac{\partial \mathcal{L}}{\partial x_{\ell+1}}. \tag{6.12}$$

上游梯度既直接传回，也经过子层雅可比传回．堆叠多层后，端到端雅可比是这些矩阵的有序乘积，不能把矩阵乘积随意改写成标量乘积，也不能忽略各层的输入依赖．

6.3.2　恒等路径的稳定性边界

即使每层都有 I 项，梯度仍可能放大或衰减．一维构造中，若 $F_\ell(x) = ax$，则 L 层导数为 $(1+a)^L$；$a = 0.1$ 时随深度增长，$a = -0.1$ 时随深度衰减，$a = -1$ 时甚至直接为零．残差提供可用路径，参数、尺度、归一化和优化过程共同决定实际稳定性．

残差流方差也不必固定．若输入与增量近似不相关，则相加后的方差约为两者方差之和；若相关，还包含协方差项．因此深层网络常配合归一化、初始化或残差缩放．采用 $x_{\ell+1} = x_\ell + \alpha_\ell F_\ell(x_\ell)$ 会把雅可比改为 $I + \alpha_\ell J_{F_\ell}$，缩放影响前向和反向，不能当作对网络函数无影响的实现优化．

6.4 LayerNorm 及 RMSNorm

6.4.1 LayerNorm 的统计轴

层归一化（Layer Normalization，LayerNorm）按一个位置的特征维计算统计量[27]. 对 $x \in \mathbb{R}^d$，定义

$$\mu = \frac{1}{d}\sum_{i=1}^{d} x_i, \qquad c = x - \mu\mathbf{1}, \qquad s = \sqrt{\frac{1}{d}c^{\mathsf{T}}c + \epsilon}, \tag{6.13}$$

$$\widehat{x} = \frac{c}{s}, \qquad \mathrm{LN}(x) = \gamma \odot \widehat{x} + \beta. \tag{6.14}$$

$\epsilon > 0$ 防止零方差造成除零，$\gamma, \beta \in \mathbb{R}^d$ 是可训练向量. 这里除以 d，不是用于估计总体方差的无偏修正 $d-1$. 归一化目标是变换当前向量，不是进行无偏统计估计.

LayerNorm 不在批轴或时间轴聚合，因此批大小和其他样本不会直接改变当前位置的统计量. 但此前注意力若错误读取填充位置，污染已经进入当前向量，LayerNorm 并不能消除这种信息泄漏. 它也不同于依赖训练阶段移动统计量的典型 BatchNorm.

6.4.2 不变性及其边界

对整体平移 $x + a\mathbf{1}$，中心化结果不变，所以 LayerNorm 精确消除该方向. 忽略 ϵ 且方差非零时，正比例缩放也被消除；有非零 ϵ 时只在其相对方差足够小时近似成立. 负比例缩放会改变中心化向量的符号，不能称为相同输出.

以 $x = (1, 3)$ 为例，在 $\gamma = (1, 1), \beta = (0, 0)$ 下，均值为二，方差为一，输出为 $\frac{-1, 1}{\sqrt{1+\epsilon}}$. $x' = (11, 13)$ 的输出相同. 加上可学习仿射变换后，输出未必仍为零均值或单位方差；γ 的各分量不同，β 也可以引入偏移.

6.4.3 LayerNorm 雅可比推导

令中心化矩阵 $P = I - \dfrac{\mathbf{1}\mathbf{1}^{\mathsf{T}}}{d}$，则 $c = Px$、$P^{\mathsf{T}} = P$、$Pc = c$. 有

$$\mathrm{d}c = P\,\mathrm{d}x, \qquad \mathrm{d}s = \frac{c^{\mathsf{T}}\,\mathrm{d}x}{ds}. \tag{6.15}$$

对 $\widehat{x} = \dfrac{c}{s}$ 应用商法则，得到

$$\mathrm{d}\widehat{x} = \left(\frac{P}{s} - \frac{cc^{\mathsf{T}}}{ds^3}\right)\mathrm{d}x, \qquad J_{\mathrm{LN}} = \mathrm{diag}(\gamma)\left(\frac{P}{s} - \frac{cc^{\mathsf{T}}}{ds^3}\right). \tag{6.16}$$

这不是一个简单的对角缩放：均值和尺度计算使同一位置的各特征相互耦合. 设输出上游梯度为 g，令 $h = \gamma \odot g$，则输入梯度也可写为

$$\nabla_x \mathcal{L} = \frac{1}{s}\left[h - \overline{h}\mathbf{1} - \widehat{x}\,\overline{h \odot \widehat{x}}\right], \quad \overline{h} = \frac{1}{d}\sum_i h_i. \tag{6.17}$$

仿射参数梯度分别为 $g \odot \widehat{x}$ 和 g，训练时再对批及位置求和．式 (6.17) 说明某一特征上的损失会通过统计量影响其他特征，因此不能把均值与方差当作不参与求导的常数．

6.4.4　RMSNorm 的尺度归一化

均方根归一化（Root Mean Square Normalization，RMSNorm）保留均值，仅用均方根缩放 [28]：

$$r = \sqrt{\frac{1}{d} x^{\top} x + \epsilon}, \qquad \mathrm{RMSNorm}(x) = \gamma \odot \frac{x}{r}. \tag{6.18}$$

这里采用无偏置版本，参数量为 d；LayerNorm 的增益与偏置共为 $2d$．RMSNorm 不具有整体平移不变性．对 $(1, 3)$，其输出为 $\dfrac{1, 3}{\sqrt{5 + \epsilon}}$，与 LayerNorm 的 $\dfrac{-1, 1}{\sqrt{1 + \epsilon}}$ 明显不同．

相同的微分方法给出

$$J_{\mathrm{RMS}} = \mathrm{diag}(\gamma) \left(\frac{I}{r} - \frac{x x^{\top}}{d r^3} \right). \tag{6.19}$$

LayerNorm 多出中心化投影 P，RMSNorm 则保留均值方向．二者都将尺度变换与梯度传播联系起来，但不存在由这两条公式就能推导出的“RMSNorm 必然更准确”结论．计算代价也应区分算子开销与整网开销，不能把某项归一化的局部加速直接当作模型吞吐收益．

6.5　Pre-LN 及 Post-LN 的结构差异

6.5.1　两种子层的定义

对一个注意力或 FFN 子层 F，**后置层归一化**（Post-Layer Normalization，Post-LN）将归一化放在残差相加之后；**前置层归一化**（Pre-Layer Normalization，Pre-LN）将其放在子层输入端：

$$y_{\mathrm{post}} = \mathrm{LN}(x + F(x)), \tag{6.20}$$

$$y_{\mathrm{pre}} = x + F(\mathrm{LN}(x)). \tag{6.21}$$

同样的 F 和同样的归一化参数通常产生不同结果．Post-LN 的残差分支最终也经过归一化，Pre-LN 的主残差流则绕过本子层归一化．因此，“Add & Norm”应展开为具体运算顺序，不能作为一个不区分位置的方框．

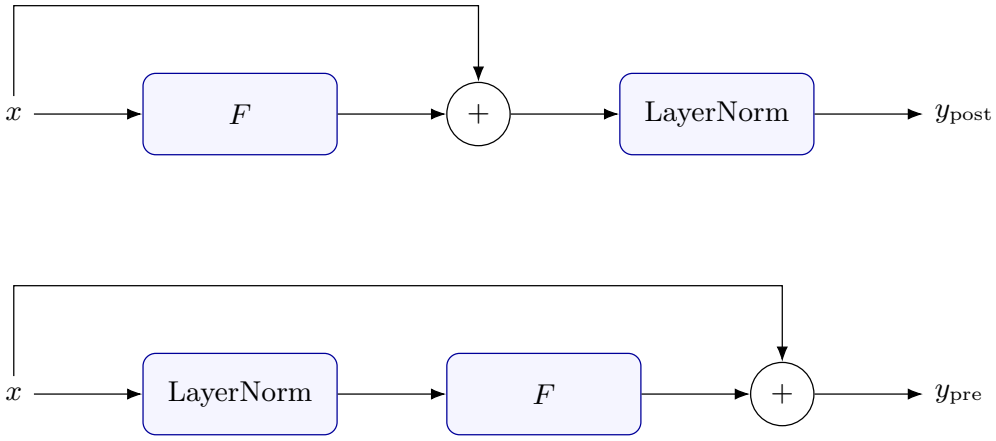


图 6.2: 归一化位置改变前向函数和反向路径. Pre-LN 的恒等项绕过本子层归一化，末端归一化仍影响最终损失.

沿图 6.2 的路径应用链式法则，二者雅可比分别为

$$J_{\text{post}} = J_{\text{LN}}(x + F(x))[I + J_F(x)], \tag{6.22}$$

$$J_{\text{pre}} = I + J_F(\text{LN}(x))J_{\text{LN}}(x). \tag{6.23}$$

Pre-LN 有显式不经过本子层归一化的恒等项；Post-LN 的两条局部路径都需乘归一化雅可比. 关于初始化附近梯度的研究说明，归一化位置会改变深层训练行为 [29]. 但该结构分析并没有消除学习率、初始化、深度和数据分布的影响.

归一化位置不能作为等价替换

Pre-LN 的局部雅可比保留显式恒等项，Post-LN 的全部局部路径经过归一化. 两者具有不同函数与优化性质；改变顺序后必须重新核对训练配置和权重语义，而不能仅比较张量形状.

6.5.2 堆叠末端及配置一致性

典型 Pre-LN 堆叠在最后添加一次归一化，再进入输出头，以控制主残差流到预测层的尺度. 末端归一化应作为完整模型的一部分记录，不能因为每层已有归一化就省略. 其雅可比仍处于损失到主干的路径中，因此不能说 Pre-LN 的全部梯度均不受归一化影响.

本章将原论文配置记为 “Post-LN、LayerNorm、ReLU”，将一个便于比较的变体记为 “Pre-LN、LayerNorm、GELU、末端 LayerNorm”. RMSNorm 与 SwiGLU 是另外两项独立设计选择，不由 Pre-LN 自动推出. 逐项改变配置才能判断差异来源；一次同时替换三项设置后观察到损失变化，不能全部归因于其中一项.

6.6　三类模块结构及四种可见性

6.6.1　编码器及因果解码器

仅编码器（Encoder-only）结构堆叠自注意力和 FFN，通常允许有效位置双向读取，输出上下文表示，之后连接分类、标注或检索任务头．**仅解码器**（Decoder-only）结构通常堆叠因果自注意力和 FFN，再预测下一词元．它不是把原始解码器原封不动保留，因为通常已经删除交叉注意力子层．

模块组成与可见性是两个维度．一个堆叠是否含交叉注意力属于模块问题；某个位置是否能读取另一个位置属于掩码问题．可通过不同掩码在相似模块上训练不同目标，不能仅凭“Decoder”名称判断全部信息路径．

6.6.2　编码器—解码器的数据流

编码器—解码器（Encoder–Decoder）先将源输入 $x_{1:S}$ 编码为记忆 $C \in \mathbb{R}^{B\times S\times d}$；解码器根据右移后的目标前缀查询这份记忆，输出 $[B,T,d]$ 表示．交叉注意力的查询来自目标侧，键和值来自源侧，权重形状为 $[B,h,T,S]$．

阅读图 6.3 时，先沿源侧观察记忆的形成，再沿目标侧观察三个子层如何更新目标表示．交叉注意力的查询来自目标侧，因此输出长度为 T；键和值来自源侧，其长度为 S．训练时，右移操作使位置 t 的输入不包含待预测的 y_t，因果掩码再阻止该位置读取后续目标输入．图中各子层均先残差相加、后归一化．

图中的上下游连接需要由层内公式具体化．以原始 Post-LN 顺序定义编码器第 ℓ 层，且各归一化参数独立：

$$U_\ell = \mathrm{LN}_{\ell,1}[C_{\ell-1} + \mathrm{MHA}_{e,\ell}(C_{\ell-1}, C_{\ell-1})], \tag{6.24}$$

$$C_\ell = \mathrm{LN}_{\ell,2}[U_\ell + F_{e,\ell}(U_\ell)]. \tag{6.25}$$

这里 MHA 的两个输入分别表示查询侧与键值侧．源嵌入加位置表示得到 C_0，最终 $C = C_{N_e}$．不同层通常具有独立参数；“堆叠相同结构”不等于重复使用同一组权重．

对应解码器层包含三个子层：

$$U_\ell = \mathrm{LN}_{\ell,1}[X_{\ell-1} + \mathrm{MHA}_{c,\ell}(X_{\ell-1}, X_{\ell-1})], \tag{6.26}$$

$$Z_\ell = \mathrm{LN}_{\ell,2}[U_\ell + \mathrm{MHA}_{x,\ell}(U_\ell, C)], \tag{6.27}$$

$$X_\ell = \mathrm{LN}_{\ell,3}[Z_\ell + F_{d,\ell}(Z_\ell)]. \tag{6.28}$$

下标 c 表示目标侧因果自注意力，x 表示交叉注意力．每个目标层使用同一份最终源记忆，但各层的交叉注意力投影参数独立．源序列不随目标生成增长，因此编码器可以在生成前运行一次．

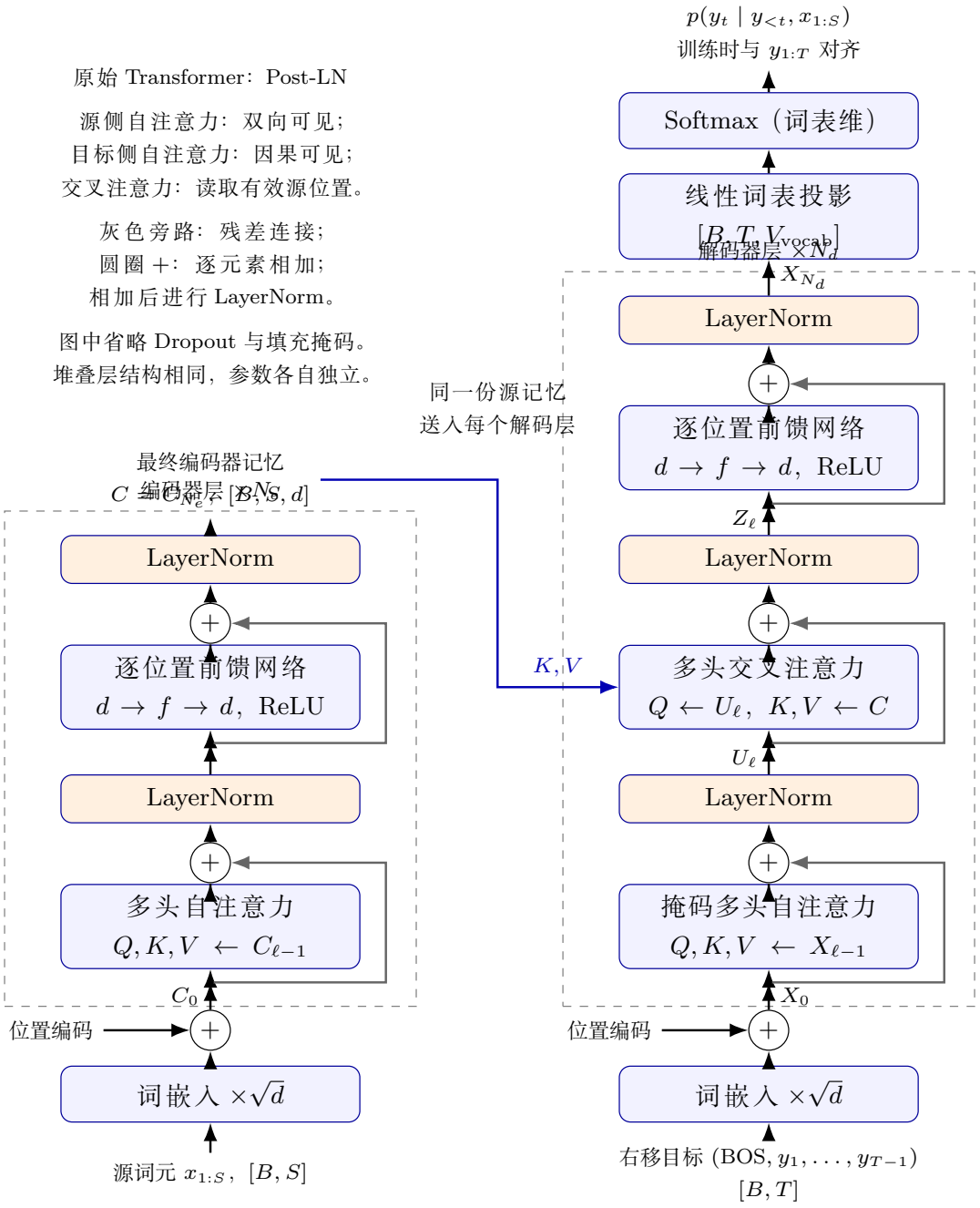


图 6.3: 原始 Transformer 的编码器—解码器架构（Post-LN）. 主数据流自下而上；灰色旁路显式表示残差连接，加法后执行 LayerNorm. 蓝色跨列箭头将最终源记忆送入每个解码器层. 依据原始结构绘制 [6].

Pre-LN 解码器则可写为

$$U_\ell = X_{\ell-1} + \mathrm{MHA}_{c,\ell}(\mathrm{LN}_{\ell,1}(X_{\ell-1}), \mathrm{LN}_{\ell,1}(X_{\ell-1})), \tag{6.29}$$

$$Z_\ell = U_\ell + \mathrm{MHA}_{x,\ell}(\mathrm{LN}_{\ell,2}(U_\ell), C), \tag{6.30}$$

$$X_\ell = Z_\ell + F_{d,\ell}(\mathrm{LN}_{\ell,3}(Z_\ell)). \tag{6.31}$$

这里明确选择只归一化交叉注意力查询侧输入，C 已由源端网络输出；若另一配置还归一化源记忆，应另作定义，不能在实现中无说明地增加运算.

6.6.3　编码器、因果及前缀掩码对照

令 $A_{ij} = 1$ 表示位置 i 允许读取位置 j，并暂时不含填充. 对长度四的序列，双向、因果和前两位置为条件前缀的掩码分别为

$$A_{\mathrm{bi}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad A_{\mathrm{causal}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad A_{\mathrm{prefix}} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}. \tag{6.32}$$

前缀解码器（Prefix Decoder）的条件部分可双向交互，生成部分读取全部前缀及生成历史. 设前缀长度为 P，其规则为

$$A_{ij} = \mathbf{1}[(i \leqslant P \wedge j \leqslant P) \vee (i > P \wedge j \leqslant i)]. \tag{6.33}$$

训练若把前缀内部各位置也按普通下一词元目标评分，可能发生泄漏，因为前缀状态可读取前缀未来；应按实际目标选择监督位置. 前缀双向条件化不等于拥有独立编码器和交叉注意力，两者的信息变换与参数结构仍不同.

第四种情况是交叉注意力，其矩阵为 $T \times S$，一般允许每个目标查询读取全部有效源位置. 它不在源轴应用目标侧因果三角形. 填充、文档隔离或流式源输入可以进一步限制各类掩码，但应在任务定义中明确.

6.6.4　多层因果性定理

如果初始位置 i 的状态只依赖输入 $x_{\leqslant i}$，且每层位置 i 只读取前层 $j \leqslant i$ 的状态，那么下一层位置 i 也只依赖 $x_{\leqslant i}$. 逐位置 FFN、残差和特征轴归一化都不会引入其他时间位置，因此归纳可知任意层均满足因果性.

编码器—解码器中，同样的结论是目标位置仅依赖源输入及真实目标前缀，而非未来目标. 这个证明的前提包括正确右移：若把当前位置真实目标直接作为同位置输入，即使使用下三角掩码，也已在初始表示中泄漏答案. 所谓因果注意力指生成顺序上的信息约束，并非统计学中的因果识别. [1]

[1] 改变未来输入检验早期输出，是检验特定前向路径的信息隔离；它既不证明模型理解了因果关系，也不穷尽所有数据泄漏来源.

6.7 张量路径及参数核算

6.7.1 单层形状表

标准等宽多头注意力要求 $d = hd_h$. 以下以目标自注意力和普通两层 FFN 为例，偏置按末轴广播.

运算	输入或参数	输出
QKV 投影	$[B,T,d]$，三个 $d\times d$ 矩阵	三个 $[B,T,d]$
拆头	$d = hd_h$	三个 $[B,h,T,d_h]$
缩放点积	$\dfrac{QK^{\mathsf{T}}}{\sqrt{d_h}}$	$[B,h,T,T]$
归一化与读取	掩码后 Softmax，再乘 V	$[B,h,T,d_h]$
拼头与输出投影	拼接末轴，乘 $d\times d$	$[B,T,d]$
残差与归一化	两条 $[B,T,d]$ 支路	$[B,T,d]$
FFN 上投影与激活	$d\times f$	$[B,T,f]$
FFN 下投影	$f\times d$	$[B,T,d]$
词表输出头	$d\times V_{\text{vocab}}$	$[B,T,V_{\text{vocab}}]$

交叉注意力将键值长度换为 S，分数形状为 $[B,h,T,S]$，输出仍为目标侧 $[B,T,d]$. 其输出长度由查询数决定，不由源长度决定.

6.7.2 参数量核算

带偏置的标准 MHA 具有四个 $d\times d$ 投影及四个 d 维偏置，共 $4d^2+4d$ 个参数；带偏置普通 FFN 为 $2df+f+d$；两个 LayerNorm 共为 $4d$. 因此，编码器层或仅解码器层的参数量为

$$P_{\text{layer}} = 4d^2 + 2df + f + 9d. \tag{6.34}$$

含交叉注意力的解码器层有两组 MHA、一个 FFN 和三个 LayerNorm，总计

$$P_{\text{encdec layer}} = 8d^2 + 2df + f + 15d. \tag{6.35}$$

两式不含嵌入、位置参数、末端归一化及输出头，也不适用于直接删偏置、共享键值头或门控 FFN 的配置.

例如 $d = 512, f = 2048$ 时，前者为 3 152 384，后者为 4 204 032. 六个编码器层与六个解码器层合计 44 138 496 个参数，但这不是模型总参数量. 若使用无偏置 SwiGLU 和两次无偏置 RMSNorm，仅解码器层对应为 $4d^2+3df+2d$，比较时必须同时说明投影是否含偏置.

6.7.3　权重共享的前提及梯度

若目标嵌入为 $E \in \mathbb{R}^{V_{\text{vocab}} \times d}$，输出头可以共享其转置：

$$z_t = h_t E^{\mathsf{T}} + b_{\text{out}}. \tag{6.36}$$

独立输出矩阵被省去，减少 $V_{\text{vocab}} d$ 个参数．共享矩阵收到输入查表和输出投影两条路径的梯度之和．输出路径的 Softmax 通常使许多词表行得到梯度，因而即使某行在嵌入查表中被设为不更新，也不能推断它在共享输出路径中绝对不变．

源嵌入、目标嵌入和输出头是否能够进一步共享，需要兼容的词表编号及维数．仅仅词表大小相同不够，同一行必须指向同一词元．共享降低参数自由度，并非保证输入表示与输出语义天然一致；这一约束是否有益仍由训练目标和数据决定．

6.7.4　计算及存储规模

标准自注意力投影规模为 $O(BTd^2)$，两次注意力矩阵乘法为 $O(BT^2d)$，普通 FFN 为 $O(BTdf)$．显式保存逐头权重需要 $O(BhT^2)$ 元素．交叉注意力包含 $O(BTSd)$ 的读取计算，并有源侧和目标侧投影．

保持总宽度 d 不变而增加头数，标准投影总参数量并不会线性增加；每头维数相应缩小．然而显式注意力矩阵含头轴，存储开销可能改变．高效内核可以避免完整物化权重矩阵，这改变访存和执行路径，不能据此说密集注意力在数学上已经不包含位置对之间的交互．

以下构造一个两位置、单头、隐藏维数为二的因果 Pre-LN 层，使用普通 ReLU FFN．它是为了便于逐步核对的具体数学网络，不代表推荐训练配置．输入已包含嵌入和位置表示：

$$X = \begin{bmatrix} 1 & 3 \\ 4 & 2 \end{bmatrix}, \qquad W_Q = W_K = W_V = W_O = I_2. \tag{6.37}$$

所有偏置为零，所有归一化增益为一，省略 Dropout．本例各次方差非零，为获得简洁精确值取 $\epsilon = 0$；实际实现应保持正 ϵ 并相应计算数值．

前馈矩阵取

$$W_1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix}, \qquad W_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & -1 \end{bmatrix}. \tag{6.38}$$

层后再做一次最终 LayerNorm，并采用三词元输出头

$$W_{\text{out}} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \end{bmatrix}, \tag{6.39}$$

三列依次对应词元 A, B, C．本例使用独立输出矩阵，不进行权重共享．

逐行归一化得到

$$\widehat{X} = \mathrm{LN}(X) = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{6.40}$$

因投影为恒等矩阵，$Q = K = V = \widehat{X}$. 缩放分数与因果权重为

$$S = \frac{QK^{\mathsf{T}}}{\sqrt{2}} = \begin{bmatrix} \sqrt{2} & -\sqrt{2} \\ -\sqrt{2} & \sqrt{2} \end{bmatrix}, \qquad A = \begin{bmatrix} 1 & 0 \\ 1-p & p \end{bmatrix}, \quad p = \frac{e^{\sqrt{2}}}{e^{\sqrt{2}} + e^{-\sqrt{2}}}. \tag{6.41}$$

$p \approx 0.944193$. 第一行只允许一个键，故权重严格为一；第二行比较两个键. 记 $c = 2p - 1 \approx 0.888386$，则

$$O = AV = \begin{bmatrix} -1 & 1 \\ c & -c \end{bmatrix}, \qquad U = X + O = \begin{bmatrix} 0 & 4 \\ 4+c & 2-c \end{bmatrix}. \tag{6.42}$$

注意力改变了表示，输出维数仍为二，因而可以与输入相加.

U 两行的特征大小关系没有逆转，因此

$$\mathrm{LN}(U) = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{6.43}$$

进入 FFN 后，

$$\mathrm{LN}(U)W_1 = \begin{bmatrix} -1 & 1 & -2 \\ 1 & -1 & 2 \end{bmatrix}, \tag{6.44}$$

$$\mathrm{ReLU}(\mathrm{LN}(U)W_1) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 2 \end{bmatrix}, \tag{6.45}$$

$$F(\mathrm{LN}(U)) = \begin{bmatrix} 0 & 1 \\ 3 & -2 \end{bmatrix}. \tag{6.46}$$

第二条残差得到

$$H = U + F(\mathrm{LN}(U)) = \begin{bmatrix} 0 & 5 \\ 7+c & -c \end{bmatrix}. \tag{6.47}$$

此处 FFN 的中间维数为三，输出重新回到二. 若遗漏下投影，则无法进行这次残差相加.

末端归一化后，$\overline{H} = [(-1, 1); (1, -1)]$. 于是词表分数为

$$Z = \overline{H}W_{\mathrm{out}} = \begin{bmatrix} -1 & 1 & 2 \\ 1 & -1 & -2 \end{bmatrix}. \tag{6.48}$$

逐行 Softmax 得到近似概率

$$P = \begin{bmatrix} 0.035119 & 0.259496 & 0.705385 \\ 0.843795 & 0.114195 & 0.042010 \end{bmatrix}. \tag{6.49}$$

若两个监督目标分别为 B, A，平均负对数似然为

$$\mathcal{L} = -\frac{1}{2}[\log(0.259496) + \log(0.843795)] \approx 0.759429. \tag{6.50}$$

至此，输入、归一化、注意力、两条残差、FFN、末端归一化、词表投影和损失已经形成完整计算链.

二维且 $\epsilon = 0$ 的 LayerNorm 把任意两个不相等分量归一化为相同或相反的符号模式，这使本例中若干归一化结果重复. 它是特意选择低维数带来的性质，不代表高维 Transformer 会将注意力信息普遍抹除. 改变归一化位置、增加维数或使用正 ϵ，数值路径都会相应改变. [2]

6.8　训练及推理接口

6.8.1　分类概率模型

Transformer 特征主干接收连续表示并返回连续表示. 完整语言模型还需要分词器、嵌入、位置机制、输出词表层、目标函数与生成循环. 仅得到 $[B, T, d]$ 输出不意味着已经定义了词元概率，更不意味着已经实现训练和推理.

训练时，源文本 x 和真实目标 y 定义条件似然

$$p_\theta(y \mid x) = \prod_{t=1}^{T} p_\theta(y_t \mid y_{<t}, x). \tag{6.51}$$

教师强制（Teacher Forcing）使用真实目标前缀，右移后的目标输入与因果掩码使各位置可并行计算. 目标为 (A, B, EOS) 时，输入为 (BOS, A, B)，同位置输出分别监督三个目标. 填充标签应排除，提示或条件前缀是否监督则由目标定义决定.

生成时用模型已选词元替代真实前缀. 编码器记忆可复用，解码器逐步追加目标词元，并在 EOS 或长度上限处终止. 缓存可以避免重复计算历史键值，但不会把自回归决策依赖变成完全并行. 缓存正确性要求位置编号、掩码、精度和参数与完整前向一致.

算法 6.1　教师强制的训练前向

输入：源词元、含 EOS 的目标词元、模型参数、目标有效长度. **输出：**有效目标位置的平均损失. **状态：**源记忆、各目标层表示及监督位置集合.

1. 对源序列构造填充掩码，计算嵌入和位置表示，经编码器得到源记忆.
2. 构造相对于目标右移一位的解码器输入：首位置为启动词元，其余位置取前一个真实目标；保持输入与目标逐位置配对.

[2] 二维例题适合检验计算顺序，不适合据此评价归一化的表达能力或训练效果. 结构示例与效果证据必须区分.

3. 构造目标因果掩码、目标填充掩码和交叉注意力源掩码，依次执行全部解码器层.

4. 经规定的末端归一化及词表投影得到分数，只在选中目标位置累计交叉熵并按有效位置数归一化.

5. 本次前向在全部目标位置处理完毕时结束；若有效监督数为零，跳过该批或按明确规则报告，而不进行零分母平均.

不变量：位置 t 的预测只依赖源输入与 $y_{<t}$；显式右移与库内部的下一词元配对只能采用一套，不能重复移位.

算法 6.2　条件生成的自回归执行

输入：源序列、启动与终止词元、最大生成长度、固定模型和词元选择策略. **输出：**每条样本的生成词元序列. **状态：**源记忆、已生成前缀、可选键值缓存、样本终止标志.

1. 关闭训练随机性，编码源序列一次；将目标前缀初始化为启动词元，各样本标为未终止.

2. 对未终止样本计算当前最后有效目标位置的分布. 使用缓存时同步更新位置编号、可见性和当前层键值.

3. 按规定策略选择下一词元并追加到前缀；生成 EOS 的样本标为终止，后续不再追加普通生成内容.

4. 若全部样本终止或达到长度上限，返回结果；否则回到步骤二.

不变量：缓存内容与当前模型参数及前缀对应；终止样本不再改变其已生成答案. 模型前向、词元选择和停止策略分别承担不同职责.

6.8.2　正则化及评价口径

训练中的**随机失活**（Dropout）可以作用于注意力权重、FFN 中间表示或残差增量. 按保留概率校正的 Dropout 在期望上保持对应分量的尺度，但一次采样会改变前向输出；注意力权重经过 Dropout 后，单次实现的行和也不必仍为一. 进行确定性结构对照时必须关闭随机性，或对齐全部随机状态.

标签平滑将独热目标替换为 $q = (1-\eta)e_y + \eta u$，u 为规定的参考分布. 损失为 $-\sum_v q_v \log p_v$，与普通真实标签负对数似然不同. 若训练记录平滑损失而验证记录未平滑损失，不能直接将二者差值解释为泛化间隙.

训练集用于参数更新，验证集用于选择检查点和停止时机，测试集用于最终评价.

固定小语料中的损失下降或少量正确译文，只能支持有限范围内的优化与行为观察. 精确匹配会把合法同义表达判错，字符或词片段指标也不能覆盖全部语义质量；评价指标与划分应在专门评估中明确. 注意力热力图可用于定位读取模式，但仍不构成独立的因果归因.

6.8.3　结构等价性及制品边界

从原理实现迁移到框架时，应先对齐参数，再对齐投影布局、归一化位置、激活定义、偏置、掩码和末端输出层. 两个独立随机初始化的网络不具有直接数值可比性. 形状相同只是必要条件，前向和反向应在指定精度容差内比较.

布尔掩码的真值方向是接口约定，不能脱离具体接口解释. 将“允许读取”矩阵转换为“禁止读取”矩阵时需要取反；加性掩码则直接作用于分数. 优化提示也不能替代对实际可见性矩阵的核验. 错误提示可能使后端选择不适用的执行路径，故接口封装应显式保存掩码语义.

模型制品还包含非训练参数状态，例如固定位置编码、频率、最大长度及归一化常数. 可保存这些状态，也可按固定配置精确重建；只保存可训练权重并不能普遍保证重载一致. 现成翻译模型的分词器、特殊词元、位置机制与权重是一组配套对象，不应与另一个原理模型的整数输入混用.

因果性检验可以固定参数并关闭随机性，扰动未来目标，检查更早位置输出是否保持不变；填充隔离检验可改变被屏蔽位置的内容，检查有效输出；归一化检验则应覆盖常量向量和很小方差. 这些证据验证数学与接口一致性，不直接证明训练质量、吞吐或生产可用性，后者还需要相应的数据和系统验证.

第七章　位置表示

词元嵌入提供离散身份的连续表示，注意力机制则根据内容在可见位置之间交换信息；二者都不会自动规定一套独立、可控的位置坐标. 第五章《注意力机制》已经建立缩放点积注意力，并证明无显式位置输入时的排列等变条件. 本章由这一结构性质出发，解释顺序信息为什么仍需明确进入模型.

本章先区分排列等变性与排列不变性，再推导可学习绝对位置、正弦位置编码和旋转位置编码，最后比较位置进入输入、注意力分数与查询键变换的不同方式. 超出训练长度后的频率调整与能力评估将在第三十七章《长上下文建模》讨论.

7.1　位置表示的必要性

7.1.1　等变性及不变性

设 $X \in \mathbb{R}^{T\times d}$ 包含一条序列的向量. 置换矩阵 Π 的每行每列恰有一个一，其余元素为零，满足 $\Pi^{\mathsf{T}}\Pi = I$. ΠX 重排输入行.

若序列变换 F 满足

$$F(\Pi X) = \Pi F(X), \tag{7.1}$$

则称其具有**排列等变性**（Permutation Equivariance）：输入如何重排，输出就对应地重排. 若某个汇总函数 g 满足 $g(\Pi X) = g(X)$，则称其具有**排列不变性**（Permutation Invariance）. 例如对所有行求和的不变性，与逐位置表示的等变性是不同性质.

等变并不意味着输出不能依赖其他词元. 它意味着在相同的无顺序规则下交换词元位置，模型只能相应地交换输出，不能仅凭交换后的数组行号获得原本不存在的顺序语义. 对于“甲帮助乙”与“乙帮助甲”，仅把词元作为可交换元素处理，难以表达由顺序确定的角色差异.

7.1.2　无位置自注意力的排列等变性

第五章《注意力机制》已经证明：各位置共享投影、不含显式位置输入、无随机算子，并且可见性关系随内容同步置换时，自注意力满足式 (7.1). 这项结论不是注意力计算的定义，而是由投影、行 Softmax 与加权汇总共同导出的结构性质.

7.1.3　掩码改变命题的前提

若注意力带加性掩码 M，允许位置取零，禁止位置取负无穷，则正确的等变关系为

$$F(\Pi X;\Pi M\Pi^{\mathsf{T}})=\Pi F(X;M). \tag{7.2}$$

也就是说，内容重排时，原来哪些位置允许互相读取的关系也要同步重排．每一行还必须有至少一个允许位置，否则归一化本身未定义．

固定的下三角因果掩码已经携带了前后可见性信息．若仅交换词元而仍使用原下三角掩码，通常有 $\Pi M\Pi^{\mathsf{T}}\neq M$，上面的无位置等变证明不再适用．因此不能声称“只要没有显式位置编码，任何因果模型都完全不能感知顺序”．显式位置表示提供了额外的可控坐标机制，但掩码与边界本身也会影响顺序信息．

加性位置向量矩阵记为 P_{pos}．词元重排但位置固定时，输入变为 $\Pi X+P_{\mathrm{pos}}$，而非 $\Pi(X+P_{\mathrm{pos}})$．二者一般不同，这正是位置编码改变上述对称性的方式．若连位置向量也跟随词元一起移动，则只是重新编号同一组带坐标元素，并未实现“交换词元但保留位置”的操作．

7.2　绝对位置及正弦编码

7.2.1　可学习位置表及加性组合

一种直接方法是为每个位置学习向量 $p_t\in\mathbb{R}^d$，并形成

$$x_t^{(0)}=\alpha E_{k_t,:}^{\mathsf{T}}+p_t. \tag{7.3}$$

词元身份与位置拥有相同宽度，因而可以相加；它们不必分别占用互不重叠的坐标轴．模型通过下游变换学习如何利用这两类信号．若位置表只定义到 $T_{\max}-1$，超出范围后没有可直接读取的训练表项，必须明确新的构造方法．

缩放系数 α 调整两类信号的相对尺度．原始 Transformer 使用 $\alpha=\sqrt{d}$，并以正弦函数构造位置向量 [6]．这是一项与初始化和架构配套的选择，不是所有模型都必须照搬的常数．嵌入的参数尺度、归一化位置和位置表示方式改变后，应重新考察相加后的统计量．

7.2.2　多频率正弦及余弦

取偶数宽度 d，位置编号 $t=0,1,\dots$，令

$$\omega_r=10000^{-\frac{2r}{d}},\qquad r=0,\dots,\frac{d}{2}-1. \tag{7.4}$$

使用从零开始的分量索引，**正弦位置编码**（Sinusoidal Positional Encoding）的向量定义为

$$(p_t)_{2r}=\sin(t\omega_r),\qquad (p_t)_{2r+1}=\cos(t\omega_r). \tag{7.5}$$

每一对分量描述同一个角频率下的二维相位. 频率随 r 几何递减，因此前面的维度变化较快，后面的维度变化较慢；这种组合提供多个距离尺度，而不是为每个位置单独分配参数. 奇数宽度需要明确最后一维的处理，不能直接假定所有分量都能配成二维对.

单个分量是周期函数；位置向量由多个频率共同构成. 由每对分量的平方和为一，有 $\|p_t\|_2^2 = \dfrac{d}{2}$，与 t 无关. 位置编码的范数保持不变，但不同位置之间的方向随频率组合变化.

取 $d = 4$，两个频率为 1 和 0.01，便可直接计算：

t	$\sin t$	$\cos t$	$\sin(0.01t)$	$\cos(0.01t)$
0	0	1	0	1
1	0.841471	0.540302	0.010000	0.999950
2	0.909297	−0.416147	0.019999	0.999800

相邻位置在快频率对中变化明显，在慢频率对中变化较小. 表中数值来自式 (7.5) 的直接计算，不代表学习后某个维度已经承担特定语言功能.

7.2.3 相对位移的线性结构

对一个频率对，令 $s_t = (\sin(t\omega), \cos(t\omega))^{\mathsf{T}}$. 由三角和角公式，

$$\sin((t+\Delta)\omega) = \sin(t\omega)\cos(\Delta\omega) + \cos(t\omega)\sin(\Delta\omega), \tag{7.6}$$

$$\cos((t+\Delta)\omega) = \cos(t\omega)\cos(\Delta\omega) - \sin(t\omega)\sin(\Delta\omega). \tag{7.7}$$

合并为矩阵形式：

$$s_{t+\Delta} = \begin{bmatrix} \cos(\Delta\omega) & \sin(\Delta\omega) \\ -\sin(\Delta\omega) & \cos(\Delta\omega) \end{bmatrix} s_t. \tag{7.8}$$

对所有频率使用分块对角矩阵，即得到 $p_{t+\Delta} = A_\Delta p_t$. 变换只依赖位移 Δ，不依赖起点 t. 这给出了“相对位置具有线性结构”的精确含义.

未经过其他投影时，两个位置向量的内积进一步满足

$$p_i^{\mathsf{T}} p_j = \sum_r \big[\sin(i\omega_r)\sin(j\omega_r) + \cos(i\omega_r)\cos(j\omega_r)\big] \tag{7.9}$$

$$= \sum_r \cos((j-i)\omega_r). \tag{7.10}$$

但加性位置进入注意力后，完整分数不只包含这一个内积. 若内容列向量为 x_i, x_j，简记 $A = W_Q W_K^{\mathsf{T}}$，则未缩放分数为

$$(x_i + p_i)^{\mathsf{T}} A (x_j + p_j) = x_i^{\mathsf{T}} A x_j + x_i^{\mathsf{T}} A p_j + p_i^{\mathsf{T}} A x_j + p_i^{\mathsf{T}} A p_j. \tag{7.11}$$

内容与位置交叉项、以及一般的投影 A，意味着这个完整分数不必只依赖位置差. 正弦编码具有相对位移结构，与完整注意力分数严格只含相对位置，是两个不同命题.

7.2.4　可计算长度及可泛化长度

正弦函数可以在未出现过的位置继续求值，因而不会像有限查表那样立即发生索引越界．然而，模型可能只在有限相位组合和相对距离上接受训练．能够生成数值正确的位置向量，不等于后续层已经学会利用远超训练范围的信号．

在有限精度下，相位乘法、三角函数求值和参数存储也会影响结果．不能把“不同位置近似独特”扩展为“任意长整数位置都能无误区分”的保证．长上下文能力需要结合训练分布、频率方案和实际任务检验，第三十七章《长上下文建模》将进一步处理这些问题．

7.3　旋转位置编码的构造

7.3.1　查询键旋转

旋转位置编码（Rotary Position Embedding，RoPE）把位置信息作用于投影后的查询和键 [30]．先按不带旋转的位置内容计算 $q_i = W_Q^{\mathsf{T}} x_i$、$k_j = W_K^{\mathsf{T}} x_j$，其中此处 x_i, x_j 为列向量，再令

$$\widetilde{q}_i = R_i q_i, \qquad \widetilde{k}_j = R_j k_j. \tag{7.12}$$

位置改变向量在若干二维平面中的方向，而不必向输入再增加一个独立的位置向量．标准的这一构造旋转查询和键，值向量仍按内容投影处理；若某个架构还对值应用变换，应按其实际定义另行分析．

我们先从二维欧氏旋转本身推导，而不直接使用高维结果．角度为 θ 的逆时针旋转矩阵为

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}. \tag{7.13}$$

对列向量 $z = (z_0, z_1)^{\mathsf{T}}$，旋转后为

$$R(\theta) z = \begin{bmatrix} z_0\cos\theta - z_1\sin\theta \\ z_0\sin\theta + z_1\cos\theta \end{bmatrix}. \tag{7.14}$$

两分量并非各自独立缩放，而是交叉混合．漏掉其中的负号或交换配对维度，都会改变旋转方向和注意力分数．

7.3.2　正交性及旋转的组合

将式 (7.13) 转置并相乘，利用 $\cos^2\theta + \sin^2\theta = 1$，得到

$$R(\theta)^{\mathsf{T}} R(\theta) = I, \qquad R(\theta)^{\mathsf{T}} = R(-\theta). \tag{7.15}$$

因此 $\|R(\theta)z\|_2 = \|z\|_2$. 旋转不会通过放大向量长度引入位置，而是通过相对方向改变交互.

将两个旋转矩阵逐项相乘，可得

$$R(a)R(b) = \begin{bmatrix} \cos a\cos b - \sin a\sin b & -(\cos a\sin b + \sin a\cos b) \\ \sin a\cos b + \cos a\sin b & \cos a\cos b - \sin a\sin b \end{bmatrix} \tag{7.16}$$

$$= R(a+b). \tag{7.17}$$

故 $R(a)^{\mathsf{T}}R(b) = R(b-a)$. 这是 RoPE 内积只显式依赖位置差的代数基础，而不只是对旋转图形的直观描述.

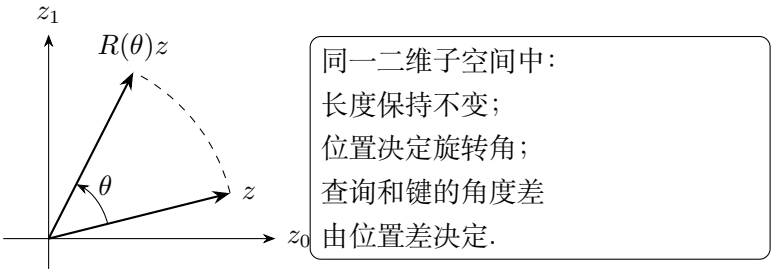


图 7.1: 二维旋转的几何含义. 图为结构示意，实线向量与虚线圆弧表示旋转关系，不是训练数据的投影.

7.3.3 相对内积推导

对频率 ω，查询位置 i 使用 $R(i\omega)$，键位置 j 使用 $R(j\omega)$. 于是

$$\widetilde{q}_i^{\mathsf{T}}\widetilde{k}_j = [R(i\omega)q_i]^{\mathsf{T}}[R(j\omega)k_j] \tag{7.18}$$

$$= q_i^{\mathsf{T}}R(i\omega)^{\mathsf{T}}R(j\omega)k_j \tag{7.19}$$

$$= q_i^{\mathsf{T}}R((j-i)\omega)k_j. \tag{7.20}$$

令 $\Delta = j - i$，$q_i = (q_0, q_1)^{\mathsf{T}}$，$k_j = (k_0, k_1)^{\mathsf{T}}$. 继续展开右侧，得到

$$\widetilde{q}_i^{\mathsf{T}}\widetilde{k}_j = (q_0k_0 + q_1k_1)\cos(\Delta\omega) + (q_1k_0 - q_0k_1)\sin(\Delta\omega). \tag{7.21}$$

第一项由普通内积调制，第二项由带方向的交叉组合调制. 位置差为零时，正弦项消失，分数恢复为未旋转内积. 位置差取反时，余弦项不变而正弦项变号，这说明位置信号可以区分方向.

旋转内积的相对位置性质

查询和键使用相同的频率组与配对约定时，$R_i^{\mathsf{T}}R_j = R_{j-i}$. 因此保持内容向量不变并同时平移两个位置，不改变其旋转内积；改变内容、可见性或频率配置则不在这一恒等式的保证范围内.

这里的“相对”是对显式位置因素的陈述，内容向量 q_i, k_j 仍然影响分数．对于深层网络，这些内容向量又可能已经依赖边界、掩码和先前上下文．因此不能把式 (7.20) 解释为整网输出只依赖词元之间的距离，也不能推断任意移动整个输入后输出必然不变．

7.3.4　多频率高维扩展

若单头查询和键宽度 d_k 为偶数，把坐标划分为 $\dfrac{d_k}{2}$ 个二维子空间．每对使用自己的频率 ω_r，定义

$$R_t = \operatorname{diag}\big(R(t\omega_0), \dots, R(t\omega_{\frac{d_k}{2}-1})\big). \tag{7.22}$$

这里 diag 表示把二维矩阵排成分块对角矩阵，而不是只取其对角元素．各块独立旋转，所以

$$R_i^{\mathsf{T}} R_j = R_{j-i}, \qquad \widetilde{q}_i^{\mathsf{T}} \widetilde{k}_j = \sum_{r=0}^{\frac{d_k}{2}-1} (q_i^{(r)})^{\mathsf{T}} R(\Delta\omega_r) k_j^{(r)}. \tag{7.23}$$

一种基本频率配置写为 $\omega_r = \beta^{-\frac{2r}{d_k}}$，其中 $\beta > 1$．频率基数、实际旋转维数与配对顺序都属于模型定义，不能仅凭隐藏维度猜测．

若只旋转前 d_r 个维度，且 d_r 为偶数，则其余维度保持不变，分数分解为旋转部分与普通内积部分：

$$\widetilde{q}_i^{\mathsf{T}} \widetilde{k}_j = (q_i^{\mathrm{rot}})^{\mathsf{T}} R_\Delta k_j^{\mathrm{rot}} + (q_i^{\mathrm{pass}})^{\mathsf{T}} k_j^{\mathrm{pass}}. \tag{7.24}$$

频率指数的分母应与实际采用的方案一致．把全部头宽与旋转宽度静默互换，可能在张量形状正确的情况下改变所有频率．

高维旋转通常可以逐对计算，无需物化 $d_k \times d_k$ 稠密矩阵．每个位置只需线性数量的乘加．相邻配对与前后半段配对可以通过固定的坐标置换建立等价关系，但同时必须变换投影权重的布局．只替换旋转程序而不处理已有权重，会得到不同函数．

算法 7.1　逐对执行 RoPE

输入：查询或键张量 $Z \in \mathbb{R}^{B\times T\times d_k}$，位置编号 $I \in \mathbb{Z}^{B\times T}$，偶数旋转宽度 $d_r \leqslant d_k$，频率 $\omega_0, \dots, \omega_{\frac{d_r}{2}-1}$；各维度均为有限长度．此处采用相邻坐标配对．
输出：同形张量 $\widetilde{Z}$．**状态：**输出数组和当前二维原始分量 a, b，二者在写回前均须保存．

1. 对每个样本索引 $b_0 = 1, \dots, B$ 和位置列 $t = 0, \dots, T-1$，读取实际位置 $m = I_{b_0 t}$．
2. 对每对 $r = 0, \dots, \dfrac{d_r}{2} - 1$，令 $\theta = m\omega_r$，$c = \cos\theta$，$s = \sin\theta$；保存 $a = Z_{b_0,t,2r}$、$b = Z_{b_0,t,2r+1}$．
3. 写入 $\widetilde{Z}_{b_0,t,2r} = ac - bs$ 与 $\widetilde{Z}_{b_0,t,2r+1} = as + bc$．不得用已更新的第一分量代替

原始 a.

4. 将坐标 $d_r, \ldots, d_k - 1$ 原样复制；全部样本、位置及配对处理完毕即停止.

不变量：每对旋转前后平方和相同；未旋转坐标相同. 对查询和键分别调用时，频率和配对必须一致，位置编号各取对应位置.
按标量运算计，旋转计算量为 $O(BTd_r)$，输出存储为 $O(BTd_k)$；频率表为 $O(d_r)$. 若预计算全部位置的三角函数，需另计表的存储. 多头情况下对各头独立使用相同过程，并计入头数因子.

算法 7.1 给出了与分块旋转矩阵等价的执行顺序. 它没有构造稠密矩阵，也没有规定特定框架接口；数值例题将用独立内积公式核对其结果.

7.4 RoPE 梯度

7.4.1 相对位置分数

取一个二维子空间，频率 $\omega = 1$，查询位置 $i = 1$、键位置 $j = 3$，内容向量为

$$q_i = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \qquad k_j = \begin{bmatrix} 3 \\ -1 \end{bmatrix}. \tag{7.25}$$

角度以弧度计. 分别执行旋转：

$$\widetilde{q}_i = \begin{bmatrix} \cos 1 - 2\sin 1 \\ \sin 1 + 2\cos 1 \end{bmatrix} \approx \begin{bmatrix} -1.142640 \\ 1.922076 \end{bmatrix}, \tag{7.26}$$

$$\widetilde{k}_j = \begin{bmatrix} 3\cos 3 + \sin 3 \\ 3\sin 3 - \cos 3 \end{bmatrix} \approx \begin{bmatrix} -2.828857 \\ 1.413353 \end{bmatrix}. \tag{7.27}$$

直接相乘给出 $\widetilde{q}_i^{\mathsf{T}} \widetilde{k}_j \approx 5.948935$. 现在改用位置差 $\Delta = 2$，普通内积为 $1\times 3 + 2\times(-1) = 1$，交叉组合为 $2 \times 3 - 1 \times (-1) = 7$，故

$$q_i^{\mathsf{T}} R(2) k_j = \cos 2 + 7 \sin 2 \approx 5.948935. \tag{7.28}$$

两条计算路径相同. 如果作为宽度为 2 的缩放点积注意力分数，还须除以 $\sqrt{2}$，得到约 4.206532. 该缩放来自注意力定义，不属于旋转矩阵本身.

将两个位置同时平移为 $i' = 6, j' = 8$，保持内容向量、频率与配对不变，差值仍为 2，结果便保持不变. 反过来，只改变键位置为 $j = 1$，差值变为零，内积恢复为 1. 例题分别检验了公共平移性质与同位置性质.

7.4.2　旋转的反向传播

位置和频率固定时，$\widetilde{q} = R_i q$ 是一个线性变换．由第三章《神经网络基础》的链式法则，若上游梯度为 $g = \dfrac{\partial \mathcal{L}}{\partial \widetilde{q}}$，则

$$\frac{\partial \mathcal{L}}{\partial q} = R_i^{\mathsf{T}} g = R_{-i} g. \tag{7.29}$$

反向使用逆旋转，且梯度范数保持不变．这一结论仅针对纯旋转模块；后续内积、Softmax 和其他层仍可能改变梯度尺度，不能据此声称完整注意力没有梯度消失或爆炸问题．

令一个二维内积 $s = q^{\mathsf{T}} R(\Delta\omega) k$．若把 ω 当作可微实数参数，则

$$\frac{\partial s}{\partial \omega} = \Delta \left[-(q_0 k_0 + q_1 k_1) \sin(\Delta\omega) + (q_1 k_0 - q_0 k_1) \cos(\Delta\omega) \right]. \tag{7.30}$$

这一表达式说明频率误差与距离尺度相关．基本 RoPE 中频率可以固定，不需要优化该导数；推导它是为了理解为什么位置很远时，小的频率变化也可能造成明显的相位偏移．

7.4.3　相对位置分数的非单调性

式 (7.21) 包含正弦与余弦，单个子空间的分数随距离振荡，而不必单调减小．例如 $q = k = (1, 0)^{\mathsf{T}}$ 时，分数就是 $\cos(\Delta\omega)$，在一个周期内既下降又上升．

多频率组合能够产生不同距离上的相位干涉，但其结果还受内容系数影响．因此不能将 RoPE 概括为“距离越远，任意查询和键的相关性就越低”．若讨论某种统计衰减或经验偏好，应另外说明向量分布、频率配置和测量条件；纯旋转恒等式本身不提供这样的逐样本保证．

7.5　位置方案的边界及一致性

7.5.1　位置机制的作用层次

在已建立的公式上，可以区分三类设计：加性绝对位置在输入表示中形成 $E_{k_t,:}^{\mathsf{T}} + p_t$；相对分数项直接将一个依赖 $j - i$ 的量加到匹配分数；RoPE 则先变换查询和键，使内积显式包含相对位移．这些方法都能携带顺序信息，但它们修改的计算对象不同．

例如直接加入相对偏置的分数可写为

$$s_{ij} = \frac{q_i^{\mathsf{T}} k_j}{\sqrt{d_k}} + b_{j-i}. \tag{7.31}$$

其中 b_{j-i} 不必与内容相乘，而 RoPE 的相对作用与内容系数交织．位置方法的选择需要考虑训练长度、所需相对关系与既有权重；保持张量尺寸相同不足以证明替换后模型等价．

方法	位置作用对象	基本代数性质	需明确的边界
可学习位置表	输入向量	每个位置独立参数	表范围与未训练位置
正弦加性编码	输入向量	位移对应线性变换	完整分数含交叉项
相对分数偏置	注意力分数	按相对位移增加项	位移索引与分组规则
RoPE	查询与键	旋转内积含相对位移	频率、配对与旋转宽度

这些类别描述位置如何进入某一项计算，并不要求整套架构只采用其中一种. 模型可以只旋转查询和键的部分维度，也可以只在部分注意力层使用 RoPE；不执行显式位置向量相加或查询键旋转的序列模块，还可能通过因果可见性、递推状态、卷积或其他带顺序的计算路径利用先后关系. 因此，“不使用 RoPE” “没有显式位置编码”与“模型不利用顺序”是三个不同判断.

7.5.2 位置编号的输入语义

向量的批量存储列号不一定等于模型位置. 使用左侧填充时，一个有效词元在数组中的列号可能随同批其他样本而变化. 如果模型预期有效词元从固定起点连续编号，应按有效位置生成相应编号；如果训练定义采用其他规则，则应复现该规则，不能以个人偏好更改.

多个文档打包在同一数组中时，还需同时确定文档边界、可见性与位置是否重置. 单独重置位置编号不会阻止跨文档读取；单独屏蔽跨文档注意力也不自动决定位置编号. 这些约定共同定义前向函数.

对于增量计算，若历史键已经按各自位置旋转，就应保留它们的位置语义. 新查询不能在每个生成步都错误地重新编号为零，否则与历史键形成的角度差将改变. 缓存索引、模型位置以及频率配置要相容；具体缓存结构将在推理部分展开.

7.5.3 位置构造规则

嵌入表依赖词表到编号的映射. 同一形状的表若搭配不同编号顺序，就会读出错误词元向量；新增词元时也需同步处理输入表、输出投影与特殊符号. 权重共享更要求这些对象指向同一语义集合.

位置函数即使没有可训练参数，其频率基数、宽度、索引起点与缩放规则仍属于模型定义. 可以保存完整的预计算位置表，也可以保存足以确定性重建它的配置；关键在于重载后计算相同，而不是规定所有实现必须使用同一种存储接口. [1]

验证应围绕具体性质展开：不含位置且可见性同步重排时满足等变关系；正弦位移满足式 (7.8)；RoPE 保持范数、满足相对内积并在统一平移下保持该内积. 最后再检查位置编号、频率配置与重载前后的输出一致性. 即使这些性质全部通过，也只证明位置计算与预期定义相容；长文本任务表现仍需独立证据.

[1] 预计算的正弦表属于可重建状态. 是否把它随权重保存是工程选择；若不保存，就必须保证构造公式、配置、精度与位置范围能被一致恢复.

第八章　自回归语言建模

一个语言模型需要同时回答两个不同问题：如何为序列分配概率，以及如何根据这些概率产生实际输出．前者由模型的条件分布与训练目标定义，后者由解码策略与终止规则定义．相同权重可以在贪心选择、随机采样和束搜索下产生不同文本；较低的预测损失也不保证自由生成时没有重复、遗漏或事实错误．

本章在第六章《Transformer 网络结构》基础上建立完整的自回归生成过程．从概率链式分解出发，依次说明因果网络如何产生下一词元分布、教师强制为何允许并行训练、模型输出如何被解码器变换与选择，以及 EOS 和长度预算如何界定最终交付．缓存只作为保持条件分布不变的状态复用机制出现，具体优化留待推理篇．

8.1　用条件分布定义序列

8.1.1　自回归分解的结构边界

设词表为有限集合 $\mathcal{V}$，词表大小为 V_{vocab}．对长度为 T 的词元序列 $x_{1:T}$，概率链式法则给出

$$p_\theta(x_{1:T}) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t}), \qquad \log p_\theta(x_{1:T}) = \sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}). \tag{8.1}$$

自回归（Autoregression）表示后续变量的分布以已经给定的变量为条件．链式分解本身不要求使用神经网络；限定上下文长度、共享各位置参数、采用因果 Transformer 才是具体建模选择．

对于给定提示或源输入 c 的条件生成，同样有

$$p_\theta(y_{1:T} \mid c) = \prod_{t=1}^{T} p_\theta(y_t \mid c, y_{<t}). \tag{8.2}$$

条件 c 可以是同一序列中的提示前缀，也可以通过编码器及交叉注意力提供．两种方式都能实现条件生成，不能把条件概率分解等同于某一种网络模块．

有限上下文模型实际计算的可能是 $p_\theta(x_t \mid x_{\max(1,t-C):t-1})$，其中 C 是可读取的上下文长度．它对完整历史施加了信息限制，仍可逐步构成一致的序列分布，但不再保

证利用更早内容. 裁掉前缀或改变位置编号会改变模型的条件，而不是仅改变内存占用.

符号	含义与约束
x_t, y_t, c	观测词元、生成词元与外部条件；词元取值属于 $\mathcal{V}$.
B, T, C	批量大小、序列位置数、上下文容量；不使用 T 表示温度.
d, V_{vocab}	隐藏宽度与词表大小.
z_t, p_t, q_t	当前词表分数、原始模型分布、经过解码变换的实际分布.
τ, k, ρ	正温度、Top-k 个数、Top-p 累积阈值；$\tau > 0$，$1 \leqslant k \leqslant V_{\text{vocab}}$，$0 < \rho \leqslant 1$.
$K, L_{\max}$	束宽与最多新增词元数，均为正整数.

8.1.2 可变长度序列及结束事件

仅对固定长度写式 (8.1)，尚未说明文本在哪里结束. 引入**序列结束符**（End of Sequence，EOS），将一条完整响应定义为非结束词元 $y_{1:T}$ 后跟 EOS：

$$p_\theta(y_{1:T}, \text{EOS} \mid c) = \left[\prod_{t=1}^{T} p_\theta(y_t \mid c, y_{<t})\right] p_\theta(\text{EOS} \mid c, y_{1:T}). \tag{8.3}$$

结束概率也是序列概率的一部分. 比较两个完整候选时，漏掉 EOS 会改变排名. 初始条件可以由提示前缀给出，也可以借助**序列开始符**（Beginning of Sequence，BOS）建立；并非所有模型都使用单独的 BOS 词元.

各步分布归一化并不自动保证生成以概率一在有限时间结束. 例如某类前缀之后 EOS 概率恒为零，就可能产生永不终止的路径. 如果每个未结束前缀上的 EOS 概率都有统一下界 $\epsilon > 0$，那么

$$\Pr(\text{前} t \text{ 步仍未结束}) \leqslant (1-\epsilon)^t \longrightarrow 0, \tag{8.4}$$

这是确保几乎必然终止的一种充分条件，真实模型与截断解码不一定满足. 实际系统因此仍需要明确的有限生成预算.

8.2 下一词元分布

8.2.1 GPT 数据流

生成式预训练 Transformer（Generative Pre-trained Transformer，GPT）以仅解码器的因果网络为常见结构. 输入编号先经嵌入与位置处理变成隐藏向量，再通过多层因果注意力、逐位置前馈变换、残差及归一化，最后映射为词表上的实数分数. 具体模型的归一化、位置方案与前馈形式属于结构配置，不能由 GPT 缩写推断所有细节.

取输入前缀 $x_{1:t}$，末位隐藏行向量为 $h_t \in \mathbb{R}^{1\times d}$．词表输出头计算

$$z_t = h_t W_{\mathrm{lm}} + b, \qquad p_\theta(x_{t+1} = v \mid x_{\leqslant t}) = \frac{\exp z_{t,v}}{\sum_{u\in\mathcal{V}} \exp z_{t,u}}, \tag{8.5}$$

其中 $W_{\mathrm{lm}} \in \mathbb{R}^{d\times V_{\mathrm{vocab}}}$，$b \in \mathbb{R}^{V_{\mathrm{vocab}}}$ 可按结构省略．z_t 是**未归一化分数**（Logits），并非概率；Softmax 之后才得到条件分布．若输入嵌入表为 $E \in \mathbb{R}^{V_{\mathrm{vocab}}\times d}$，权重共享可设置 $W_{\mathrm{lm}} = E^{\mathsf{T}}$，但这是显式参数约束，不是维度相容的必然结果．

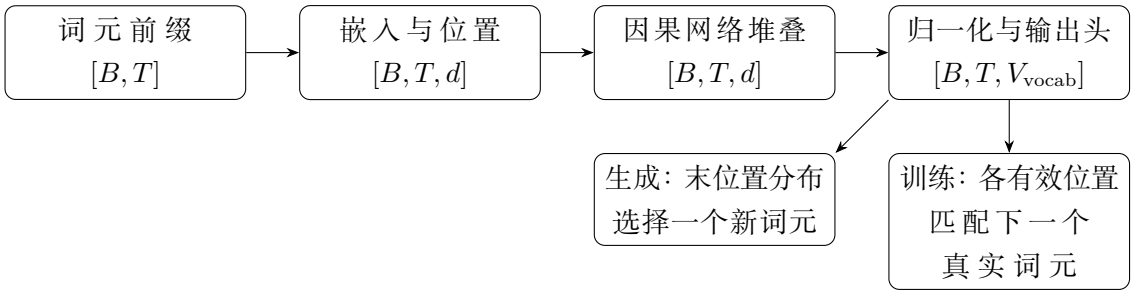


图 8.1: 相同因果网络的两种使用方式．训练利用所有有效位置的已知目标，生成读取当前前缀末位的分布．

8.2.2　因果性及前缀一致性

第五章《注意力机制》定义的因果注意力通过因果掩码使位置 t 只能读取不晚于 t 的信息．如果每层其他操作均逐位置执行，则完整序列前向在位置 t 得到的条件分数，与只输入前缀 $x_{1:t}$ 时末位的分数在数学上相同．

这个结论需要相同的参数、位置编号、有效性掩码和运行模式．随机失活开启时，两次调用可能使用不同随机掩码；浮点内核也可能因张量形状不同产生舍入差异．上述因素不应与理论上的未来信息泄漏混为一谈．因果掩码不会自动修正标签对齐错误，下面需要另外定义预测对象．

8.3　教师强制及移位监督

8.3.1　输入目标错位

教师强制（Teacher Forcing）在训练时始终给定数据中的真实前缀，再预测其后一个真实词元．设原序列为 (BOS, 我, 学习, 模型, EOS)，一个明确的训练对为

$$\begin{aligned} \text{输入} &= (\text{BOS}, \text{我}, \text{学习}, \text{模型}), \\ \text{目标} &= (\text{我}, \text{学习}, \text{模型}, \text{EOS}). \end{aligned} \tag{8.6}$$

每个输入位置可以读取自身及过去输入，但其目标是下一位置的词元．为避免“左移”“右移”因参照物不同引起歧义，应直接说明索引关系：若输入位置 t 存放 x_t，则

该位置分数监督 x_{t+1}. 构造解码器输入时也可以说 “在目标前面放入 BOS，再移除最后一个词元”，二者是同一个对齐关系.

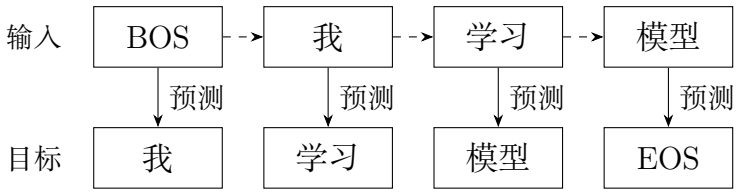


图 8.2: 监督索引与因果前缀. 竖直箭头给出各位置的预测对象；横向虚线表示前缀依赖，实际注意力仍读取全部允许历史.

移位可以由数据整理器完成，也可以由模型损失内部完成. 前一种接口接收已经对齐的输入和目标；后一种接口可能接收相同的完整词元数组，再在内部使用前 $T-1$ 个分数与后 $T-1$ 个目标. 两处同时移位会变成跨两步预测，两处都不移位则可能学到复制当前位置. 应以明确索引式作为协议，不依赖参数名猜测.

8.3.2 平均负对数似然

给定训练文档 $x^{(b)}$，最大化其序列似然等价于最小化各条件负对数概率之和. 设 $m_{bt} \in \{0,1\}$ 标记位置 (b,t) 是否提供有效监督，则

$$\mathcal{L}(\theta) = -\frac{1}{N} \sum_{b,t} m_{bt} \log p_\theta(x_{t+1}^{(b)} \mid x_{\leqslant t}^{(b)}), \qquad N = \sum_{b,t} m_{bt} > 0. \tag{8.7}$$

填充目标通常不计入 N，文档末尾的 EOS 若属于训练目标则计入. 指令响应训练中，提示位置可以参与条件计算而不承担损失；这相当于选择监督区域，不能通过屏蔽所有提示键来替代.

若批次含不同有效长度，应聚合负对数似然总和后除以有效目标总数.两个批次有效目标数分别为 10 与 90、平均损失分别为 1 与 3 时，整体平均是 $\dfrac{10 \cdot 1 + 90 \cdot 3}{100} = 2.8$，不是 $\dfrac{1+3}{2} = 2$. 按序列平均和按词元平均也是不同的目标权重设计.

在使用自然对数时，**困惑度**（Perplexity，PPL）为 $\exp \mathcal{L}$. 它概括同一数据和分词规则下的条件预测损失，不是模型候选答案数量，也不能跨不同分词器直接比较.

8.3.3 因果训练并行性

训练样本已经给出全部真实目标，所以位置 $1, \ldots, T$ 所需的合法前缀同时可用. 因果掩码阻止未来泄漏，矩阵运算仍可在同一次网络调用中处理全部位置. 各网络层之间保留先后依赖，所谓并行是位置维度的批量计算，并不是所有运算在时间上同时完成.

自由生成时，模型还不知道 y_t，而计算 $p(y_{t+1} \mid c, y_{\leqslant t})$ 需要先确定它．生成因此沿实际选出的前缀逐步推进．推测解码可以批量提出并验证候选，但只有保留的前缀才成为后续条件；它不废除自回归分解．

算法 8.1　移位目标的教师强制损失

输入：各文档的词元序列、文档边界、填充标记和监督区域．**输出：**标量损失与有效目标数．**状态：**当前固定参数 θ；本算法只计算一次目标，不推进优化器．

1. 在每个合法文档范围内建立输入位置 x_t 与目标 x_{t+1} 的对应关系；依据是否包含 BOS、EOS 明确首尾监督．
2. 构造因果和填充可见性，标记有效目标 m_t．跨文档是否连接必须遵守事先定义的目标，不在批次边界临时决定．
3. 运行因果网络，得到与输入位置对应的分数；确认移位仅在数据侧或损失侧执行一次．
4. 累加有效位置的负对数概率和以及目标数 N；$N = 0$ 时返回明确的无监督批次状态，不执行除零或伪造零梯度．
5. 返回总和除以 N．各目标只能读取其前缀，填充位置不贡献监督，输入和目标索引保持图 8.2 的关系．

8.4　真实前缀及生成前缀

8.4.1　暴露偏差的含义及限度

训练时前缀来自数据分布，生成时前缀来自模型和解码策略．有限模型在少见或偏离训练分布的前缀上可能预测较差，一步错误又会成为下一步的条件．这种训练与使用条件的差别通常称为**暴露偏差**（Exposure Bias），是序列训练研究的重要动机之一 [31]．

但是这不意味着教师强制的最大似然目标在数学上错误．对真实序列分布 P 和模型分布 p_θ，在支持集及可积性条件满足时，

$$\mathbb{E}_{x\sim P}[-\log p_\theta(x)] = H(P) + D_{\mathrm{KL}}(P\|p_\theta). \tag{8.8}$$

因序列链式法则，教师强制恰好计算这一序列负对数似然．若模型准确表达了真实条件分布，其生成分布也可以一致．实际困难来自有限数据、模型失配、分布外前缀，以及最终任务指标与局部概率目标之间的差异．不能把所有生成错误都归因于暴露偏差．

人为混入模型预测前缀，会改变训练数据条件，并且仍需判断原目标是否与被修改的前缀相容．它不是不加约束的通用修复．序列级训练、偏好优化和可验证奖励从

不同角度调整行为，后训练篇将讨论其目标与条件.

8.4.2 生成误差累积

如果在一组明确的条件下，每一步出现某类错误的条件概率不超过 ϵ，则由并集界，前 T 步至少一次错误的概率不超过 $T\epsilon$，且上界至多取一. 在独立且等概率的特殊模型中，该概率为 $1-(1-\epsilon)^T$. 真实生成中的错误相互影响，通常不满足独立同分布假设；教师强制平均损失也不能直接推出每个生成前缀都有这样的逐步错误上界.

需要区分预测某个指定词元的概率与完成任务的正确率. 某个问题可以有许多正确表述；合法同义词的概率较低不一定是任务错误，重复的高概率短句也不一定完成了任务. 自由生成评价应同时关注内容、约束、终止及整体连贯性.

8.5 幻觉、事实性及证据支持

8.5.1 事实性判定基准

幻觉（Hallucination）通常指生成结果包含相对于任务规定参照不受支持或错误的内容 [32]. 这个词描述输出与参照之间的可检验关系，不表示模型具有人的感知、信念或欺骗意图. 参照可以是指定上下文、带版本与时间的外部事实、工具实际返回值，或任务明确给出的约束；若不先说明参照，“是否幻觉”便可能没有唯一判定.

例如，某个结论可能碰巧符合外部世界，却没有得到当前证据包支持；它在答案正确性上可能通过，在证据忠实度上仍失败. 反过来，模型也可能忠实复述一份已经过期的材料，因而有证据支持却不符合问题要求的当前事实. 这两个维度不能合并为一个模糊的“看起来正确”.

现象	相对于已声明参照的判定
事实错误	可核实声明与目标时间、适用范围内的有效事实冲突.
上下文矛盾	回答声明与给定证据直接冲突，或忽略证据中的必要限定.
无依据生成	回答新增了可核实声明，但给定证据既不支持它，也不足以推出它.
虚构归因	回答捏造来源、引用、工具结果或把真实来源错误地说成支持某一声明.

这些类型可以重叠：虚构引用所支持的结论也可能同时是事实错误. 漏答、文风不佳、格式错误和正确拒答则不自动属于幻觉；它们应按各自任务维度评价，避免一个宽泛术语吞并所有生成失败.

8.5.2　词元概率及事实性

模型训练直接约束的是式 (8.7) 中的下一词元条件概率，而不是每个自然语言声明的真值．对回答中的声明 A_i、证据集合 E 和目标时间 t^*，模型给出的

$$p_\theta(y_j \mid c, y_{<j}) \quad \text{与} \quad \Pr(A_i \text{ 在} E, t^* \text{ 下成立}) \tag{8.9}$$

是不同对象，前者不会自动成为后者的校准估计．一个常见而流畅的错误说法可以由高概率词元组成；一个罕见但正确的名称也可能具有较低词元概率．

幻觉可能来自训练语料的遗漏、冲突或过时，有限模型与优化误差，问题歧义或证据缺失，以及生成前缀偏离后形成的误差反馈．后训练若主要奖励完整、肯定和流畅的回答，也可能使模型在证据不足时仍倾向作答．外部检索又会引入漏召回、错误排序、版本冲突和证据未被正确利用等系统原因．因此，观察到错误答案后，不能仅凭文本断定故障发生在预训练、解码器还是检索层．

降低温度会集中已有分布，但不会增加知识或执行事实核验；若最高概率路径本来就是错误的，贪心或低温只会更稳定地返回它．提高温度则可能增加输出变化和低概率错误，但同样不是幻觉的必要条件．温度、Top-k 和 Top-p 都不是“真实性旋钮”．

8.5.3　幻觉缓解机制

减少幻觉应把生成系统拆成可检查阶段．首先规定允许使用的知识范围、目标时间和不可回答条件；其次检索带来源、版本与权限的证据，并在生成时限制结论范围、保留声明到来源的映射；随后对关键声明执行语义支持检查、规则检查、计算或工具验证；证据不足或冲突无法解决时，输出限定回答、部分回答或拒答．RAG 可以提供新鲜证据，却不能在漏召回、错误证据或生成器忽略证据时自动消除幻觉，完整机制见第十三章《检索增强生成》．

评估时应同时包含可回答、不可回答和证据冲突样例，把回答拆成可核实声明，并分别标记支持、冲突与未知．答案正确率、声明级忠实度、引用精确率、正确拒答率和过度拒答率的分母不同，应分别报告；具体测量方法见第九章《模型评估》．只有平均语言模型损失或词面相似度，无法说明系统的幻觉率．

流畅生成与可靠断言是两项能力

自回归模型可以对语言形式给出连贯的条件分布，但事实判断还需要明确参照、可用证据、时间范围和验证规则．降低随机性可以减少答案波动，却不能把语言概率转换成事实保证；可靠系统必须允许在证据边界处收缩结论或拒答．

8.6 采样分布变换

8.6.1 温度的概率形式

温度缩放（Temperature Scaling）在生成时使用 $\tau > 0$ 改变分数尺度：

$$p_v^{(\tau)} = \frac{\exp(\frac{z_v}{\tau})}{\sum_u \exp(\frac{z_u}{\tau})} = \frac{p_v^{\frac{1}{\tau}}}{\sum_u p_u^{\frac{1}{\tau}}}. \tag{8.10}$$

此处温度用于生成分布变换，不意味着经过验证集校准后获得了事实置信度．正温度不改变分数排序，但改变候选间相对概率：

$$\frac{p_i^{(\tau)}}{p_j^{(\tau)}} = \exp\left(\frac{z_i - z_j}{\tau}\right). \tag{8.11}$$

当 $\tau \to 0^+$ 且最大分数唯一时，分布趋于该候选上的点质量；若多个有限最大值相同，则趋于这些最大值上的均匀分布．$\tau = 0$ 不能直接代入除法，若要贪心选择，应采用明确的离散分支．$\tau \to \infty$ 时，在有限分数支持集上趋于均匀．已被硬屏蔽的候选不因此恢复．

8.6.2 温度及熵的关系

温度单调性的适用条件

本节固定同一个前缀、同一组有限分数和非空候选支持集，只改变正温度．求导期间不重新训练、不修改截断集合，也不切换到另一个采样前缀．概率与熵均指这一固定条件下的分布．

对固定有限分数集合，令 $\beta = \dfrac{1}{\tau}$、$Z(\beta) = \displaystyle\sum_v e^{\beta z_v}$．熵写为

$$H(\beta) = \log Z(\beta) - \beta \mathbb{E}_{p^{(\tau)}}[z]. \tag{8.12}$$

由 $\dfrac{\mathrm{d}\log Z}{\mathrm{d}\beta} = \mathbb{E}[z]$ 以及 $\dfrac{\mathrm{d}\mathbb{E}[z]}{\mathrm{d}\beta} = \mathrm{Var}(z)$，有

$$\frac{\mathrm{d}H}{\mathrm{d}\beta} = -\beta\,\mathrm{Var}(z), \qquad \frac{\mathrm{d}H}{\mathrm{d}\tau} = \frac{\mathrm{Var}_{p^{(\tau)}}(z)}{\tau^3} \geqslant 0. \tag{8.13}$$

因此在固定支持集与固定分数下，提高温度不会降低条件熵．这是关于同一时刻分布的结果；不同采样路径产生不同前缀后，不能以此保证最终文本的质量或多样性按同样方式单调改变．

8.6.3　Top-k 及核采样

前 k 项采样（Top-k Sampling）保留概率最大的 k 个候选，令集合为 $\mathcal{K}$，再归一化：

$$q_v = \frac{p_v \mathbf{1}[v \in \mathcal{K}]}{\sum_{u \in \mathcal{K}} p_u}. \tag{8.14}$$

边界并列时可以按稳定规则恰好选择 k 项，也可以保留所有达到阈值的项；后一种方式的支持集可能多于 k. 两种定义均应明确，不能用同一个配置名掩盖实际集合差异.

核采样（Nucleus Sampling，Top-p）按概率从大到小排序，取累计概率至少达到 ρ 的最小前缀集合 [33]. 若排序概率为 $p_{(1)} \geqslant \cdots \geqslant p_{(V_{\text{vocab}})}$，则

$$m = \min\left\{ r : \sum_{j=1}^{r} p_{(j)} \geqslant \rho \right\}, \qquad \mathcal{N}_\rho = \{(1), \ldots, (m)\}. \tag{8.15}$$

之后在 $\mathcal{N}_\rho$ 内重新归一化并采样. 跨过阈值的那个候选必须保留，尤其在第一候选就超过阈值时，集合仍至少包含一个词元. Top-p 中的 p 是方法名称，正文用 ρ 表示阈值，以区别模型概率.

Top-k 固定候选数量，Top-p 固定保留的累计概率门槛. 尖锐分布下 Top-p 可能只留下一个词元，平坦分布下则可能留下很多. 二者均改变支持集，可能移除 EOS 或任务所需的低概率候选，不应视为无代价的质量保证.

8.6.4　采样变换顺序

设某个前缀后的原始概率为

$$p = (0.4, 0.3, 0.2, 0.1), \tag{8.16}$$

对应分数可取 $z = (\log 0.4, \log 0.3, \log 0.2, \log 0.1)$. 温度 $\tau = \dfrac{1}{2}$ 相当于将概率平方再归一化，平方和为 0.30，所以

$$p^{(\frac{1}{2})} = \left(\frac{8}{15}, \frac{3}{10}, \frac{2}{15}, \frac{1}{30} \right). \tag{8.17}$$

原分布做 Top-2 得到 $(\frac{4}{7}, \frac{3}{7}, 0, 0)$；原分布做阈值 $\rho = 0.75$ 的 Top-p 时，前两项之和为 0.7，必须保留第三项，得到 $(\frac{4}{9}, \frac{1}{3}, \frac{2}{9}, 0)$.

若先降温再做同样的 Top-p，前两项的质量为 $\dfrac{8}{15} + \dfrac{3}{10} = \dfrac{5}{6} > 0.75$，只保留两项，最终分布为 $(\frac{16}{25}, \frac{9}{25}, 0, 0)$. 如果先在原分布上做 Top-p，再降温，支持集仍有三项，最终为 $(\frac{16}{29}, \frac{9}{29}, \frac{4}{29}, 0)$. 因此温度与 Top-p 的次序会改变分布. 温度不改变候选排序，所以在没有其他变换、并列规则相同的条件下，它与固定 Top-k 支持集选择的关系不同；不能据一个特例推断所有处理器都可交换.

8.6.5 生成分布偏移

设经过温度、截断和其他规则后的条件分布为 $q_t(\cdot \mid c, y_{<t})$，实际随机生成的路径概率为

$$q(y_{1:T} \mid c) = \prod_{t=1}^{T} q_t(y_t \mid c, y_{<t}). \tag{8.18}$$

它通常不等于原始 $p_\theta(y_{1:T} \mid c)$. 即使只用相同温度，每个前缀的归一化常数不同，也不能一般地将结果解释为“整条序列概率统一取 $\frac{1}{\tau}$ 次方后归一化”. 逐步变换定义的是另一组条件分布.

记录生成概率、计算重要性比率或检验精确采样时，必须说明使用原始模型分布还是实际解码分布. 温度和截断不会改变模型权重，也不会为模型补充事实知识；它们改变的是已给定分数被转化为决策的方式.

8.7 贪心及束搜索

8.7.1 局部最优及序列最优

贪心解码（Greedy Decoding）每一步选择条件概率最大的候选，按固定规则解决并列. 它最大化当前一步，而非全部未来. 对完整序列，应比较式 (8.3) 中的乘积，计算时通常累加对数.

构造一个有限分支概率模型：第一步 $p(a) = 0.6, p(b) = 0.4$；在 a 后，EOS 与 c 的概率各为 0.5；在 b 后，EOS 概率为 0.9、c 为 0.1；一旦产生 c，下一步必为 EOS. 其完整序列分布为

完整序列	概率
a, EOS	$0.6 \cdot 0.5 = 0.30$
a, c, EOS	$0.6 \cdot 0.5 \cdot 1 = 0.30$
b, EOS	$0.4 \cdot 0.9 = 0.36$
b, c, EOS	$0.4 \cdot 0.1 \cdot 1 = 0.04$

四条路径概率之和为一. 贪心第一步选 a，无论随后并列选择 EOS 还是 c，完整概率均为 0.30，低于 b, EOS 的 0.36. 这个模型用零概率屏蔽未列分支，仅用于精确说明序列决策，不代表某个训练模型的测量.

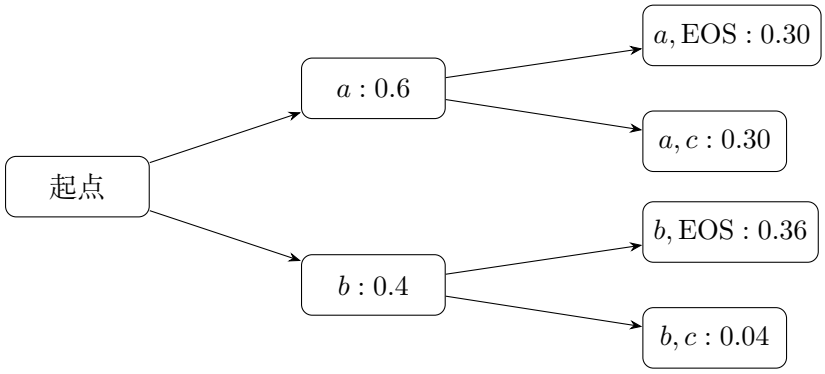


图 8.3: 局部贪心错过完整高概率序列的概率树. 节点数值为累计概率；含 c 的前缀下一步以概率一结束.

8.7.2　束搜索保留多个前缀

束搜索（Beam Search）以有限宽度 K 保留候选，反复扩展与剪枝. 设前缀分数为

$$s(y_{1:t}) = \sum_{j=1}^{t} \log p_\theta(y_j \mid c, y_{<j}). \tag{8.19}$$

每次扩展一个候选 v，只需令 $s' = s + \log p_\theta(v \mid c, y_{1:t})$. 同一深度的候选可以批量前向，但各自前缀、缓存和结束状态必须保持对应，不能在排序后错配状态.

图 8.3 中宽度为二时，第一步保留 a, b；第二步最高的扩展是 b, EOS，得分对应 0.36，其次为两条 0.30 分支之一. 因此保留多个前缀可修复该例的贪心选择. 但有限宽度仍可能过早删除一条未来更优的路径；增大束宽不提供一般的全局最优保证，更不保证任务质量.

算法 8.2　以原始对数概率评分的束搜索

输入： 条件 c、束宽 K、EOS 集合、最大新增长度 $L_{\max}$ 和确定的并列规则. **输出：** 一个完成候选及终止原因，或未完成候选及长度截断标记. **状态：** 至多 K 条记录，每条含前缀、累计分数、完成标记及其模型状态.

1. 初始化空响应候选，累计分数为零，状态与条件 c 对应.
2. 对每条未完成候选计算下一步分布，为每个有限分数词元生成扩展并累计式 (8.19)；若词元属于 EOS 集合，将该扩展标记完成. 已完成候选原样进入候选池，不再追加词元或重复计分.
3. 从候选池按原始累计分数保留最高的 K 条，连同其对应模型状态一起重排. 若不足 K 条则保留全部.
4. 若当前最佳完成候选的分数不低于所有未完成前缀分数，可以停止；若全部候选完成，也停止. 否则继续扩展，最多进行 $L_{\max}$ 次.

5. 有完成候选时返回其中最高分者；没有时返回最高分未完成前缀并报告截断. 本规则不把外部预算耗尽伪装成模型选择 EOS.

本算法让完成候选占据束内位置，定义了一种明确的有限束策略；$K = 1$ 时与采用相同并列及终止规则的贪心解码一致. 维护独立完成池的实现可以使用不同的候选预算，必须另行说明.

8.7.3 束搜索停止条件

每个条件概率不超过一，所以 $\log p \leqslant 0$，任何未完成前缀的未来扩展分数都不超过当前分数. 如果一个完成候选已经不低于所有保留前缀分数，那么继续扩展这些前缀不可能超过它. 这证明了算法 8.2 中的停止条件，但仅相对于尚未剪去的候选成立，无法重新发现此前被删除的路径.

长度归一化会改变这个性质. 常见目标可写为 $\dfrac{s(y)}{|y|^{\alpha}}$，$\alpha > 0$，其中长度是否包含 EOS 必须固定. 因为 $s(y)$ 为负，分母随长度增大可能使归一化分数上升，当前前缀分数不再直接是未来得分上界. 若存在确定的最大长度，可以根据该目标另行构造上界；若还加入奖励或覆盖项，则必须按完整评分重新推导，不能沿用原始对数概率的停止证明.

8.7.4 长度偏好及任务目标

较长路径累加更多非正项，原始序列概率常偏向某些短输出，但这不是“序列越长概率一定更低”的跨路径定理，因为不同路径的条件概率不同. 长度归一化、最小长度与重复惩罚可以改变搜索行为，同时也改变目标，应该明确其服务的任务约束.

最大概率输出与高质量输出不是同一个数学目标. 开放式任务有大量合理续写，高概率路径可能重复或过于笼统；有严格约束的翻译与结构化任务也可能因长度或覆盖问题失真. 解码选择应围绕任务评价，而不能把更大的束宽当作模型能力提升. 核采样相关研究讨论的退化现象也有特定模型和任务条件，不能外推成所有任务上唯一最优策略 [33].

8.8 终止、约束及重复退化

8.8.1 生成停止条件

模型采样出 EOS 表示模型分布选择结束；命中应用规定的停止串表示外部协议结束；耗尽 $L_{\max}$ 表示预算截断. 三者的语义不同，应保留在返回状态中. EOS 通常不作为自然语言正文展示，但它仍属于模型的输出事件.

停止串可能跨词元边界，也可能跨流式分片边界；应对连续解码状态判断匹配，不能逐分片做独立字符串检查．是否保留停止串、匹配多个停止串时如何处理、特殊词元是否参与文本解码，都属于输出协议．[1]

最小生成长度通常通过暂时禁止 EOS 来实现，它在允许长度之前改变分布．若用户请求短答案却强制较长最小长度，模型可能被迫继续补写．最大新增词元数与“提示加响应的最大总长度”也不同，前者控制输出预算，后者受到上下文容量约束；记录配置时必须区别．

8.8.2　带明确状态的逐步生成

算法 8.3　温度与核采样的有限生成过程

输入：已经序列化的条件词元、词表与模板、温度 $\tau > 0$、核阈值 ρ、EOS 和停止串、最大新增长度 $L_{\max}$．**输出：**响应词元、文本与停止原因．**状态：**当前前缀、随机数状态、可选键值缓存以及连续文本解码状态．

1. 固定推理模式；检查条件长度与位置范围．将当前前缀设为条件，响应置空，不把已结束请求作为活跃候选继续处理．
2. 从当前合法前缀计算末位置分数．施加已声明的硬约束；若没有任何允许候选，返回约束冲突，不对全屏蔽分数执行 Softmax．
3. 用 $\frac{z}{\tau}$ 得到稳定归一化分布，按确定的并列规则构造累计概率至少为 ρ 的最小核；在核内重新归一化．
4. 根据本次随机数状态抽取一个词元，将它追加到响应和下一步前缀，保持缓存、位置与新前缀一致．
5. 若采到 EOS，报告自然终止；否则更新连续文本状态，命中停止串时按协议结束．尚未终止且已新增 $L_{\max}$ 个词元时，报告长度截断．
6. 未结束时返回第 2 步；最终返回结果及停止原因．返回的每个词元都只依赖其之前已经确定的前缀．

贪心模式应使用独立的最大值选择分支，不将 $\tau = 0$ 送入除法．Top-k、重复惩罚或其他处理器若参与执行，还必须说明它们在第 2 至 3 步中的顺序．

[1]某些对话格式区分“一个消息结束”和“本轮助手回答结束”．这些标记即使都表现为特殊词元，也不能未经模型模板说明就合并为同一 EOS 语义．

8.8.3 重复的跨步反馈

一段重复输出进入前缀后，模型可能继续给同类续写较大概率；新产生的重复又成为条件，形成跨步反馈．低温度会在重复词元已是最高分时进一步集中概率，却不是重复的充分条件．贪心和束搜索同样可能保留重复路径，随机采样也可能重复．

重复 n 元组屏蔽可以禁止已经出现的片段再次出现，但会误伤需要复述的名称、代码、公式或合法韵律．按已出现词元次数调整分数是一种软约束，也会改变语义词的使用概率．若所有合法候选都被规则排除，算法必须报告冲突或使用事先声明的回退，而不是临时偷偷取消约束．

还应区分模型重复与系统重复．流式分片重复拼接、错误重试或缓存前缀错配也能产生重复文本，它们并非调高温度就能解决．模型分数、解码规则和传输状态对应不同的原因，应分别定位．

8.9 条件生成及缓存的边界

8.9.1 提示、任务边界及模板

仅解码器条件生成通常把系统说明、用户输入与历史消息序列化为一个前缀，再续写响应．消息角色与边界标记属于模型看到的条件，不是界面装饰．把相同自然语言放入不同角色、删除分隔符或改变响应起始标记，可能产生不同的条件分布．

训练时若只监督响应，应保留提示对响应的可见性，同时将提示的损失权重设为零．测试条件生成时通常比较 $p(y \mid c)$，无需把提示本身的概率 $p(c)$ 加入响应排名；不同提示长度的总序列损失也不能直接作为响应质量比较．

在编码器–解码器模型中，条件 c 先编码为记忆，目标侧仍通过教师强制或逐步生成处理．交叉注意力通常允许读取全部有效源记忆，目标自注意力保持因果．两种结构的分词器、特殊标记与权重布局不可只凭输入输出形状互换．

8.9.2 缓存复用的分布等价性

如果权重、位置和可见性不变，新增未来词元不会改变历史位置的隐藏状态．因此历史各层键和值可以保存，下一步只计算新增位置并读取缓存．缓存复用应产生与重新计算相同前缀相同的数学条件分布；它不是把原始长前缀压缩为一个固定维数向量．

如果前缀修改、位置方案变化、模型参数更换或掩码不再允许原来的信息范围，缓存便可能失效．滑动窗口也必须明确舍弃哪些历史状态以及如何解释位置．性能优化不能以错误复用改变概率模型．

8.9.3　生成复现条件及概率语义

同一权重和提示不足以确定随机采样结果，还需要解码配置、随机数状态、输入模板和执行环境．固定随机种子有利于同一条件下复核，却不保证跨设备、后端或数值精度逐词元一致．很小的概率差异也可能使一次离散选择落到不同分支，继而改变整个后续前缀．

词元概率表示模型在给定上下文中的条件偏好，不是某个事实命题的可靠置信度．对多个词元组成的回答，也不能把其中一个高概率词元当作整段真实性证据．最终评价应依据任务要求，并保持模型原始分布、解码后分布和实际返回文本的区别．

生成质量来自三个相互独立的定义

模型定义 p_θ，解码器定义如何选择或抽样，停止协议定义输出在哪里结束．训练损失验证条件预测，采样规则决定实际输出分布，序列任务评价判断结果是否有用．三者不能互相替代，也不能用调整一个解码参数补足所有模型与系统问题．

习题参考结果　第 2 题完整概率为 0.28，负对数似然为 $-\log 0.28$，平均损失为 $-\dfrac{\log 0.28}{3}$．第 4 题的核集合分别包含前两项与全部四项；第一种重新归一化为 $(\dfrac{4}{7},\dfrac{3}{7},0,0)$．第 7 题若把产生 EOS 的一步也计入，则步数服从参数为 ϵ 的几何分布，期望为 $\dfrac{1}{\epsilon}$；非 EOS 词元数的期望为 $\dfrac{1-\epsilon}{\epsilon}$．这些结论依赖固定条件概率假设，不是实际模型生成长度的通用分布．

第九章　模型评估

评估要回答的是一个有条件的决策问题：在什么任务与用户分布上，某个系统是否比基线更有效，代价和风险是否可以接受，以及现有证据还有多大不确定性．有限测试集上的分数只是这个问题的一部分．任务定义错误、标签失真或测试污染都可能产生很精确却不对应目标的数字．

本章将指标、测量误差和统计推断放在同一条证据链上．检索与智能体的组件指标分别见第十二章《信息检索》、第十三章《检索增强生成》及第十六章《智能体运行系统》；这里解释如何把逐例观测聚合为有边界的比较，如何设计人工与模型评审，以及怎样避免把离线相关性当作线上因果效果．

9.1　评估对象

9.1.1　总体、协议及目标量

待估目标量（Estimand）是统计分析真正要估计的对象．设任务输入及其参考信息为 $X \sim \mathcal{P}$，固定系统为 f_v，随机生成条件为 ω，评估协议为 e，逐例效用为 u．目标可以写为

$$\mu_v = \mathbb{E}_{X\sim\mathcal{P},\omega}\left[u(X, f_v(X;\omega); e)\right]. \tag{9.1}$$

这里的系统版本 v 不只是模型名称，还包括提示、分词器、解码、检索、工具和输出解析等配置．改变协议 e 也可能改变目标量，例如把不适用项计零与从分母排除，会得到不同含义的分数．

基准测试（Benchmark）是在约定任务和协议下形成的可比较坐标．部署中的任务效用还包含错误损失、完成时间、重试、人工介入与资源成本；基准得分不能不经论证就替代式 (9.1)．即使某个题库中所有答案都正确，也不能据此推断另一语言、工具环境或时间范围的任务都正确．

9.1.2　能力及证据分层

知识评估需要限定知识时间、领域与可访问来源；语言评估关注理解、表达和语言覆盖；推理评估需要检验中间约束与最终结论；长上下文评估需要控制证据位置、干扰与组合跨度；工具使用评估还需要判定参数、授权、执行结果和副作用．通用能

表 9.1: 主要统计对象.

符号	含义
n, G, m_g	记录数、独立组数、第 g 组记录数
$s_i^{(A)}, s_i^{(B)}, d_i$	同一案例两系统分数与配对差
$p, \hat{p}$	总体通过概率与样本通过比例
$\alpha, z_{1-\frac{\alpha}{2}}$	显著性水平与标准正态分位数
Δ, δ	系统期望差与预先规定的最小效应或非劣界
$\mathcal{P}, \mathcal{Q}$	目标任务分布与实际抽样分布

力、领域能力与安全对齐必须按任务目的分别定义，不能用大量普通问答平均掉少数高损失失败.

组件指标解释局部机制，端到端任务指标观察整体完成情况，鲁棒性和安全切片观察特定失败面，在线随机实验估计真实流量中的干预效果. 四类证据不能互相替代. 一个检索器召回相关文档，并不证明生成回答忠实；一个回答通过人工评分，也不证明工具调用遵循了权限.

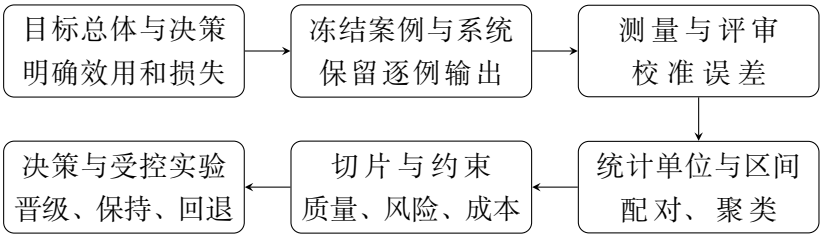


图 9.1: 从任务定义到决策证据. 每一层都保留自己的适用条件.

9.2　常用数据集及基准协议

9.2.1　环境构建

评测材料至少有三种形态. 静态数据集提供问题、参考答案或标签；交互基准还提供工具、初始状态、时间限制和执行后的验收器；综合指数则汇总多个基准的结果. 它们都可以用于比较系统，但一个指数不是一套题库，一个终端环境也不只是若干文本问题. 以下所称“常用”包括研究中的基础基准和模型发布中具有代表性的基准，并不表示所有基准都适合任意业务.

对静态选择题，核心观测通常是答案是否正确. 对交互任务，**评测运行框架**（Evaluation Harness）负责组织提示、工具调用、环境恢复、预算与结果收集. 同一个模型换用不同运行框架，可能执行不同的搜索与修复策略. 因此，一个可解释的分数必须绑定

$$\mathcal{E} = (D_v, M_v, H_v, B, R, A), \tag{9.2}$$

其中 D_v 为数据或环境版本，M_v 为模型版本，H_v 为运行框架，B 为词元、时间、工具调用和尝试次数预算，R 为逐例计分规则，A 为聚合规则. 安全控制及回退模型也属于系统配置. 两个结果只要这些要素存在实质差异，就不能仅凭同一个基准名称推断为受控对照.

9.2.2 知识、科学及数学

研究生级难检索问答（Graduate-Level Google-Proof Question Answering，GPQA）由生物、物理和化学领域专家编写多项选择题，考察专门知识与推理；Diamond 是其中的特定子集，不能与完整版本混写. 主要指标是答案准确率. 题目难以通过普通检索解决，是基准设计目标，不构成永远不存在网络答案的保证 [34].

人类最后的考试（Humanity's Last Exam，HLE）覆盖多学科的高难度封闭式问题，包含多模态材料，公开定稿集为 2500 题，并有单独保留的私有测试材料. 报告中需要区分文本子集、完整输入、工具可用性及数据修订版本. 其准确率与置信度校准考察不同性质：答对更多问题不意味着模型能识别剩余错误，更不直接证明具有开放式科研能力 [35].

FrontierMath 面向高难度数学问题. Tier 4 是特定难度层，不等于全部题目；v2 修订过题目与集合，因此版本必须随分数保留. 官方协议允许使用计算工具，并提交可验证答案. 研究级问题上的答案验证与完整证明审查也不是同一件事：一个数值或符号对象通过验收，不能自动替代对证明每一步的核查 [36].

基础知识及中文评估

大规模多任务语言理解（Massive Multitask Language Understanding，MMLU）使用跨学科选择题，为知识与推理提供基础坐标；MMLU-Pro 强化了推理要求并调整题目与选项，属于另一套基准. 比较时须同时固定少样本示例、选项顺序、答案解析和学科权重 [37, 38]. 如果模型已经接近某个题集的上限，少量分数变化提供的区分信息会减少；仍可将其用于回归检测，但选型还需要更贴近目标任务的材料.

C-Eval 与**中文大规模多任务语言理解**（Chinese Massive Multitask Language Understanding，CMMLU）分别覆盖 52 个学科与 67 个主题，适合补充中文及本地知识的评估. 二者不是同一套英文题的不同译名；C-Eval Hard、验证集与测试集也必须分别标识. 中文选择题得分不能替代中文长文写作、引用问答或业务流程评估 [39, 40].

基础数学及竞赛题

GSM8K 以多步基础算术应用题为主；MATH 包含竞赛数学题及参考解答. 通常依据解析后的最终答案计算正确率，但数值格式、符号等价判定和工具使用都属于协议，不能仅报告模型名称 [41, 42]. MATH-500 与完整测试集的样本构成不同，报告中应保留子集名称.

美国数学邀请赛（American Invitational Mathematics Examination，AIME）是逐年举行的竞赛，每套 15 题，答案为 0 至 999 的整数；它不是一个自带统一模型采样协议的静态数据集．模型评测应写明年份、I 卷或 II 卷、是否合并、每题尝试次数及答案选择方法 [43]．例如一套 15 题中多答对一题就改变约 6.67 个百分点，故小题集的年度差异与抽样变异尤其不能忽略．

函数生成及时间分组的代码题

HumanEval 根据函数签名和文档字符串要求生成 Python 实现，通过测试估计 pass@k；其组合估计见式 (9.9)．LiveCodeBench 持续汇集竞赛编程题，并区分代码生成、自修复等场景；用于比较时应固定发布版本、题目日期窗口和测试裁剪方式 [44, 45]．按日期划分有助于控制时间泄漏，但并不自动证明待测模型从未接触题目．基于参考答案判断失败后再反复采样，也不能伪装成一次生成的准确率．

多模态、长上下文及指令遵循

大规模多学科多模态理解（Massive Multi-discipline Multimodal Understanding，MMMU）将大学学科问题与图表等视觉材料结合，按答案正确性计分．需要固定图像分辨率、多图处理及测试划分，避免把视觉预处理差异误认为纯文本推理差异 [46]．MMMU 与 MMMU-Pro 分属不同协议．公开题目与标签的状态也会改变：不能因早期论文采用隐藏测试标签，就永久假定当前测试答案仍未公开．

LongBench 原版混合长文问答、摘要、检索与代码等任务，分别采用 F1、ROUGE 或相似度等指标；LongBench v2 改用更强调长上下文推理的多项选择题．原版总分不是统一的答案准确率，v2 也不能被当作原题集仅增加长度 [47, 48]．对长度的计量还应注明字符、词还是具体分词器的词元．

多轮共指消解（Multi-Round Coreference Resolution，MRCR）要求在含相近请求的长对话中找回指定的既有回答．OpenAI 公开版本按针数及长度范围分组，计分器先检查指定前缀，再计算文本相似度；它不是一般问答的二元准确率 [49]．这类合成检索成绩需要与多文档推理互补，不能据此声称整个长上下文都被可靠理解．

指令遵循评估（Instruction-Following Evaluation，IFEval）利用程序检查词数、关键词与输出形式等可验证约束．它同时区分提示级和指令级、严格和宽松两组口径：提示级要求一条提示内所有约束均通过，指令级按单个约束汇总；宽松计分只容许协议列出的有限输出变换，并不是任意人工放宽要求 [50]．格式约束通过不证明事实正确或业务价值充分．

9.2.3　代码、浏览及计算机操作

SWE-bench 根据真实软件仓库的问题描述要求系统生成修补，并在仓库环境中运行测试；Verified 是经过人工筛选的 500 个实例子集．主要指标为解决比例．该结

果受基础提交、依赖环境、测试验收及智能体运行框架影响；函数级代码生成成绩不能替代仓库级修复证据，仓库测试通过也不能证明所有潜在行为均正确 [51].

Terminal-Bench 系列通过终端环境中的可执行任务评估智能体. 需要把通用终端任务与科学工作流分支分开：例如发布材料中的 Terminal-Bench 4.0 与 Terminal-Bench Science 0.1 是不同版本和任务分布，不能合并成一项“代码准确率”. 科学工作流还涉及数据分析、模拟与模型拟合；最终产物是否满足验收器，比回答是否声称完成更有意义 [52].

BrowseComp 以难以定位、但答案较短且可核验的信息检索问题为对象，原始集合包含 1266 题. 它适合研究多步搜索与证据定位，主要报告答案准确率；浏览器、搜索源、访问时间和预算都会影响结果. 长篇报告的论证质量、引用覆盖和用户需求理解仍需另外评估 [53].

OSWorld 2.0 是跨软件的长程计算机操作基准，可按完整集或离线子集评价，也区分部分奖励与二元完成判定. ScreenSpot-Pro 则侧重屏幕元素定位；定位正确并不等于完成整个工作流. 前者需要检查最终状态和多步约束，后者的测量对象更接近动作选择中的视觉定位组件 [54, 52].

9.2.4 交互推理及计分单位

ARC-AGI-3 要求智能体探索陌生的交互环境并从反馈中学习，计分包含相对于人类的行动效率. 它与 ARC-AGI-1、ARC-AGI-2 的静态任务形式不同；三个版本的百分数不是同一种准确率，也不能因一个基准接近满分就推导出对所有环境的普遍能力 [55].

即使分数都被写为百分数，也必须检查分子和分母. 设任务 i 有 m_i 个验收项，权重 $w_{ij} \geqslant 0$ 且 $\sum_j w_{ij} = 1$，验收结果为 $r_{ij} \in \{0, 1\}$. 可以构造任务部分得分与全通过指标

$$r_i^{\text{partial}} = \sum_{j=1}^{m_i} w_{ij} r_{ij}, \qquad r_i^{\text{complete}} = \prod_{j=1}^{m_i} r_{ij}. \tag{9.3}$$

这里假定每个验收项均为完成任务所必需，且没有额外一票否决规则. 两个任务各有五项等权要求，分别满足四项与两项，则平均部分得分为 60%，完整任务成功率却为 0. 这个构造例子用于解释计分差异，不是复刻某个实际基准的验收器. 正式实验应使用该版本规定的规则.

9.2.5 模型发布评测

以下记录 2026 年 9 月 8 日核实的 GPT-6 Astra 与 Claude Fable 5.1 发布材料，用于说明如何阅读模型发布表. 它们是厂商报告的历史观测，不是本书重新运行的实

验，也不是持续更新的模型排名．表 9.2 选取共同或具有代表性的项目；空项表示本表没有对应发布数值，不能填成零．

表 9.2: 发布时的代表性评测结果

基准或计分项	Astra	Fable 5.1	指标及限定
Terminal-Bench 4.0	57.9%	55.8%	终端任务得分；运行框架须另核对
Terminal-Bench Science 0.1	64.6%	52.6%	科学工作流得分
HLE，有工具	57.2%	65.0%	答案准确率；工具与裁判协议不同
HLE，无工具	—	60.9%	与有工具配置分开
AutomationBench	41.4%	31.4%	业务流程任务通过比例
GDPval-AA v2	—	1853	Elo，非百分数
CursorBench 3.2.0	—	73.4%	Cursor 编码任务评分
OSWorld 2.0，Astra 发布协议	72.6%	—	v2026.08.08 离线集，部分得分
OSWorld 2.0，Fable 发布协议	—	77.9%	8 月任务版及修复，部分得分
同上，严格完成口径	—	41.7%	全部要求满足的任务比例
GPQA Diamond	96.0%	—	科学选择题准确率
FrontierMath Tier 4 v2	97.6%	—	指定数学题集得分
ARC-AGI-3	99.9%	—	交互推理与行动效率
BrowseComp	91.5%	—	浏览检索答案准确率

数据来源分别为 OpenAI 的 Astra 发布表与 Anthropic 的 Fable 5.1 发布主表 [52, 56]．Fable 主表共七个基准、九个计分行，已全部列入；其系统卡中的其他基准不在这里冒充发布主表．Astra 仅选取代表项目，未将全部内部评测展开．Astra 引言把 FrontierMath 成绩取整为 98%，这里保留正式表中的 97.6%．

GPT-6 Astra 的发布表还包含专业工作、科学健康、安全和长上下文等评测：例如 Agents' Last Exam、BenchCAD、DeepSWE v1.1、FrontierCode 1.1、HealthBench Professional、GeneBench Pro、LifeSciBench、MRCR v2，以及内部设计、数据科学和数据库迁移任务．Artificial Analysis 的 Intelligence Index 与 Coding Agent Index 是综合指数，不是单个数据集．内部任务可以提供补充证据，但缺少公开题目和完整协议时，本书不能声称独立复现 [52]．

读发布表时，尤其要保留下列限定．第一，Astra 表中的 OSWorld 2.0 明确为 v2026.08.08 离线集合与部分得分，不能与不同任务修改及评分设置的结果直接相减．第二，Astra 发布表报告各推理档位中的最高分；它不是固定成本下的统一对照．推

理强度的模型相关语义与对照方法见第二十六章《可验证推理训练》. 第三，安全干预后改由另一个模型完成的任务，应被解释为包含回退路径的系统结果，不能归给一个未经切换的模型. [1][52]

Fable 5.1 发布案例中的 GDPval-AA v2 根据职业任务产物的盲配对比较形成 Elo；AutomationBench 检查跨应用业务流程的最终状态；CursorBench 3.2.0 使用 Cursor 的编码任务与运行框架. Elo 是相对评级，1853 不能读成 1853% 的准确率，跨日期的评级还可能因参评系统与拟合结果变化而改变. 它们分别提供产物偏好、流程验收和编码工作流证据，不应直接平均为一个百分数 [57].

Fable 的系统卡还说明了影响解释的实验条件：HLE 分别运行有工具与无工具配置，使用固定裁判，并对检索到题库答案的轨迹重新判错；OSWorld 被安全机制中止的指定任务计零，其他受干预的特定任务可能由较早模型接替. Terminal-Bench Science 0.1 各模型报告的标准误约为 3.5 至 4.5 个百分点，主要不确定性来自任务差异. 这是标准误范围，不是每个模型统一的 95% 置信区间 [57]. 同一道任务的多次运行也不能都当成新的独立任务，本章后文的聚类与重复测量方法仍然适用.

这些限制不是把结果全部否定，而是决定证据能支持什么判断. 若目标是选择固定月度预算下的代码助手，就应在相同任务和成本约束下比较质量；若目标是研究最大可达能力，可以提高预算，但需同时报告尝试次数、运行时间与成本. 跨厂商榜单通常不能提供同题配对结果，因此不应直接套用本章配对检验给分数差赋予显著性.

算法 9.1 选择并冻结评测组合

输入： 目标任务分布、错误损失、候选系统、时间与费用预算.
输出： 版本化基准组合、逐例验收规则与统计计划.

1. 将目标拆成知识、推理、检索、工具动作和最终产物等可观察能力，确定哪些约束必须同时满足.
2. 选取相关公开基准，并加入来自真实业务、与训练和开发过程隔离的保留案例；记录每组材料的来源与权重.
3. 冻结式 (9.2) 中的全部要素，特别是工具、预算、失败计分、安全控制与回退路径.
4. 在开发材料上校验验收器和答案解析；冻结后保留所有测试案例，包括超时、拒答、无效产物与环境故障的分类记录.
5. 保存逐例观测，按预先定义的独立单位计算区间及切片结果；同时报告质量、延迟、费用和高损失失败.

[1] Astra 发布材料还注明，部分标在 Fable 名下的 ScreenSpot-Pro、ExploitGym 数值来自安全限制不同的 Mythos 配置. 模型名称、访问配置与实际执行者必须共同记录.

停止与边界：达到预定样本及预算后停止．公开分数无法提供的配置或逐例信息应明确缺失，不通过猜测补齐．评测组合变化后重新定义目标量．

9.3　分类、概率及决策代价

9.3.1　混淆矩阵及平均口径

混淆矩阵（Confusion Matrix）以真实标签为行、预测标签为列．二分类中的真阳性、假阳性、真阴性和假阴性分别记为 TP、FP、TN、FN．**准确率**（Accuracy）、**精确率**（Precision）和**召回率**（Recall）为

$$\mathrm{Acc}=\frac{\mathrm{TP}+\mathrm{TN}}{n},\quad P=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}},\quad R=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}. \tag{9.4}$$

F1 分数（F1 Score）是精确率和召回率的调和平均：

$$F_1=\frac{2PR}{P+R}=\frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}. \tag{9.5}$$

分母为零时须显式规定未定义、排除或计零，并报告受影响类别．把“本次没有这种任务”与“这种任务全失败”都记成零会改变结论．

构造一个 $n=100$ 的例子：实际正类 20 条，其中 12 条预测为正；实际负类 80 条，其中 4 条预测为正．因此 $\mathrm{TP}=12,\mathrm{FN}=8,\mathrm{FP}=4,\mathrm{TN}=76$．准确率为 0.88，正类精确率 $\frac{12}{16}=0.75$，召回率 $\frac{12}{20}=0.60$，F1 为 $\frac{24}{36}=\frac{2}{3}$．若把负类也按一对其余类别计算，负类 F1 为 $\frac{152}{164}\approx 0.9268$．

宏平均（Macro Average）使各类别等权，本例 Macro-F1 约为 $\frac{0.6667+0.9268}{2}=0.7967$．**微平均**（Micro Average）先汇总各类计数，再计算指标．在单标签、类别全集齐全的多分类中，每次错误同时贡献一个 FP 和一个 FN，故 Micro-F1 等于准确率，本例为 0.88．这一等式不应推广到多标签或只选择部分类别的评估．

9.3.2　阈值及概率校准

若 $p=\Pr(Y=1\mid x)$ 是已校准条件概率，误报和漏报损失为 $c_{\mathrm{FP}},c_{\mathrm{FN}}>0$，预测正类的条件风险是 $c_{\mathrm{FP}}(1-p)$，预测负类的风险是 $c_{\mathrm{FN}}p$．选择前者当且仅当

$$c_{\mathrm{FP}}(1-p)<c_{\mathrm{FN}}p\quad\Longleftrightarrow\quad p>\frac{c_{\mathrm{FP}}}{c_{\mathrm{FP}}+c_{\mathrm{FN}}}. \tag{9.6}$$

例如误报损失为 9、漏报为 1，阈值为 0.9．这个推导假设正确决策损失为零且损失不随个体变化；有额外行动、拒答或个体成本时，应比较全部行动的条件风险．

概率校准（Probability Calibration）要求预测为某概率水平的样本具有相应事件频率．排序好不等于校准好．一个模型把所有概率从 p 单调变换为 p^2，排序可以保

持，式 (9.6) 的概率含义却不再自动成立．二分类 Brier 分数为 $n^{-1}\sum_i (p_i - y_i)^2$；若真实概率为 q，其条件期望等于 $(p-q)^2 + q(1-q)$，因此在 $p = q$ 处最小．

期望校准误差（Expected Calibration Error，ECE）常将预测分桶，计算各桶预测均值与事件比例差的加权绝对值．它依赖分桶规则，粗分桶可能掩盖局部错误，样本稀少时又有较大估计噪声．阈值和校准器都应在开发或校准数据上确定，最终测试数据用于评估冻结方案．

9.3.3　排序曲线

提高阈值使预测正类集合缩小，因此召回率非增；精确率不具有一般的单调保证，因为删除的下一个样本可能是真阳性，也可能是假阳性．**受试者工作特征曲线**（Receiver Operating Characteristic，ROC）描绘真阳性率与假阳性率；其面积可解释为随机正例分数高于随机负例的概率，平分时通常计半分．

精确率—召回率曲线（Precision–Recall Curve，PR）尤其需要同时报告正类基率．按排序位置 j 记精确率 P_j 和召回增量 $R_j - R_{j-1}$，一种**平均精确率**（Average Precision，AP）定义为 $\sum_j P_j (R_j - R_{j-1})$，不等于任意插值方式的曲线面积．曲线概括阈值族，部署阈值仍应由实际损失与约束决定．

9.4　生成指标的测量对象

9.4.1　概率及词面一致性

困惑度（Perplexity，PPL）将逐词元平均负对数似然指数化：

$$\mathrm{PPL} = \exp\left(-\frac{1}{T}\sum_{t\in\mathcal{S}} \log p(x_t \mid x_{<t})\right), \qquad T = |\mathcal{S}|. \tag{9.7}$$

$\mathcal{S}$ 是实际计分位置集合．分词器、文本、上下文窗口、模板、起止符和掩码共同规定“每词元”的单位；重叠滑窗不能无意重复计分．跨分词器的 PPL 不能直接按数值大小比较模型能力．其概率模型基础见第八章《自回归语言建模》．

完全匹配（Exact Match，EM）比较规范化后的答案是否相等．规范化必须声明大小写、空白、标点、数值单位及多参考规则．**词元 F1**（Token F1）令候选和参考的词元多重集合重合数为 m，候选与参考长度分别为 c, r，则 $P = \frac{m}{c}, R = \frac{m}{r}, F_1 = \frac{2m}{c+r}$．重复词元的匹配次数不能超过参考中的出现次数．

双语评估替代指标（Bilingual Evaluation Understudy，BLEU）使用经参考计数

截断的 n 元组精确率 p_n，并施加长度惩罚 [58]：

$$\mathrm{BLEU} = \mathrm{BP}\exp\left(\sum_{n=1}^{N_g} w_n \log p_n\right), \quad \mathrm{BP} = \begin{cases} 1, & c > r, \\ e^{1-\frac{r}{c}}, & 0 < c \leqslant r. \end{cases} \tag{9.8}$$

$w_n \geqslant 0$ 且总和为 1，c, r 为按协议得到的候选和有效参考长度. 通常需在语料层累加计数，而非随意平均逐句 BLEU. [2]零匹配时的平滑、空输出和多参考长度选择都应固定.

面向召回的摘要评估指标（Recall-Oriented Understudy for Gisting Evaluation，ROUGE）是指标族 [59]. ROUGE-n 的基本召回形式为匹配 n 元组数除以参考 n 元组数；ROUGE-L 则利用最长公共子序列，具体输出召回、精确率或 F 分数仍须声明. 词面相近可以对应事实相反，否定词、数值和单位错误也可能只改变很少词元.

以参考词元序列“甲乙丙丁”、候选“甲乙戊”为例，$m = 2, c = 3, r = 4$，Token F1 为 $\frac{4}{7}$，ROUGE-1 召回率为 $\frac{1}{2}$，EM 为零. 只计算一元与二元 BLEU、各权重 $\frac{1}{2}$ 且不平滑时，$p_1 = \frac{2}{3}, p_2 = \frac{1}{2}$，故分数为 $e^{-\frac{1}{3}}\sqrt{\frac{1}{3}} \approx 0.4137$. 这些数字描述不同计数结构，没有一个直接回答“戊是否为错误事实”.

9.4.2　事实、执行及拒答

事实评估应把回答分解为可判定论断，分别标记证据支持、冲突与未知. 引用存在、来源相关和来源支持论断是三个不同事件. 引用精确率的分母可以是已给出的引用，论断覆盖率的分母则是需要证据的论断；二者不得混用. 检索覆盖与答案忠实性的关系见第十二章《信息检索》及第十三章《检索增强生成》.

对于数学、代码和结构化操作，**可执行验证器**（Executable Verifier）以可执行约束或测试判定结果. 测试通过仍受测试覆盖限制，不能一般化为程序完全正确. 评估执行环境需隔离待测代码和副作用，超时、异常及无效输出必须按预定规则进入分母，而不是只保留成功返回的候选.

若对一题独立同分布生成 n 个候选，其中 c 个通过固定验证器，从这些候选均匀无放回选 $k \leqslant n$ 个，全部失败的子集数为 $\binom{n-c}{k}$，总子集数为 $\binom{n}{k}$，故

$$\widehat{\mathrm{pass@}k} = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}. \tag{9.9}$$

在同分布独立采样前提下，这个子集平均对 k 次独立尝试至少一次通过的概率无偏 [44]. 例如 $n = 5, c = 2, k = 2$，结果为 $1 - \frac{3}{10} = 0.7$；直接代入样本通过比例得到

[2]BLEU 的名称来自机器翻译评估. 将其用于其他生成任务时，仍需论证参考文本及词面重合与实际任务质量的关系，不能仅沿用一个熟悉的指标名称.

$1-(\frac{3}{5})^2=0.64$，两种估计并不相同. 相关采样、不同温度混合或自适应重试会改变目标分布. 即使候选集中有正确答案，也不代表产品能识别并选择它，故 pass@k 不是验证选择后的单次交付成功率.

拒答评估还应区分不可回答任务上的正确弃权，与可回答任务上的过度拒绝. 安全失败、事实错误和风格偏好应分维度报告，不能以更流畅的文风抵消一个禁止行动.

9.5 评审员测量模型

9.5.1 人工量表及模型裁判

评分量表（Rubric）需要将“好回答”拆成可观察维度，为各等级提供锚点、正反例及不适用规则. 人工评审先独立评分再仲裁，保留原始分歧；只保存仲裁标签会隐藏量表歧义. 盲评隐藏候选身份、运行成本等无关信息，候选顺序随机化，必要时限制高风险材料的访问与保留.

模型评审（LLM-as-a-Judge）可扩展开放式判断，但会受位置、长度、风格、自我偏好和提示注入影响 [60]. 交换顺序后必须先映射回原候选身份，再比较判定. 相同事实的长短改写可以识别冗长偏好；跨模型家族的交叉设计用于识别自我偏好. 一个总体人工一致率无法替代这些专门对照.

对人工判定不通过的样本，假通过率定义为 $\Pr(J=1\mid H=0)$；在裁判判定通过的样本中，错误比例则为 $\Pr(H=0\mid J=1)$，两者分母不同. 裁判模型、量表、语言或候选输出分布改变后，应重新估计这两个量. 只有两个顺序都通过才接受，是一种保守决策规则，可能降低假通过，也可能提高假拒绝；两个判断通常相关，不能把两次错误率直接相乘.

9.5.2 一致性及胜率计算

科恩一致性系数（Cohen's Kappa，κ）以边际分布为基准校正观测一致率 [61]. 若观测一致率为 p_o，两位评审在类别 c 的边际比例为 p_{1c},p_{2c}，则

$$p_e=\sum_c p_{1c}p_{2c},\qquad \kappa=\frac{p_o-p_e}{1-p_e},\quad p_e<1. \tag{9.10}$$

例如 100 条二分类评分中，两人都判通过 40 条、都判不通过 30 条，甲通过乙不通过 20 条，甲不通过乙通过 10 条. 于是 $p_o=0.7$，甲通过率 0.6、乙通过率 0.5，$p_e=0.6\times0.5+0.4\times0.5=0.5$，$\kappa=0.4$. 它衡量超过特定偶然基准的一致程度，不是二者的准确率；类别极不均衡会改变这个基准. 序数、多评审和缺失评分需要适合其结构的统计量.

成对偏好中若胜、负、平分别为 W,L,T，半分平局胜率为 $\dfrac{W+\frac{T}{2}}{W+L+T}$. 例如

$(12, 6, 2)$ 得到 0.65．单条贡献可以是 $0, \frac{1}{2}, 1$，因此它是有界均值而非普通伯努利比例，不能直接把“13 胜 20 题”当作真实二项计数套用所有二项区间．

9.6　区间估计

9.6.1　Wilson 区间推导

独立二项比例

本节假设每个统计单位产生一次通过或失败，单位相互独立、通过概率相同，样本来自目标抽样机制，系统与判定规则已冻结．多轮会话、重复用户和多个随机种子并不自动满足这些条件．

令 $K \sim \mathrm{Binomial}(n, p)$，$\hat{p} = \dfrac{K}{n}$，则 $\mathbb{E}\hat{p} = p$、$\mathrm{Var}(\hat{p}) = \dfrac{p(1-p)}{n}$．直接把 p 换成 $\hat{p}$ 得到 Wald 区间，在边界处可能退化或越界．**威尔逊区间**（Wilson Interval）通过反解关于候选 p 的分数检验不等式获得 [62]：

$$(\hat{p} - p)^2 \leqslant \frac{z^2 p(1-p)}{n}, \tag{9.11}$$

$$(n + z^2)p^2 - (2n\hat{p} + z^2)p + n\hat{p}^2 \leqslant 0. \tag{9.12}$$

取二次方程两根之间的区间，得到

$$\frac{\hat{p} + \frac{z^2}{2n} \pm z\sqrt{\frac{\hat{p}(1-\hat{p})}{n} + \frac{z^2}{4n^2}}}{1 + \frac{z^2}{n}}, \qquad z = z_{1-\frac{\alpha}{2}}. \tag{9.13}$$

这是依赖正态分位数近似的区间，不是任意 n, p 下覆盖率恰好等于标称值的保证．**置信区间**（Confidence Interval，CI）的频率学派解释是：在相同抽样机制反复取样并按同一规则构造区间时，长期覆盖固定真值的比例接近声明水平．它不表示观察到本区间后，固定参数以 95% 的概率随机落在其中．

对于 $n = 20, k = 20, z = 1.96$，上界为 1，下界为 $\dfrac{20}{20 + 3.8416} \approx 0.8389$．二十次全通过仍允许相当大的未观测失败率．另从零失败的精确二项关系 $(1-q)^n = \alpha$，得到失败率 q 的单侧上界 $1 - \alpha^{\frac{1}{n}}$；当 q 小时近似为 $-\dfrac{\log \alpha}{n}$，95% 时约为 $\dfrac{3}{n}$．单侧界与前面的双侧 Wilson 界使用不同构造，不能混为一个指标．

区间仅反映给定模型下的抽样误差．判定器偏差、样本代表性和训练污染不会因为 n 很大就自动消失．

9.6.2 配对差及效应大小

同一批案例比较 A 与 B，令 $d_i = s_i^{(A)} - s_i^{(B)}$. 在独立案例和有限方差条件下，

$$\bar{d} = \frac{1}{n}\sum_i d_i, \quad s_d^2 = \frac{1}{n-1}\sum_i (d_i - \bar{d})^2, \quad \widehat{\mathrm{SE}}(\bar{d}) = \frac{s_d}{\sqrt{n}}. \tag{9.14}$$

其优势来自 $\mathrm{Var}(A-B) = \mathrm{Var}(A) + \mathrm{Var}(B) - 2\,\mathrm{Cov}(A,B)$. 两模型通常同时在难题上失败，正相关可以降低配对差的方差；把结果当作无关样本会丢弃这部分信息.

例如 100 个独立案例中，A 独胜 15 条、B 独胜 5 条，其余 80 条一致. 差值总和 10、平方和 20，所以 $\bar{d} = 0.10$，$s_d^2 = \dfrac{20 - 100 \times 0.1^2}{99} = \dfrac{19}{99}$，标准误约 0.0438，正态近似 95% 区间约为 $[0.014, 0.186]$. 该算例支持正向平均差，但区间下界未达到假设的最小实用提升 0.02，因而“区别于零”和“达到实用标准”是不同结论.

置换检验（Permutation Test）在零假设下交换配对中的系统身份，并比较统计量. 独立配对且零假设赋予交换对称性时，可枚举或抽样翻转 d_i 符号；仅假设均值为零而不具备相应对称或随机化条件，并不足以自动保证任意符号检验有效. p 值是零假设及检验模型下至少同样极端结果的概率，不是零假设为真的概率.

9.6.3 聚类、Bootstrap 及多种子

自助重采样（Bootstrap）从经验样本重复有放回抽样，以近似统计量的抽样分布 [63]. 计算非线性指标如 Macro-F1 时，应在每次重采样后重算指标，而非只重采样一个已经聚合的数字.

簇重采样（Cluster Bootstrap）将用户、文档等独立组作为抽样单位，保留组内记录共同出现. 若每组有 m 条记录，单条方差 σ^2，组内任意两条相关系数为 ρ、组间独立，则 $n = Gm$ 条记录均值的方差为

$$\mathrm{Var}(\bar{X}) = \frac{1}{n^2} G\left[m\sigma^2 + m(m-1)\rho\sigma^2\right] = \frac{\sigma^2}{n}[1 + (m-1)\rho]. \tag{9.15}$$

方括号是这一简化模型的设计效应. 例如 10 位用户各 10 条请求、$\rho = 0.5$，设计效应为 5.5，按方差等价得到有效样本量约 $\dfrac{100}{5.5} = 18.2$，不能视为 100 个独立用户.

组大小不一致时还必须明确估计请求平均还是用户平均. 二者分别为

$$\bar{s}_{\text{request}} = \frac{\sum_g \sum_i s_{gi}}{\sum_g m_g}, \qquad \bar{s}_{\text{user}} = \frac{1}{G}\sum_g \frac{\sum_i s_{gi}}{m_g}. \tag{9.16}$$

按用户重采样可以支持两种目标，但每次聚合的权重必须匹配预定目标.

多个案例、多个种子构成交叉重复测量. 同一案例跨种子相关，同一种子运行内也可能共享状态. 只在每个种子内部聚合再按种子重采样，回答的是固定测试集上的运行变异；若要推断新案例总体，还需按案例来源重采样，或采用适合交叉结构的分层模型与双向方案. 增加重采样次数只能减少 Monte Carlo 误差，不能创造更多独立用户或独立种子.

算法 9.2　配对簇重采样区间

输入：两方案在同一组案例上的逐例结果、独立簇标识、目标聚合函数 T、重复次数 R、水平 $1-\alpha$.

输出：配对差点估计及百分位区间.

状态：完整的 G 个簇与重采样差值列表.

1. 固定缺失、超时和不适用规则；以相同案例键对齐两方案. 确认簇的独立性依据及聚合权重.
2. 计算原始差 $\hat{\Delta} = T(A) - T(B)$.
3. 对 $r = 1, \ldots, R$，从 G 个簇有放回抽取 G 次；重复抽到的簇作为重复副本加入，保留其全部配对记录.
4. 在同一组簇副本上分别计算 $T(A^*)$ 与 $T(B^*)$，保存差值. 不能独立为两个方案抽样.
5. 取差值分布的 $\frac{\alpha}{2}$ 与 $1-\frac{\alpha}{2}$ 分位点，连同 G、R、分位数约定及点估计返回.

停止与边界：完成 R 次后停止；若独立簇太少或统计量在大量副本中未定义，应报告不稳定及原因，不用更多重复次数掩盖问题. 百分位方法也有有限样本和偏差局限.

9.7　污染、切片及选择偏差

训练集用于拟合，开发集用于方案选择，校准集用于阈值或裁判标定，最终测试集评估冻结方案. 公开题目、答案解析、近重复、来源标题以及文件名都可能泄露信息. 测试污染不仅是文本完全相同，也可能是同一模板、同一文档或答案结构进入开发过程. 数据血缘与分组切分机制见第十八章《训练数据工程基础》.

切片评估（Slice Evaluation）按语言、领域、长度、风险或能力结构分析子群. 若各互斥切片权重为 w_j、得分为 μ_j，总体为 $\mu = \sum_j w_j \mu_j$. 假设简单任务得分 0.95、困难任务 0.50，简单任务占 80% 时总体 0.86，占 50% 时总体 0.725. 模型在任何切片都未变化，总分却下降 0.135. 因此比较时需同时报告切片得分和构成变化.

重叠切片不能把样本数相加当作独立总量. 事后发现的切片适合诊断，应标记探索性并在新数据中复验. 高严重度失败的硬性约束需要由任务损失定义；某个教学门槛或“每组 30 条”不是统计充分性的普遍定理.

反复测试许多方案再报告最大分数会产生选择偏差. 若 M 个有效检验各用水平 α，即使每个零假设都成立，也不能把整体误报率当作 α. Bonferroni 原则给每项 $\frac{\alpha}{M}$，利用并集上界控制总体误报概率，但可能保守. 实际应预先声明主要目标、保留独立

最终集，并将探索结果与确认性结论分开．未显著不等于等效；非劣性要求相对于预定损失界单独设计比较．

9.8 在线实验及多目标决策

9.8.1 随机化及因果效果

随机对照实验（Randomized Controlled Experiment）将合格单位随机分配到不同系统．令 $Y_i(1), Y_i(0)$ 为同一单位在两种系统下的潜在结果，目标平均处理效应为 $\mathbb{E}[Y(1) - Y(0)]$．随机分配使分组与潜在结果独立；在无干扰、处理定义一致和观测完整等条件下，组均值差估计该效应．

随机化单位应与记忆和干扰结构一致．同一用户跨方案可能发生学习、缓存或会话串扰；用户之间存在网络效应时，仅按用户分流也未必足够．稳定哈希可以实施固定分组，但不能替代对身份、排除规则、互斥实验和样本比例异常的处理．

例如两组各 1000 个独立用户，A 成功 600、B 成功 650，差值为 0.05．用独立二项方差估计，标准误为

$$\sqrt{\frac{0.6 \times 0.4}{1000} + \frac{0.65 \times 0.35}{1000}} \approx 0.0216, \tag{9.17}$$

近似 95% 区间为 $[0.0076, 0.0924]$．这与离线同题配对是不同设计，不能套用同一协方差结构．若目标是至少提高两个百分点，这个区间仍不足以证明达到目标．

意向处理分析（Intention-to-Treat Analysis，ITT）按最初分配组统计效果，避免只保留成功返回或完成使用的样本所造成的处理后选择偏差．超时、未完成请求和标签延迟需按预先协议处理．固定观察窗口、样本量和停止规则后再查看结果；未经校正地反复查看显著性并提前停止，会改变误报率．

在两组样本量相等、共同基率近似为 p 的简化设计中，检测绝对差 δ、双侧水平 α、功效 $1-\beta$，每组样本量的粗略正态近似为

$$n \approx \frac{2p(1-p)(z_{1-\frac{\alpha}{2}} + z_{1-\beta})^2}{\delta^2}. \tag{9.18}$$

例如 $p = 0.5, \delta = 0.05, z_{0.975} = 1.96, z_{0.8} \approx 0.84$，得到每组约 1568．聚类、损耗、稀有事件或多重指标会改变设计，不能把这个近似值当作所有实验的通用要求．

9.8.2 质量、风险、延迟及成本

对质量越高越好、成本与延迟越低越好的目标，若 A 各项不劣于 B 且至少一项严格更优，则 A 支配 B．**帕累托前沿**（Pareto Frontier）由所有不被支配的方案构成．构造三个方案：A 为质量 0.90、成本 2、延迟 200；B 为 0.92、3、240；C 为 0.88、

2.5、220，其中成本和延迟均采用同一任意单位．A 支配 C，A 与 B 互不支配．若质量下限 0.91，A 虽在前沿上仍不可行．

成本需包含生成、检索、评审、人工与失败重试；延迟需固定输入长度、输出长度、并发、缓存及工具路径．采用最近秩定义时，P95 是排序后第 $\lceil 0.95n \rceil$ 项；小样本尾部分位数不稳定，也不宜与不同实现的插值分位数混比．质量与成本均带不确定性时，还应避免仅按点估计宣布稳健支配．

证据不足是一种有效决策

发布判断需要同时满足适用的硬约束、效应标准与证据强度．平均分提高不能抵消未满足的权限约束，未发现失败不能证明风险为零，未达到显著性也不能证明两系统等效．保持当前方案并补充针对性证据，可以是统计分析直接支持的结果．

可复现记录应绑定数据快照、模型制品、提示与索引、生成条件、评审协议、逐例输出、统计计划和环境．哈希证明引用的是同一制品，不证明制品质量合格．[3]保留逐例记录才能重算指标并追溯失败，同时应控制敏感评测材料的访问与删除．上线后输入分布、标签和裁判误差持续变化，重新评估应针对实际变化确定范围．

[3]固定随机种子主要用于约束与重放随机过程；跨硬件、框架和并行内核时，仍可能存在数值差异，故复现实验还需声明环境与容差．

第十章　模型安全

模型能够正确生成工具调用格式，不代表它有权执行该调用；模型在一组测试中没有输出敏感信息，也不代表系统不会通过缓存、日志或检索泄露信息．安全问题发生在模型、数据、身份和执行能力相互连接的位置，因此必须以完整系统为分析对象．

本章承接第十三章《检索增强生成》中的知识与权限边界，把相同原则扩展到训练数据、模型制品、工具执行和运行治理．目标是建立可解释的威胁模型、可检查的控制以及可恢复的运行机制；不把提示词、分类器或一项基准结果当作整个系统的安全证明．

10.1　安全问题的对象及范围

10.1.1　可靠性、行为安全、防护及隐私

可靠性（Reliability）关注系统在给定条件下能否持续完成预期功能．**行为安全**（Safety）关注失败和误用是否造成不可接受的伤害．**安全防护**（Security）关注主动对手能否破坏资产、权限和系统行为．**隐私**（Privacy）关注个人信息如何被收集、使用、保留和披露．这些目标相互关联，却没有共同的单一分数．

例如，系统可能稳定执行错误授权，这是一种可靠的越权行为；模型也可能在无人攻击时给出有害建议．加密传输可以保护链路，却不能阻止已授权服务把过量数据放入提示．准确率、内容拒绝率、权限控制和数据最小化分别回答不同问题．

保障论证（Assurance Case）将目标声明、适用假设、控制和证据连接起来．例如“租户 A 不能读取租户 B 的文档”应由服务身份绑定、数据层授权、缓存隔离及对应测试支持，不能只由系统提示的一句话支持．论证同时记录残余风险、责任人和失效时的处理方式，使结论有明确边界．

本章以 NIST AI RMF 1.0（2023 年）与生成式人工智能配置文件 NIST AI 600-1（2024 年）作为风险治理参考，以 OWASP 2025 版相关条目辅助检查应用风险 [64, 65, 66]．这些资料提供组织与技术视角，不会自动替具体系统完成风险分析．[1]

[1]框架的版本与网页更新时间不同．本章标明所引用文档版本，不把框架名称视为某一部署已经合规或获得认证的证据；实际法律义务需按部署场景和适用地区另行核对．

10.1.2　威胁影响

威胁模型（Threat Model）说明要保护的资产、对手能力、攻击入口、信任边界及可能后果. 资产不仅是权重和用户数据，还包括分词器、模板、适配器、索引、会话记忆、工具凭据、日志和算力预算. 攻击者可能只控制一页公开网页，也可能能提交训练样本、修改软件依赖或使用一个普通租户账号；不同能力会产生不同可达路径.

信任边界（Trust Boundary）是信息或能力从一个授权与可信假设范围进入另一个范围的位置. 外部文件进入解析器、检索文本进入模型、模型建议进入执行器，都是需要独立检查的边界. 数据在前一阶段通过检查，不意味着后续任何用途都被批准. 例如允许读取某份文件，并不等于允许将其全文发送给第三方.

治理先确定使用边界、责任和风险接受权限；场景分析识别资产与后果；测量通过评测、红队和运行数据形成证据；管理则决定缓解、限制能力、接受残余风险或停止使用. 四类活动相互反馈，不是一次通过后永久结束的流水线. 每次数据、模型或工具变化都应明确哪些既有证据仍有效，哪些需要补充.

风险登记应先描述不可接受后果，再沿系统数据流寻找可达路径. 一个完整条目包含资产、主体、前提、入口、影响、现有控制、验证证据、负责人和处理状态. 空白条目表示尚未分析，不表示没有风险. 扩展模型能力、接入工具、更换检索源和改变用户群都会改变威胁模型，不能只在首次发布时登记一次.

本章风险与授权模型的前提

概率推导以明确的请求总体、时间窗口和损失单位为前提；构造数字仅说明计算方法. 授权由可信身份和资源状态提供，模型不能自报身份或提升权限. 攻击者可控制其获准提交的输入和部分外部内容，但不能任意修改受保护的授权引擎；若该假设失效，必须另行分析控制平面的失陷.

表 10.1: 风险、策略与执行记录使用的主要符号. 序数等级不具有概率单位.

符号	含义与范围
u, a, r, c	已认证主体、动作、资源与运行上下文.
P_v, Allow	版本为 v 的策略及其资源级授权判定.
ℓ, L_i, p_i	随机损失、场景损失和概率，$L_i \geqslant 0$、$p_i \in [0,1]$.
F_j	第 j 个控制层未阻止指定攻击的事件.
$w_i, \mathcal{A}_i$	预先确定的风险权重和第 i 个样本的允许决策集合.
N, x, α	评测试验数、失败数和单侧置信界显著性水平.
h_a, k	最终动作摘要与幂等键，分别绑定动作内容和逻辑执行请求.

10.2　风险度量边界

10.2.1　期望损失及主观等级

若场景构成互斥划分，场景 i 发生概率为 p_i，其条件平均损失为 L_i，则

$$\mathbb{E}[\ell] = \sum_i p_i L_i. \tag{10.1}$$

若各类损失是可相加的不同分量，也可利用期望的线性性分别求和，但必须避免同一后果被多个攻击路径重复计数. 损失单位可以是同一种货币或其他明确计量；无法合理折算的生命、隐私与业务影响，应分别呈现，不必强行合并.

例如构造一个季度风险模型：某类事件概率 0.02，发生后的平均恢复损失为 50000 个成本单位，则该分量的期望为 1000. 若概率只知道落在 0.01 至 0.04，损失在 30000 至 80000，则可给出 300 至 3200 的粗范围，不能保留 1000 的点估计却隐去不确定性. 低概率而不可接受的后果还可设独立能力限制；平均损失不能替代组织对不可接受后果的判断.

“可能性 1 至 5、严重性 1 至 5、暴露程度 1 至 5” 的评分属于序数尺度. 乘积可作为预先约定的排序启发式，但不具备真实期望损失语义，排序甚至对保持顺序的重新编码不稳定. 两个条目为 $(1,5)$ 和 $(2,3)$，乘积分别 5 和 6；若只把最高等级 5 重新编码为 100，等级顺序未变，乘积却变为 100 和 6. 这个反例说明评分结果依赖量表约定，不能用乘积差一倍声称风险差一倍.

10.2.2　控制效果及相关失败

设 R_0, R_1 为同一总体在无某控制和有该控制时的期望损失，且 $R_0 > 0$，定义有效率 $e = 1 - \dfrac{R_1}{R_0}$，才有

$$R_1 = R_0(1 - e). \tag{10.2}$$

它是根据已定义量得到的恒等关系；难点是获得可比较的反事实估计. 把专家打分的“有效性 80 分” 直接替换为 $e = 0.8$ 没有统计依据. 若控制使攻击转移到别处，或改变了用户总体，两个风险也不再天然可比.

纵深防御（Defense in Depth）使用多个控制点限制攻击影响. 若攻击成功需要所有层都漏放，则

$$\Pr(\cap_{j=1}^{m} F_j) = \Pr(F_1) \prod_{j=2}^{m} \Pr(F_j \mid F_1, \ldots, F_{j-1}). \tag{10.3}$$

只有独立失败时才可将条件概率替换为边缘概率. 两个各有 0.1 漏放率的过滤器，独立时联合为 0.01；若它们在相同攻击上一起失败，联合仍为 0.1. 对两个事件，在只知道边缘概率时有

$$\max(0, p_1 + p_2 - 1) \leqslant \Pr(F_1 \cap F_2) \leqslant \min(p_1, p_2). \tag{10.4}$$

因此堆叠相同模型的多次判断，不会自动产生乘法安全收益．更有价值的组合是不同失效机制，例如身份校验、资源范围限制、隔离执行、内容判断与审计，并在完整链路上验证．

10.3　提示注入及指令边界

10.3.1　直接及间接注入

提示注入（Prompt Injection）利用模型可能把不可信数据解释为有效指令的机制，诱导其偏离授权任务．**直接提示注入**（Direct Prompt Injection）来自用户提交的输入；**间接提示注入**（Indirect Prompt Injection）来自网页、文件、检索材料或工具返回值等外部来源．研究已经展示间接内容能够影响集成语言模型的应用 [67]．本章关心的是数据如何越过权限边界，而不是某一组固定攻击措辞．

并非所有命令式文字都是攻击．用户可能要求分析一份含指令的文档；这时文档内容是分析对象．反过来，文字看起来像普通说明也可能试图改变输出目的．字符关键词、语言和排版均不能完整表达授权关系．图像内文字、音频转写和工具结果同样可能携带这类内容，跨模态输入不会消除边界问题．

越狱（Jailbreak）通常指诱导模型突破其行为约束；它与提示注入可能重叠，但不是完全相同问题．即便模型没有生成行为政策禁止的文字，也可能把内部摘要发送到错误资源；即便某次越狱只改变了聊天内容，也不代表执行器被攻破．评估必须分别记录模型行为偏离和实际权限或数据影响．

10.3.2　内容可信度及执行权限分开

系统可以通过角色分隔、来源标记和训练改善模型对指令层级的理解，但文本标记不是密码学身份．外部文档不能修改认证主体、租户、工具白名单或审批规则；模型生成的“已批准”也不是批准凭据．真正的权限应保存在模型可读文本之外的受保护状态中．

输出检查同样不能替代数据最小化．如果模型上下文含有它不需要的秘密，任何过滤失误都会扩大泄露后果．更强的边界是不给模型不必要的秘密；工具凭据由执行器临时取得，而非写入系统提示．系统提示可能包含知识产权或内部规则，但不能作为密钥库或唯一安全屏障．

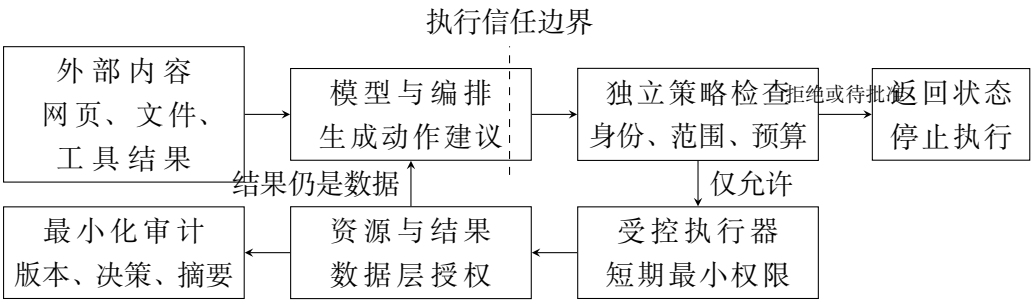


图 10.1: 模型建议到执行能力之间的独立边界. 身份和凭据由控制层提供，外部内容不能授予权限.

10.3.3 风险传播链

考虑构造的内部知识助手：用户只要求总结某项目的内部报告，检索系统找到一份附有恶意说明的文档. 模型把说明当作额外任务，提出向外部地址发送完整报告. 若编排层只检查调用是合法 JSON，并使用拥有广泛发送权限的服务账号执行，就会把文本误解转化为数据外发.

此失败链至少包含四个不同环节：外部内容影响任务目的；模型提出超出原请求的动作；策略层没有验证用途、收件范围和数据级别；执行凭据允许现实外发. 关键词过滤可能阻止第一步中的部分变体，却无法独立保证后三步. 相反，即使模型仍提出错误建议，外部发送能力默认不可用、目的地址受限和数据出域策略也可阻止该后果.

另一条路径不需要显式发送工具：模型生成包含外部资源的富文本，客户端自动加载时可能发起网络请求. 因此输出消费位置也属于威胁模型，应限制自动远程资源加载、按目标格式进行安全渲染，并避免将敏感内容放入可被外部访问的地址参数. 不能只保护工具网关，却假定展示层没有副作用.

10.4 工具授权及执行事务

10.4.1 外部授权判定

令 u 为已认证主体，a 为动作，r 为规范化资源，c 包含租户、时间、用途和环境. 执行许可至少满足

$$\mathrm{Execute}(u,a,r,c) \Longrightarrow \mathrm{Allow}_{P_v}(u,a,r,c)=1. \tag{10.5}$$

这是必要条件而非充分条件，还要检查参数模式、业务约束、预算与操作前置状态. 主体来自认证会话，资源归属来自可信数据层，不能信任模型参数中的自报字段. 用户授权、服务身份能力和当前任务范围应取交集，而不是任选一个足够宽的权限.

最小权限（Least Privilege）要求只提供完成任务必要的资源与动作范围．工具接口宜以明确业务能力为边界，而非暴露任意命令执行器．读取、修改、外发和权限变更具有不同影响，应使用独立能力和凭据范围．OWASP 2025 版将过多功能、过宽权限和过度自治列入过度代理风险讨论 [68]．

结构模式（Schema）校验约束字段、类型和取值形式，但合法格式仍可能引用错误租户、过量资源或不允许的目的地．资源标识必须规范化后验证，并在执行时使用同一个规范结果．先校验一个地址、再跟随未经验证的重定向，或者先检查路径字符串、再解析到其他对象，都会破坏这一对应关系．

10.4.2　动作级批准

对需要批准的高影响或不可逆操作，应在内容、对象、参数和影响已经确定后展示给具有相应权限的人．已存在且覆盖当前动作的授权不需要重复索取，但不能把模糊任务目标扩张成无限授权．批准记录可绑定规范化动作摘要

$$h_a = H(a, r, \text{params}, u, \text{scope}, v, \text{expiry}), \tag{10.6}$$

其中 H 表示确定的摘要构造．参数、资源或权限状态改变后，原批准可能不再有效，需要重新判定．摘要能绑定内容，批准本身仍须由可信身份签发和验证．

检查与使用时差（Time-of-Check to Time-of-Use，TOCTOU）指检查状态与实际执行间状态发生变化的风险．例如文档所有权在批准后改变．执行器应在执行点重新验证权限和必要资源版本，或使用数据库条件写入、事务与能力凭证绑定约束，避免把过期检查当作当前事实．

10.4.3　幂等、超时及未知结果

幂等性（Idempotency）要求同一逻辑请求的重复提交不重复产生业务副作用．它需要稳定幂等键 k、动作摘要和执行结果的绑定，而不只是客户端随机生成一个字符串．相同键对应不同参数应拒绝；同一请求重试应查询或复用原执行记录．

工具超时可能发生在副作用已经完成、结果尚未返回之后．因此“超时”不等于“未执行”．状态应区分已准备、已授权、执行中、成功、明确失败和结果未知．未知状态先查对账或结果接口，不能直接用新幂等键重新发起．补偿操作也不等于真实回滚：已经发送的数据无法靠撤销调用收回，某些业务动作只能进行后续补偿．

算法 10.1　受控工具执行

输入是已认证请求、模型动作建议和当前策略；输出是拒绝、待批准、执行结果或结果未知状态．

1. 从受保护会话取得主体与任务范围，解析模型建议；任何身份或租户自报字段都不能覆盖会话状态．

2. 规范化资源与参数，验证结构、范围、用途、资源归属及预算；策略明确拒绝或必需状态不可取得时停止.
3. 计算最终动作摘要. 若策略要求批准，验证批准主体、摘要、范围及有效期；没有满足条件的批准则返回待批准状态并停止.
4. 以“租户与幂等键”的唯一约束，原子建立并认领绑定动作摘要的执行记录；只有认领成功的执行者进入发送步骤. 已有记录时返回已知结果或查询执行状态；相同键不同摘要则拒绝，不通过新建动作绕过冲突.
5. 在执行点重查当前授权与资源前置条件，以最小权限短期凭据调用指定工具，不向模型暴露凭据.
6. 将成功、明确失败或结果未知写入账本. 未知结果进入对账流程；重试遵守相同逻辑请求和预算，不重复创建副作用.
7. 按数据披露策略处理工具结果，记录最小化审计证据，再返回编排层. 工具返回文本不改变后续授权规则.

该算法要求账本支持原子认领，远端工具对同一动作键提供幂等提交或可靠的结果查询. 认领后崩溃和发送后超时均须通过原动作身份恢复；远端既不支持幂等、也不能核对结果时，应保持结果未知并停止自动重发，不能仅凭本地唯一约束宣称外部恰好执行一次. 持久执行、发件箱与隔离代数的机制见第十六章《智能体运行系统》.

不变量是未经授权的动作不能到达执行器，批准与实际参数一致，并在上述远端契约成立时使同一逻辑请求的重试不产生额外副作用. 系统中存在绕过网关的其他执行路径时，必须纳入同一控制.

10.5 数据及模型供应链

10.5.1 投毒、后门及评测污染

数据投毒（Data Poisoning）通过改变训练或检索数据影响系统行为. **后门**（Backdoor）使模型在特定触发条件下产生攻击者希望的行为，同时可能在普通样本中表现正常. 两者的关系取决于攻击路径：后门可以通过训练数据、参数或依赖引入，数据投毒也可能只降低整体质量而不形成触发机制.

数据清单应保存来源、取得依据、用途、时间范围、处理版本、敏感等级和内容标识，并建立样本到数据快照、训练任务与模型版本的关系. 来源集中度、重复率、标签变化和分布异常可帮助发现问题，但这些检测无法保证发现所有隐蔽投毒. 下载位置公开不等于内容适合训练，也不等于来源拥有授权.

评测污染（Evaluation Contamination）使测试材料进入训练或调参过程，从而削弱评测结论．安全数据尤其需要控制：反复使用同一攻击模板优化，可能只提高已知变体拦截率．应独立维护训练、开发、保留测试和历史回归集合；将新事件转为回归用例后，它不再代表完全未知攻击．

10.5.2　制品性校验

制品完整性（Artifact Integrity）关注实际加载内容是否与指定制品一致．内容哈希可以检测字节变化，但攻击者若同时替换文件和未受保护的摘要，哈希相符也没有来源保证．**数字签名**（Digital Signature）把制品声明绑定到签名主体，但仍需验证签名身份、信任链、签名用途和撤销状态．可信发布者签名也不证明模型行为无害．

权重之外，配置、分词器、会话模板、适配器、处理器、自定义代码和运行依赖都会影响行为．**软件物料清单**（Software Bill of Materials，SBOM）列出软件组件及版本关系，模型制品清单还应补充权重与数据、评测和处理器关系．OWASP 2025 版供应链条目强调预训练模型、数据和第三方组件带来的风险 [69]．清单是追溯证据，不是自动通过发布的证明．

Safetensors 采用以张量为中心的序列化方式，旨在避免依赖任意对象反序列化来加载权重 [70]．它减少一类加载风险，却不审查同仓库的自定义代码、模板或模型中的行为后门．远端代码执行能力应默认关闭；业务确实需要时，按固定代码版本审查，并在隔离构建与加载环境中限制凭据和网络．

10.5.3　构建、评测及部署绑定

不可变发布应把权重、配置、处理器、模板、依赖锁定文件和镜像摘要绑定为一个版本集合．评测清单引用的摘要必须与部署集合一致，否则“模型名相同”不足以支持复用证据．改变适配器或会话模板也可能改变安全行为，应按影响重新评估．

构建环境需要限制外部执行和秘密访问，加载环境需要限制文件与网络能力，运行环境需要限制服务身份．漏洞扫描只能检查其覆盖的已知问题，行为评测只能覆盖相应任务分布；二者应互补．回滚时必须明确是回滚权重、索引、模板、策略还是完整制品集合，不能恢复了旧权重却继续加载受污染的适配器．

10.6　隐私及多租户隔离

10.6.1　数据最小化

个人可识别信息（Personally Identifiable Information，PII）包括能够识别或关联个人的信息，其具体范围应按场景确定．隐私控制从明确用途和最小必要数据开始，

再处理访问、保留和披露. 正则脱敏可能漏掉间接标识，也可能删除任务所需内容；一次脱敏不等于整个数据生命周期受到保护.

将姓名替换为稳定标识属于去标识处理的一种形式，但映射表或其他字段组合仍可能恢复关联. 向量、摘要、缓存和调试日志也可能保留敏感信息，不能因为不是原文就自动视为公开数据. 加密保护传输和存储状态，对获得解密权限的服务仍需数据范围和用途限制.

训练模型还可能记忆样本信息. 删除原始记录可以阻止继续摄取，却不自动删除已训练参数中的影响；需要依据实际训练与删除机制分析是否重训、使用专门方法或限制模型使用. 不得把数据目录中的删除状态解释为已经完成参数层面的精确遗忘.

10.6.2 多租户状态隔离

租户标识来自服务端认证状态，并贯穿文档读取、向量索引、对象存储、会话记忆、答案缓存和遥测. 向量检索后的最终过滤不能挽回已经传给外部重排器的未授权内容. 缓存键还应考虑授权范围和策略版本；权限撤销应及时阻止旧缓存读取，不等待普通到期时间.

会话记忆的写入同样是数据变更. 外部文档不能把自己声明为长期系统规则；记忆应保留来源、作用域、有效时间和删除路径. 跨会话共享摘要必须有明确的数据授权，不能只凭模型判断“这条信息以后有用”.

密钥应由专用系统向执行器提供短期、窄范围能力. 生产、测试和不同工具的服务身份分开，可以限制凭据泄漏后的影响. 审计日志默认记录标识、版本、决策与必要摘要，对敏感全文的临时取证另行控制访问和保留期. 没有必要把完整提示和工具响应无差别复制到监控平台.

10.6.3 训练记忆、成员推断及输出泄漏

隐私泄漏需要按信息来源分别分析. 推理时泄漏来自当前请求、检索结果、工具返回或历史记忆；训练时泄漏则可能来自权重对样本的记忆. Carlini 等人在 GPT-2 上通过查询恢复了部分训练文本，包括个人可识别信息 [71]. 这是特定模型与实验条件下的证据，不能据此断言任意模型都能恢复任意人的记录.

成员推断（Membership Inference）判断某条记录是否参加过训练；训练数据提取尝试恢复记录内容. 两者影响不同，例如仅暴露某人属于一个敏感群体的数据集，也可能造成隐私损害. 验证必须分别报告成员判断的误报与漏报、原文恢复的匹配标准，以及攻击者预先掌握的信息；模型生成一条看似真实的个人记录并不证明它来自训练集.

防护应贯穿数据生命周期：入库前排除非必要秘密、核实用途并控制重复；训练与微调时限制敏感样本暴露，保留删除和重训路径；推理时先授权检索，再最小化进入上下文的字段；输出时按接收者权限检查答案、引用、附件与工具参数. **数据防泄**

漏（Data Loss Prevention，DLP）通过数据识别与披露策略限制敏感信息外发．规则匹配、实体识别和上下文分类可以互补，但须覆盖改写、跨轮拼接与间接标识，不能仅检查身份证号等固定格式．

流式输出还存在不可收回的披露窗口：先发送全文再审核，无法撤销已经到达客户端的内容．高敏感字段应在释放前检查；需要跨句判断时可缓冲一个可判定的片段，必要时审核完整回答，并明确增加的延迟．检查范围包括可读思考摘要和中间工具结果．故障时应返回明确状态或限定回答，避免审核服务不可用自动变成无限放行．

例如员工只能查询自己的报销记录，系统应由会话身份绑定员工编号，在数据层返回获准字段，随后计算金额．输出审核检查其他员工姓名与账户信息，日志只记录必要标识和结果摘要．仅在最终答案遮盖姓名仍不足以保护已发送给外部模型、重排器或日志平台的完整记录．供应商是否用请求训练、保存多久、哪些功能保留状态，是三个需要分别核对的条件；私有部署同样需要内部访问和日志控制．

10.6.4　差分隐私边界

差分隐私（Differential Privacy，DP）约束单个受保护单位对随机机制输出分布的影响．令相邻数据集 D, D' 相差一个约定单位，$\mathcal{M}$ 为训练发布机制，S 为任意可测输出集合，则 (ε, δ)-差分隐私要求

$$\Pr[\mathcal{M}(D) \in S] \leqslant e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] + \delta, \qquad \varepsilon \geqslant 0, \quad 0 \leqslant \delta < 1. \tag{10.7}$$

相邻单位可以是记录，也可以是一个人的全部记录，二者保证不同．DP-SGD 通过逐样本梯度裁剪、聚合时加噪声和累计隐私核算实现训练保护 [72]．仅做普通梯度裁剪或在答案中任意加噪声，不能宣称满足上述定义．

采用该方法需要报告相邻关系、采样方式、噪声与训练步数，以及最终隐私预算；额外的私有数据调参和发布也要纳入核算．它带来效用和计算成本取舍，不能修复推理时直接把未授权文档放入上下文的路径，也不保证输出的所有事实都不敏感．训练隐私与服务授权应分别验证．

10.7　模型提取及未经授权的蒸馏

10.7.1　保护资产及可观察信息

模型提取（Model Extraction）指通过访问目标模型等方式构建行为近似的替代模型．知识蒸馏本身是正常训练方法，其机制见第三十四章《模型优化》；本节讨论超出授权范围的数据采集和能力复制．应分别确定允许的调用、批量生成和训练用途，不能仅凭用户要求解释步骤就判定滥用．

对手可能取得权重，也可能只能通过 API 观察文本．权重一旦交付到对手控制的环境，服务端限流和输出策略就无法约束其本地执行；开放权重的发布决策须接受

这一控制边界. API 服务可以隐藏权重、中间表示与完整词元分布，并控制采集规模，但只要持续提供有用答案，观察者就可能利用答案训练其他模型. 防护目标应表述为限制暴露、提高采集成本和发现未经授权的行为，而非保证输出无法被学习.

设请求为 x，教师生成的推理文本为 z，答案为 y，学生参数为 ϕ. 在已取得且获准使用的数据上，答案监督和轨迹监督可分别写成

$$\mathcal{L}_{\text{answer}} = -\mathbb{E}_{(x,y)} \log p_\phi(y \mid x), \tag{10.8}$$

$$\mathcal{L}_{\text{trace}} = -\mathbb{E}_{(x,z,y)} \log p_\phi(z, y \mid x). \tag{10.9}$$

期望针对相应训练样本分布；序列概率按词元自回归分解. 第二个目标增加了中间步骤监督，但轨迹错误、冗长或与学生能力不匹配时也可能降低效果. 隐藏 z 可以减少直接可用的轨迹，仍留下答案监督和由答案另行构造解释的途径；这种事后解释也不能被当作教师实际生成过的内部轨迹.

10.7.2 请求安全监测

单条请求内容合规，不代表长期采集用途获得授权. 运行控制可结合可信账号归属、组织级预算、并发与批量限制，以及跨请求的任务集中度、模板重复、调用节奏和账号关联. 聚合分析必须有明确用途、最小必要字段和保留期限；共享出口地址或相似课程作业都可能形成正常聚集，单一关联信号不足以定性.

Anthropic 在 2026 年 2 月 23 日的公告中介绍了流量分类、行为指纹、跨账号协同检测和强化账号验证，并提到识别为构建训练数据而诱导思维链的行为 [73]. 这些是厂商公布的防护方向，公告没有提供足以独立复现的完整判定规则或统一防护成功率. 该证据也不能支持把某种签名算法描述成其全部反蒸馏机制.

工程上可按证据强度逐级限制额度、暂停批量能力或进入人工核查，并保留申诉和恢复路径. 验收既要统计受控采集场景中的可获得样本量、检测延迟和漏检，也要检查正常研究、教学与企业批处理的误限制. 用于行为检测的数据不能原样复制到学生训练或其他无关用途. 故意向正常用户提供错误答案会损害产品可信度，不宜作为通用防护措施.

10.8 思维链保护及思考签名

10.8.1 明文、摘要及受保护状态

思维链（Chain of Thought，CoT）是模型生成的中间推理文本，并非神经网络全部内部计算的记录. 安全设计应区分完整推理文本、面向用户的解释或摘要，以及供后续推理恢复的受保护状态. 完整文本可能复述上下文中的敏感内容，也可能成为蒸馏监督；摘要减少直接披露，却仍需按数据权限审核.

用户需要核验结论时，可以提供依据、计算步骤、来源与不确定性，而不必发布全部内部轨迹. 解释质量应以可验证性评估，不能因一段文字语气连贯，就把它当作模型因果过程的忠实记录. 隐去原始轨迹也不等于消除模型从最终答案泄露秘密的可能.

10.8.2　Anthropic 的 signature 字段

按 2026 年 9 月 11 日核查的 Claude Messages API 官方 Thinking 文档，`thinking` 块中的 `signature` 是承载加密完整思考内容的不透明字段，API 在回传时用它验证思考块来自 Claude. `display` 设为 `summarized` 时返回可读摘要，设为 `omitted` 时可读字段为空，受保护内容仍用于续接；具体支持情况随模型而异 [74].

集成时应按协议原样保存和回传需要的思考块，包括 `redacted_thinking` 块；流式签名通过 `signature_delta` 到达. 加密字段不应被解析为可读思维链或当作普通答案展示. 上述文档给出接口行为，并未公开足以据此指定其密码算法、密钥派生和全部绑定范围的细节，因此不能将字段名直接解释为某种可由第三方公钥验证的标准数字签名 [74].

从控制作用看，加密限制无解密能力的观察者读取完整轨迹；来源与完整性验证限制伪造或篡改后冒充有效思考状态. **对已公开明文添加签名，并不能阻止别人复制该明文作为训练数据.** 即使完整 CoT 受保护，可读摘要、答案和工具轨迹仍可能提供监督；声称反蒸馏有效还需要相应采集与训练评估.

表 10.2: 思考保护与反蒸馏控制的作用边界

控制	直接作用	仍需单独处理的风险
签名或认证校验	核验来源与完整性	可读文本被复制、内容本身错误.
加密保存与回传	限制无密钥者读取状态	解密端权限、明文输出与日志泄漏.
摘要或省略展示	减少公开轨迹	摘要与答案的敏感信息及训练价值.
配额与行为检测	限制并识别规模化采集	误限制、分散采集及正常用途区分.

10.8.3　应用侧的状态隔离及验收

对于自建系统，若客户端只需恢复会话，可在服务端保存推理状态并返回受限句柄；需要客户端携带状态时，应使用经过审查的认证加密方案，同时在受保护上下文中绑定租户、会话、用途、版本与有效期，并设计重放处理. 这里给出的是应用设计要求，不是对 Anthropic 内部实现的推断. 密文也应按敏感状态控制访问、日志与保留，不能仅因不可读就任意跨租户共享.

验收应分为三个独立目标：用获准的合成敏感标记检查答案、摘要、工具参数与日志的泄漏；用篡改、跨会话替换、过期和重放检查状态协议；用受控批量请求检查反采集策略及正常任务误拦. 伪造状态被拒绝只能证明相应校验路径有效，不能推出

隐私或反蒸馏整体通过. 合成标记测试属于泄漏路径检查，也不能单独构成训练集隐私保证.

10.9 安全评估及发布控制

10.9.1 漏放、误拦及基率

将安全过滤看作二分类任务，危险请求被放行为**漏放**（False Negative），正常请求被阻止为**误拦**（False Positive）. 阈值通常在二者之间权衡. 设危险请求基率为 π，危险漏放概率为 q，正常误拦概率为 f，则在所有拦截中危险请求所占比例为

$$\Pr(\text{danger} \mid \text{blocked}) = \frac{\pi(1-q)}{\pi(1-q) + (1-\pi)f}. \tag{10.10}$$

例如构造总体含 10000 条请求，其中 100 条危险；过滤器拦截 90 条危险请求、误拦 198 条正常请求. 危险召回率为 90%，正常误拦率为 2%，但 288 条拦截中仅 90 条危险，精确率为 31.25%. 高拦截召回并不表示审核队列大多是真风险.

若放行危险请求的代价为 C_h、误拦代价为 C_b，且有校准的后验危险概率 p，放行与拦截的期望成本分别为 pC_h 和 $(1-p)C_b$. 拦截条件为 $p > \dfrac{C_b}{C_h + C_b}$. 这只是两动作、固定成本模型；涉及不可接受后果、人工复核和部分能力限制时，需要更完整决策规则. 分类器原始分数不应未经校准直接作为 p.

10.9.2 加权指标及零失败的置信界

对 N 个样本，给定允许行为集合 $\mathcal{A}_i$ 与非负权重 w_i，可定义

$$\widehat{v} = \frac{\sum_i w_i \mathbf{1}[\widehat{y}_i \notin \mathcal{A}_i]}{\sum_i w_i}, \qquad \sum_i w_i > 0. \tag{10.11}$$

权重需预先规定，结果应同时呈现逐类失败数、分母和高严重度个案. 没有正常样本的切片，其误拦率是不适用，不是零. 还应测量要求执行控制的机会中实际执行了控制的比例，避免只评估过滤器本身而遗漏绕行路径.

若 N 次独立同分布试验中失败数为 0，真实失败率为 p，观测零失败的概率为 $(1-p)^N$. 单侧置信度 $1-\alpha$ 的精确上界满足

$$(1 - p_{\text{upper}})^N = \alpha \quad \Longrightarrow \quad p_{\text{upper}} = 1 - \alpha^{\frac{1}{N}}. \tag{10.12}$$

当 $\alpha = 0.05, N = 100$，上界约 2.95%；当 $N = 300$，上界约 0.994%. 所以 $\dfrac{0}{100}$ 不能证明失败概率为零. 利用 $e^x \approx 1 + x$，大样本时上界近似 $-\dfrac{\log \alpha}{N}$，95% 置信度下约为

$\frac{3}{N}$. [2]

10.9.3　组件、链路及未知路径

评测应从规则和参数校验，到模型单轮多轮行为，再到身份、检索、输出和工具的完整链路. **红队评估**（Red Teaming）在授权范围内主动寻找未知失效路径，结果应记录前提、真实影响和绕过的控制. 仅使模型输出不合适文字与真正跨租户读取是不同后果，不能合并成一个攻击成功标签.

样本需要覆盖正常易误拒请求、边界案例、不同语言与模态、长上下文以及历史事件. 随机生成设置和版本应固定并报告重复次数. 人工审核者也需明确判据、访问分级和暴露保护；模型评审器自身可能受注入或分布变化影响，应通过人工样本估计误差.

发布证据应绑定变化范围、风险登记、制品摘要、切片结果、权限验证和恢复能力. 高严重度后果可形成独立门槛，不能被大量正常样本平均掉. 受控放量应限制流量、租户、工具和资源范围；观察窗口要考虑事件基率与统计不确定性，不存在对所有场景都合适的固定百分比.

10.10　监测及事件响应

10.10.1　异常调查记录

监测需关联请求、租户、模型、策略、检索快照、工具动作和结果状态. 拒答率突变可能来自攻击，也可能来自模型更新或数据变化；外部写入增加可能是业务高峰，也可能是控制绕过. 告警规则应结合变更记录与分组基线，避免把单个指标波动直接等同于安全事件.

资源也是安全资产. 限制输入解码大小、上下文长度、检索分支数、工具轮数、并发、输出和总截止时间，可以约束资源耗尽. 取消和失败需要传播到后台任务，避免前端已结束而工具循环仍在运行. 对调用量异常的租户应按权限与配额处理，不能依靠模型自觉少调用.

10.10.2　遏制、保全、修复及恢复

事件响应（Incident Response）是识别、遏制、调查和恢复受影响系统的组织过程. NIST SP 800-61 Rev.3（2025 年）将响应准备与检测、响应、恢复放入网络安全风险管理中 [75]. 对模型系统，应预先具备停用工具、隔离索引、撤销凭据、切换制品与保护证据的能力，而不是事件发生后才临时设计开关.

[2] 这里的置信区间是重复抽样覆盖性质，不是对固定参数赋予后验概率. 自适应红队选择、相关变体和持续调阈值通常不满足独立同分布前提，不能机械套用该上界作为对未知攻击的保证.

发现疑似泄露后，先限制继续扩大的路径，例如停用外发工具和失效凭据；同时保留受控证据，包括请求时间线、来源版本、策略决策与执行账本. 调查要区分已提出、已拒绝、已执行和执行结果未知的动作. 没有发现成功日志不等于没有执行，需结合工具端记录确认.

影响分析应覆盖用户、租户、数据范围和副作用，并判断缓存、记忆与派生制品是否受污染. 修复位置可能在数据、模型、模板、策略或执行器. 只更新提示可能掩盖一条攻击文本，却保留错误授权；只回滚模型也不能撤销已经外发的信息或已执行的业务变更.

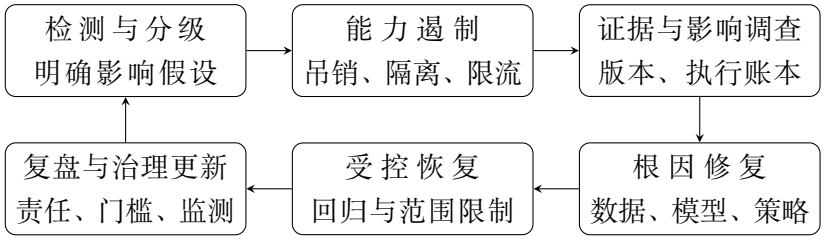


图 10.2: 事件处理闭环. 证据保全与遏制可并行开展，恢复以修复证据和受限能力为前提.

恢复前应验证受影响控制、历史失败路径和相邻变体，并检查凭据、缓存和索引的清理结果. 恢复可以分能力进行，例如先开放只读检索，继续关闭外部写入；不能因为聊天答案恢复正常就把全部工具一起恢复. 通知、保留与披露要求按事件和组织适用规则执行，记录由谁作出相应判断.

10.10.3 构造事件的时间线

回到内部报告外发的构造场景. 记录显示某请求在 10:00 提出外发建议，策略错误允许，工具在 10:01 执行成功，10:03 出现告警. 响应人员在 10:05 停用发送能力并撤销服务凭据，随后固定相关文档版本、动作摘要与收件范围. 检测延迟从已确认执行到告警为 2 分钟，遏制延迟从告警到停用为 2 分钟；指标必须明确起止定义.

调查确认根因是“合法结构即允许”的错误策略，而非只有某个提示词漏洞. 修复需加入任务范围、目的地址和数据级别校验，缩小工具身份权限，并处理可能保存敏感摘要的缓存. 已经送达的报告不能通过权重回滚收回，需进入实际数据事件处置. 随后以相同动作参数、不同表达、工具超时重试和缓存读取构造回归条件，分阶段恢复能力. 这个时间线是教学构造，不是对某真实系统事件的描述.

第二部分

应用智能体系统

第十一章　上下文学习

一个固定参数的语言模型，在收到不同指令、示例和背景信息时，会给出不同的条件预测．增加几个示例、要求分解任务，或生成多个候选再选择，都会改变求解过程；但这些变化分别作用于输入信息、生成路径和决策规则，不能统称为“提示更强”．理解它们的区别，才能解释质量提升来自哪里，以及是否值得额外计算．

本章讨论不更新模型参数的运行时策略．目标不是收集提示句式，而是建立一套可分析的过程：构造有边界的上下文，选择有限示例，生成候选，提取答案，再依据频率、验证证据和预算作出决策．提示方案同模型权重一样需要版本化，单道题的成功不能证明其总体有效．

11.1　固定模型上的条件学习

11.1.1　输入条件及参数更新

上下文学习（In-Context Learning，ICL）通过输入中的任务描述与示例改变模型的预测，当前请求不更新参数 [9]．记参数为 θ，当前输入为 x，有序示例序列为 $D=((x_1,y_1),\dots,(x_m,y_m))$，其预测写为

$$p_\theta(y \mid x, D, I, \mathcal{B}, O, T), \tag{11.1}$$

其中 I 为任务指令，$\mathcal{B}$ 为背景，O 为输出约束，T 为消息序列化与生成起始协议．即使自然语言文字不变，角色分隔符和模板改变也会改变真正进入模型的词元序列．

零样本提示（Zero-Shot Prompting）没有当前任务的示例对，但可以包含指令和背景．**少样本提示**（Few-Shot Prompting）额外提供少量示例；这里的“少”描述上下文内样本数，并不意味着基座在预训练或后训练中未见过相关任务．ICL 的表现可能来自对已学任务的识别、格式适应、信息提取或新映射的临时推断，不能仅凭成功回答就判断是哪种机制．

第二十一章《参数高效微调》中的连续提示训练需要更新任务变量，而本章的少样本文本示例本身不通过当前请求的梯度学习．检索得到的背景也不必是示例：背景提供事实，示例展示输入到输出的关系．两者可以同时存在，但其来源、正确性与评价方式应分开．

表 11.1: 本章主要符号与对象.

符号	含义
θ, x, D	固定模型参数、任务输入、有序示例
C	完整运行时上下文，包括模板与生成起点
z, a	中间步骤文本与最终答案
$q(a, z \mid C)$	实际解码策略产生的联合分布
$h(z, a)$	版本化答案解析与规范化函数
K, n_a	候选数量与答案 a 的票数
$v(C, z, a)$	验证器对候选的评分
L_C, L_i, B	上下文长度、候选输出长度、预算

11.1.2 上下文的四种职责

任务指令说明要执行的操作、判断标准与适用边界；示例具体呈现输入输出关系；背景提供任务所需信息；输出约束规定返回内容及可解析形式. 它们共同形成 C，但并不具有相同权威. 背景中的引用、检索结果与用户提供的待处理文本应作为数据解释，不能因为出现命令句式就被提升为控制指令.

明确边界的意义不仅是防止指令冲突，也在于减少统计歧义. 例如分类任务的标签应是固定集合，示例不能同时把同一标签用于不同含义；结构化输出应规定字段类型、单位和缺失值. 语法正确只说明结果可被解析，不证明它符合事实或满足业务约束.

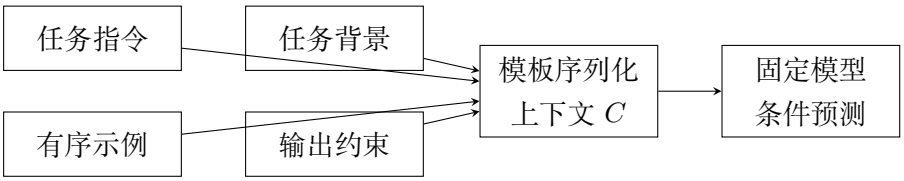


图 11.1: 不同上下文组件进入同一条件预测过程；当前请求不改变模型参数.

11.2 上下文样本配置

11.2.1 相关性、覆盖和预算

示例库（Example Bank）是可复用的示范资产，应记录来源、任务类型、答案、用途和版本. 候选首先满足访问权限与数据质量，再进行相关性排序. 把无权使用的信息放入模型后再要求其“不要泄露”，不能替代前置隔离.

示例相关性可以依据词汇重合、语义向量、任务结构或已知难度. 词汇近似并不保证解法类似：两题可能仅替换实体名称，却需要相反规则；反之，表面差异很大的题可能共享相同约束. 因此选择目标不能只有最近邻距离.

一种显式的设计目标是，在候选集合 $\mathcal{E}$ 中选择子集 S，最大化

$$F(S;x)=\sum_{e\in S} r(e,x)+\lambda\,\mathrm{Coverage}(S)-\mu\sum_{\{e,e'\}\subseteq S}\mathrm{Redundancy}(e,e'),\quad \sum_{e\in S}\ell_e\leqslant B_D. \tag{11.2}$$

r 衡量相关性，覆盖项表示不同必要技能或标签，冗余项惩罚重复，ℓ_e 为序列化后的示例长度. 它是用于明确取舍的设计形式，不是保证任务最优的通用算法. 若所有候选相关性均不足，可回退到更少示例或零样本，不应为达到固定数量硬塞无关样本.

例如预算 $B_D=200$ 个词元，三个示例长度为 $(120,100,80)$，相关性为 $(0.9,0.8,0.6)$. 仅按相关性从高到低且放不下就跳过，会选第一、第三条，总相关性 1.5；第二、第三条仅需 180 个词元、总相关性 1.4，若二者覆盖两个互补技能而第一、第三条重复，覆盖项可以使其更优. 这里没有已知测试答案参与选择，全部评分必须来自预先规定的信息.

11.2.2　上下文顺序效应

模型接收的是有序序列，所以一般没有

$$p_\theta(y\mid x,D)=p_\theta(y\mid x,\pi(D)) \tag{11.3}$$

对任意排列 π 都成立的保证. 位置编码、因果注意力与前文示例形成的局部模式使次序具有作用. 已有研究直接观察到少样本顺序敏感性 [76]；但某个模型上的最好排列未必能迁移到另一个模型或另一个任务.

输入长度也使“相同示例”不必等价. 若输出预算固定，增加示例可能挤掉必要背景；若上下文被截断，实际进入模型的可能只剩一部分示例或残缺答案. 截断后的词元清单才是真正的条件，不能仅记录原始示例编号. [1]

11.2.3　标签分布及内容先验

示例同时展示格式、输入分布、标签空间和具体映射. 针对部分分类与多项选择任务的研究表明，这些因素可以分别影响 ICL 结果 [77]；这并不允许在一般任务中把错误标注当作无害. 若任务需要通过示例学习陌生符号映射，错误标签可能直接破坏映射识别.

标签词（Verbalizer）把类别对应到模型输出文本. 不同标签词的预训练频率、词元长度和上下文位置会引入偏好. 若标签含多个词元，应该比较完整标签序列的概率及必要结束边界，不能只比较首词元概率. 用平均词元对数概率替代序列概率又会改变决策目标，需明确声明.

[1] 示例前缀可被缓存以降低重复预填充成本，但缓存改变的是执行复用，不会消除示例在上下文窗口中占用的位置.

一种**上下文校准**（Contextual Calibration）方法，用无任务内容的输入估计标签偏好 $b_a > 0$，再对目标输入概率作比值校正 [78]：

$$\widetilde{p}(a \mid x, C) = \frac{\frac{p_\theta(a|x,C)}{b_a}}{\sum_{a'} \frac{p_\theta(a'|x,C)}{b_{a'}}}. \tag{11.4}$$

例如两标签的原概率为 $(0.6, 0.4)$、偏好估计为 $(0.8, 0.2)$，比值为 $(0.75, 2)$，归一化后为 $(\frac{3}{11}, \frac{8}{11})$. 排序发生反转，但这不证明校正后的答案正确. 无内容输入是否能代表应扣除的偏好、是否需要平滑，以及真实类别先验是否本来就不均衡，都会影响适用性. 校准系数应在开发数据上确定.

算法 11.1　预算内的示例构造

输入：问题、版本化示例库、选择策略、上下文上限与保留输出长度. **输出：**实际序列化上下文及示例清单. **状态：**可用词元预算、已选集合、覆盖状态.

1. 过滤不符合权限、任务与质量条件的候选；排除与保留评估样本不允许的家族重叠.
2. 扣除指令、问题、必要背景、模板和输出预留，得到示例预算；预算不足时采用明确的缩减或失败策略.
3. 按预定相关性、覆盖和冗余准则选择可容纳示例，采用固定平局规则. 达到预算或无合格候选时停止.
4. 按已声明顺序构造完整消息并实际分词；如长度变化导致超限，整条移除最低优先级示例后重新构造，不截断到示例答案中间.

不变量：最终上下文长度符合预算，示例来源与顺序可追溯，选择过程没有使用当前测试题的标准答案.

11.3　分步生成的概率结构

11.3.1　中间文本提供新的条件位置

思维链提示（Chain-of-Thought Prompting，CoT Prompting）通过步骤示范或任务分解要求，引导模型先生成中间文本再形成答案 [79]. 令中间文本为 z、答案为 a，在同一生成协议下有

$$p_\theta(z, a \mid C) = p_\theta(z \mid C) p_\theta(a \mid C, z), \qquad p_\theta(a \mid C) = \sum_z p_\theta(z, a \mid C). \tag{11.5}$$

中间词元成为后续位置的条件，能够保存局部结果、展开约束和组织计算. 但是这些文本可能包含错误、遗漏或事后合理化，不能视为神经计算过程的完整因果记录.

式 (11.5) 描述允许生成 z 的协议. 要求直接回答可能改变提示和输出语法，其答案分布不必等于先生成步骤再边缘化的分布. 用两个不同提示的输出差异论证“只是多了相同计算的可见解释”，没有概率依据.

单轨迹生成只访问一个候选 z，不等于计算求和. 分步文字更长也不保证有效计算更多：重复前提、同义复述或复制错误会耗费预算却不改善答案. 对于可外部执行的算术、程序和约束，应把验证对象写成明确输入、操作与结果，不能仅以叙述自信程度判断正确性.

11.3.2　路径最大概率及答案最大概率

考虑一个构造分布：四条完整路径的概率分别为 $0.30, 0.25, 0.25, 0.20$. 第一条给出答案 B，第二、第三条给出答案 A，第四条给出答案 C. 最大概率路径给出 B，但答案边缘概率为 $p(A) = 0.50, p(B) = 0.30, p(C) = 0.20$，最大概率答案是 A.

这说明“寻找最可能的一条解答”与“汇总所有产生同一答案的路径”是不同目标. 即便能够精确找到最大概率完整路径，也不保证找到最大概率答案；逐词贪心还不保证找到最大概率完整路径，相关搜索边界见第八章《自回归语言建模》.

11.4　自一致性及多数投票

11.4.1　采样分布估计

自一致性（Self-Consistency）生成多条候选路径，提取最终答案并按出现频率聚合 [80]. 令真实解码分布为 $q(z, a \mid C)$，答案解析函数为 h，则 K 次采样的计数为

$$n_b = \sum_{i=1}^{K} \mathbf{1}[h(z_i, a_i) = b], \qquad \widehat{q}_K(b) = \frac{n_b}{K}. \tag{11.6}$$

在独立同分布采样下，$\mathbb{E}[\widehat{q}_K(b)] = q_h(b)$，方差为 $\dfrac{q_h(b)(1 - q_h(b))}{K}$，其中 q_h 是经过解析映射后的答案分布. 温度、top-k、top-p 或停止规则改变 q，因此投票一般估计的是解码后的分布，而非未经变换的 p_θ.

如果重复执行确定性贪心，通常只得到同一个答案，候选数量并未提供新的采样信息. 反之，同一个固定模型使用独立随机数，可以在给定问题与提示的条件下产生独立样本；“同一模型”本身并不等同于统计不独立. 跨题的共同偏差、候选复用同一已生成前缀、相互参考修改或共享随机状态，则会改变应采用的独立性假设.

二元多数正确率

固定一题，候选的正确性是独立同分布 Bernoulli 变量，单次正确率为 p. 候选数 K 为奇数，决策仅区分正确和错误两类；不存在解析失败、拒答或多种错误标签竞争.

满足上述假设时，多数正确概率为

$$P_K = \sum_{j=\frac{K+1}{2}}^{K} \binom{K}{j} p^j (1-p)^{K-j}. \tag{11.7}$$

$p = 0.6, K = 5$ 时，$P_5 = 10(0.6)^3(0.4)^2 + 5(0.6)^4(0.4) + (0.6)^5 = 0.68256$. $p > \frac{1}{2}$ 是这一二元改善机制的关键；$p < \frac{1}{2}$ 时，增加候选反而可能更稳定地选择错误.

真实答案空间一般不是二元. 若正确答案概率 0.4，其他答案分别为 0.35 和 0.25，正确答案虽然不足半数，却是唯一众数；独立大量采样的最高票可能收敛到正确答案. 若一个错误答案拥有最大概率，则自一致性趋向于强化它. 所有正确性结论都需要联系真实答案分布，而不是把二元公式直接套到开放生成.

11.4.2 相关性、平票及解析失败

若正确性指示变量具有相同边际概率 p、任意两个的相关系数为 ρ，则平均正确率方差为

$$\mathrm{Var}(\bar{C}) = \frac{p(1-p)}{K}[1 + (K-1)\rho]. \tag{11.8}$$

该式只在所声明的等相关结构下成立，而且没有给出相关投票的完整尾概率；不能仅知道 ρ 就继续使用式 (11.7). 候选共享一种错误解释时，经验收益可能很早饱和.

答案规范化（Answer Canonicalization）把单位、数值格式和等价表达映射到任务定义的同一答案. 例如在要求以米给出长度的题中，1.5 米与 150 厘米可经单位换算合并；不带单位的 150 不能无条件按同一答案处理. 大小写、空白、小数容差与集合顺序是否可忽略，取决于任务本身.

解析失败（Parse Failure）应作为显式状态，而不是从分母中消失. 若把失败结果记为 $\perp$，则所有计数之和仍为 K. 可以只在有效答案中选择，但必须同时报告有效覆盖率 $1 - \frac{n_\perp}{K}$、总候选上的支持率和有效候选中的支持率. [2]

奇数候选并不排除多类别平票. 五个候选可以形成 $2:2:1$. 预先定义的平票处理、最低有效数与最低票差可以触发继续采样、验证或拒答；不能事后选择更符合标准答案的候选. 票数领先是稳定性线索，不是经校准的正确概率.

[2]把无效输出排除后计算的正确率是条件于“输出有效”的指标；它可以用于诊断解析器，但不能代替覆盖全部请求的端到端正确率.

11.5　候选验证及覆盖率

11.5.1　投票排序及验证排序

验证器（Verifier）检查候选的某些性质或为其正确性评分. **多候选择优**（Best-of-K Selection，Best-of-K）可写为

$$i^{*}=\arg\max_{1\leqslant i\leqslant K} v(C,z_i,a_i),\qquad \widehat{a}=h(z_{i^{*}},a_{i^{*}}). \tag{11.9}$$

这与票数选择不同：罕见答案可能通过严格验证，而最高票答案可能违反约束. 训练验证器从多个数学候选中选择，是一条已有研究路线 [81]；验证器本身如何训练与评价属于另一个问题，不能把它当成完美判定器.

若每次独立生成正确候选的概率为 p，则至少有一个正确候选的概率为

$$P(\text{覆盖正确答案})=1-(1-p)^{K}. \tag{11.10}$$

$p=0.6, K=5$ 时为 0.98976，远高于前面的多数正确率，但这是完美选择器能够达到的候选覆盖，不是实际系统正确率. 如果正确候选已经出现，验证器仍可能选错. 扩大 K 还可能引入更多能欺骗评分器的高分错误，因而固定验证器的选择质量未必单调提高.

11.5.2　验证器适用范围

形式验证、程序执行、量纲检查、数据库约束和外部证据分别检查不同性质. 一个算术表达式计算正确，不证明表达式正确建模了原题；程序通过有限测试，不证明所有输入正确；引用存在，不证明引用支持当前主张. 验证报告应说明验证对象、方法与覆盖范围.

模型自评与独立工具验证也不同. 生成器再读自己的答案可能发现局部不一致，也可能重复同一误解. 若验证器与生成器共享训练偏差，表面独立的两个调用并不提供逻辑上的独立证据. **拒答**（Abstention）应是允许的决策状态，用于共识不足、验证条件不满足或信息缺失，而不是强制返回最高分候选.

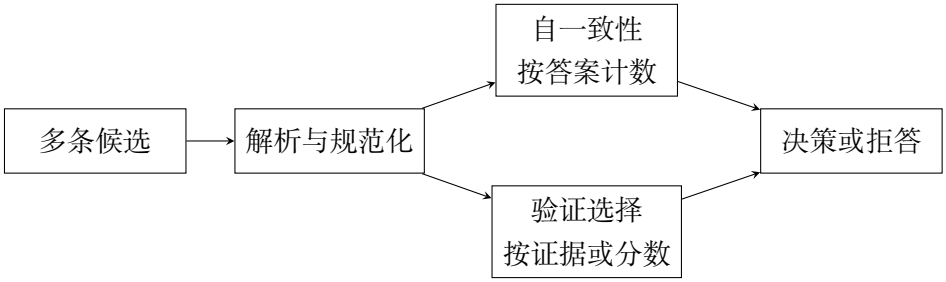


图 11.2: 聚合与验证是不同决策分支，可以组合，但应分别记录各自依据.

11.6 预算及自适应停止

11.6.1 推理总成本

测试时计算（Test-Time Compute）包括上下文处理、候选生成、验证和工具调用. 若每个候选输入长 L_C、输出长 L_i，不共享前缀时的逻辑词元量为

$$N_{\text{in}} = KL_C, \qquad N_{\text{out}} = \sum_{i=1}^{K} L_i. \tag{11.11}$$

可用 $C_{\text{cost}} = c_{\text{in}}N_{\text{in}} + c_{\text{out}}N_{\text{out}} + C_v + C_t$ 描述给定口径的成本，其中 C_v, C_t 为验证与工具成本. 输入输出单价或等价权重可能不同；不能把词元总数直接当作实际费用.

共享提示前缀可以降低预填充工作，但每条分支的生成状态仍需维护. 并行生成可能降低墙钟延迟，却增加并发内存并不消除总计算. 每个请求还需要单独满足上下文约束 $L_C + L_{i,\max} \leqslant L_{\max}$. 因此预算至少包含候选上限、输出上限、总计算或费用、时间与工具次数.

增加示例数 m 与增加候选数 K 争用资源. 更有用的示例可能提高单候选质量，但较长输入会使每条候选更昂贵；更多采样只有在生成分布包含合适答案且选择器有效时才有意义. **边际收益**（Marginal Utility）可用开发集上追加一次计算带来的质量改善与成本比较，而非假设候选越多越好.

11.6.2 固定预算下的不可逆领先

若最多还允许生成 R 个候选，当前第一、第二名票数分别为 n_1, n_2，且 $n_1 > n_2 + R$，则剩余候选即使全部投给竞争答案也无法改变唯一第一名. 这个条件可以在固定总候选上限下提前结束投票，保持最终最高票决策不变.

它不证明领先答案正确，也不适用于候选生成后还会修改旧票、改变答案规范化或按验证器重新加权的过程. 若采用概率置信阈值提前停止，重复查看样本再使用固定样本置信区间可能破坏覆盖率；应采用适用于序贯决策的方法，或把停止规则作为完整策略在独立数据上评价.

算法 11.2 有预算的候选聚合与验证

输入： 固定上下文、解码策略、解析器、候选与词元上限、最低有效数、平票及验证规则. **输出：** 答案、拒答或预算终止状态. **状态：** 答案计数、失败计数、已用词元、验证记录.

1. 初始化计数. 每次开始生成前，为候选最大输出和可能的验证预留预算；不足时停止追加.
2. 独立采样候选并记录实际停止原因；按固定解析器转为规范答案或失败状态，不静默丢弃失败.

3. 更新票数与预算．若满足预定的验证接受条件或固定上限下不可逆领先条件，进入最终决策．

4. 达到候选上限、超时或预算上限时停止；对平票、有效数不足和未通过验证的状态执行预定升级或拒答策略．

不变量：每个候选都有终态，全部成本被计入，解析与停止规则不根据当前标准答案临时修改．验证接受条件所证明的范围应随结果保留．

设任务要求计算商品总价：每件 30 元、购买 6 件、优惠 30 元．明确运算为 $30 \times 6 - 30 = 150$ 元．该运算用作本算例的外部判定依据，而不是由投票自动获得的真值．

假设构造上下文含固定指令、问题与模板 180 个词元，两个示例分别为 90 与 110 个词元，因此 $L_C = 380$．取 $K = 5$，输出长度依次为 $(70, 85, 60, 100, 75)$，合计 390．按不复用输入、输入输出等权的教学计量，候选成本为 $5 \times 380 + 390 = 2290$ 个词元单位．若一个批量验证调用再耗费 120 个单位，总量为 2410．这些是给定长度的预算计算，不是实际模型生成测量．

表 11.2: 构造的五候选记录，只列最终结果和验证状态．

候选	解析前结果	规范答案	约束与算术验证
1	150 元	150 元	通过
2	人民币 150.00	150 元	通过
3	180 元	180 元	未扣优惠，失败
4	最终结果缺失	$\perp$	无法验证
5	120 元	120 元	扣减与题意不符，失败

表 11.2 中有效候选为四条，解析覆盖率为 $\frac{4}{5} = 80\%$．答案 150 元得两票，整体支持率 $\frac{2}{5} = 40\%$，在有效候选中的支持率为 $\frac{2}{4} = 50\%$；第二名各一票，票差为一．它是唯一最高票，但不满足严格过半数．若策略预先规定至少四条有效候选且需严格过半数，本次投票应转入验证，而非事后放宽阈值．

验证器检查原题运算和单位后，两个等价的 150 元候选通过，其他有效候选失败，可以返回该答案及简洁计算依据．若验证只检查表达式执行，则 “$30 \times 6 = 180$” 也能通过算术执行，仍需检查优惠约束；这说明验证定义直接决定选择结果．

若总预算只有 2300 个单位，当前五候选虽然可以生成，却无法完成预定的 120 单位验证．正确策略应在生成前预留验证成本，或者采用不同候选上限并按其规则决策．不能先耗尽预算，再把未经验证的最高票结果标作已验证．

11.7 实验迁移性

11.7.1 提示变量隔离

比较零样本、少样本、分步生成和多候选策略时，应固定模型版本、题目、模板、答案规范化与评分规则. 若目标是比较策略在自然成本下的效果，可以允许成本不同但同时报告；若目标是比较预算效率，则应在相同资源上限内比较. 两种研究问题都合理，但不能混用结论.

使用同一题目作成对比较有助于减少题目难度差异. 设两个策略在 N 题上的正确指示为 c_i^A, c_i^B，差异估计为

$$\widehat{\Delta} = \frac{1}{N}\sum_i (c_i^A - c_i^B). \tag{11.12}$$

应同时记录 A 对而 B 错、B 对而 A 错的题目，而不只比较两个总百分比. 采样策略还需要预先规定种子或重复方式；同一问题多个候选不是多个独立测试问题，不能据此虚增评估样本量.

数据应覆盖直接问答、组合运算、约束满足、领域知识与不可回答问题，并按语言、长度和难度报告分子与分母. 解析失败、截断、拒答与错误答案是不同状态. 只在成功解析的样本上报告准确率，会隐藏格式成本与覆盖损失.

11.7.2 迁移条件校准

一个提示在某模型上有效，可能依赖其后训练习惯、标签偏好和模板. 更换模型、分词器、量化版本、示例顺序或解码参数后，条件分布已经变化. 不能因为文字提示相同，就认为实验条件完全保留. 单个种子或单道题得到的改善更不能支持跨模型结论.

提示开发中反复查看测试题并调整示例，会把测试信息编码到运行时资产. 应保留独立开发集，冻结示例库、解析器与停止规则后再评估. 涉及近重复与问题家族的划分，见第十八章《训练数据工程基础》. 安全或权限边界仍由运行时控制，ICL 示例不应成为改变服务端权限的通道.

三个不同的证据问题

分步生成回答“如何组织一条候选”；自一致性回答“哪些答案在采样中出现得更多”；验证选择回答“哪些候选满足指定检查”. 三者可以组合，但只有在各自假设、预算和判定范围都明确时，额外计算才具有可解释的收益.

第十二章　信息检索

检索系统需要在规模庞大且不断变化的语料中，找到与当前信息需求有关、可访问并且版本有效的材料．词语是否相同、向量是否接近、排名是否靠前，分别只是相关性的不同近似．系统必须同时回答两个问题：怎样表示并排序候选，以及怎样在有限时间和内存内找到它们．

信息检索（Information Retrieval，IR）输出的是有序材料集合．本章讨论词法评分、表示学习、近似索引与检索评估；下游如何将材料组织为生成上下文、如何核对回答与引用，属于检索增强生成的架构问题．高召回可以提供必要材料，却不自动证明答案正确．

12.1　检索对象及语料结构

12.1.1　文档、片段及相关性

令查询为 q，检索对象集合为 $\mathcal{D}$，相关性等级为 $r(q,d)$．这里的对象 d 可以是整篇文档、段落或表格片段，必须与标注和评估一致．一个系统用片段排序、另一个系统按文档统计命中时，直接比较数字可能掩盖重复片段占据前列的问题．

语义切分（Semantic Chunking）依据标题、段落、句子、代码或表格结构形成边界，再结合长度限制组织片段．较短片段有利于定位，但可能丢失否定条件、表头和适用范围；较长片段提供上下文，却可能稀释主题并增加编码成本．重叠窗口减少边界遗漏，也增加重复和索引体积，因此应同时计量召回与重复率．

解析阶段应保留标题层级、页码或源偏移、版本、生效时间、语言与访问条件．表格单元脱离表头常失去意义，扫描文档的文字识别错误又会影响精确词匹配．语料质量问题不能一律通过更复杂的编码器补救．原始文档、派生片段与索引条目之间的血缘关系，遵循第十八章《训练数据工程基础》所述原则．

12.1.2　词法分析及匹配对象

词项（Term）是检索分析器生成的匹配单位，不一定等于语言模型词元．中文分词、大小写、字符规范化、数字编号、同义词与停用词规则都会改变词项集合．查询与文档应使用兼容的分析规则；变更规则一般需要重新构建相应索引．

表 12.1: 主要符号与形状.

符号	含义
$N, n_t, f(t,d)$	检索对象总数、包含词项 t 的对象数、对象内词频
$\lvert d\rvert, \bar{\ell}$	对象的分析器词项长度与平均长度
k_1, b	BM25 饱和参数与长度归一化参数
$u, v_d \in \mathbb{R}^h$	查询与文档的 h 维表示
$\tau, s(q,d)$	对比学习温度与检索分数
$K, R_q^{(K)}, G_q$	截断长度、前 K 个结果、相关对象集合
M, m, c	图邻接预算、PQ 子空间数、RRF 平滑常数

过度规范化可能破坏型号、错误码和带连字符的标识. 对精确编号和普通自然语言，可以使用不同字段及匹配策略，而非强迫所有内容共享同一种切分. 字段权重也应作为评分配置，不能在解释分数时忽略.

12.2 倒排索引及概率排序

12.2.1 词项候选定位

倒排索引(Inverted Index)为每个词项保存包含它的对象列表,即**倒排列表**(Posting List). 每个条目可含文档标识、词频、位置和字段信息. 查询不再扫描全部文本，而是读取命中词项的列表，合并候选并计算分数. 词项位置可支持短语与邻近关系，只有文档标识和词频则不能直接恢复这些约束.

设查询词项集合为 T_q，若完整遍历相关列表，处理量与 $\sum_{t\in T_q} |P_t|$ 有关；常见高频词会产生长列表. 跳跃结构和基于分数上界的剪枝可以减少计算，但必须使用有效上界，才能保持指定评分下的精确前 K 名. 压缩列表减少内存与读取量，也增加解码代价.

12.2.2 概率动机及独立性近似

概率排序思想是在给定查询和固定损失条件下，优先返回相关概率更高的对象. 为说明稀有词为何重要，考虑二元相关变量 R 和词项出现指标 $x_t \in \{0,1\}$. 假设给定 R 后词项条件独立，记 $p_t = P(x_t = 1 \mid R = 1)$、$u_t = P(x_t = 1 \mid R = 0)$，则贝叶斯对数比为

$$\log \frac{P(R=1 \mid x)}{P(R=0 \mid x)} = \text{常数} + \sum_t \left[x_t \log \frac{p_t}{u_t} + (1-x_t) \log \frac{1-p_t}{1-u_t} \right] \tag{12.1}$$

$$= \text{常数} + \sum_t x_t \log \frac{p_t(1-u_t)}{u_t(1-p_t)}. \tag{12.2}$$

忽略与候选无关的常数后，词项提供可加的对数证据．缺少相关性标注时，对 p_t, u_t 作近似并使用文档频率估计，会得到与逆文档频率相关的权重．真实词项并不独立，此模型是排序近似；BM25 的词频与长度处理还引入进一步建模，不能只用上述二元公式一步推出全部 BM25[82].

12.2.3　BM25 的饱和及长度归一化

最佳匹配评分（Best Matching 25，BM25）的一种常用形式为

$$s(q,d)=\sum_{t\in T_q}\mathrm{IDF}(t)\frac{f(t,d)(k_1+1)}{f(t,d)+k_1(1-b+\frac{b|d|}{\bar{\ell}})}, \tag{12.3}$$

其中**逆文档频率**（Inverse Document Frequency，IDF）取非负变体

$$\mathrm{IDF}(t)=\log\left(1+\frac{N-n_t+0.5}{n_t+0.5}\right). \tag{12.4}$$

本章用自然对数，并将查询重复词去重；其他实现可能保留查询词频因子、采用允许负值的 IDF 或按字段计算长度．公式版本必须固定，不能把不同实现的分数直接比较.

固定长度项 $c_d=k_1(1-b+\frac{b|d|}{\bar{\ell}})>0$，令 $g(f)=\dfrac{(k_1+1)f}{f+c_d}$，有

$$g'(f)=\frac{(k_1+1)c_d}{(f+c_d)^2}>0,\qquad g''(f)=-\frac{2(k_1+1)c_d}{(f+c_d)^3}<0. \tag{12.5}$$

词频贡献递增但增益递减，且 $f\to\infty$ 时趋于 k_1+1．在固定 $f>0, b>0$ 时，长度增大提高分母，使分数下降；$b=0$ 不作长度归一化，$b=1$ 使用完整长度比例．它们校正篇幅带来的机会差异，却不意味着长文档内容必然不重要.

BM25 的数值解释

统计对象与评分对象一致，$N>0, \bar{\ell}>0, k_1>0, 0\leqslant b\leqslant 1$．长度使用同一分析器词项数，语料统计绑定一个索引快照．空对象不参与平均长度的异常除法，查询词项规则已固定.

12.2.4　双文档评分

设 $N=100$，词项甲出现在 10 个对象，乙出现在 50 个对象．则 $\mathrm{IDF}(\text{甲})=\log(1+\dfrac{90.5}{10.5})\approx 2.2637$，$\mathrm{IDF}(\text{乙})=\log 2\approx 0.6931$．取 $k_1=1.2, b=0.75, \bar{\ell}=100$.

文档 A 长 100，甲、乙词频为 $(2,1)$，长度项为 1.2，两个词频因子为 $\dfrac{4.4}{3.2}=1.375$ 和 $\dfrac{2.2}{2.2}=1$，总分约为 $2.2637\times 1.375+0.6931=3.8057$．文档 B 长 200，词频为 $(3,2)$，

长度项为 2.1，因子为 $\frac{6.6}{5.1}\approx 1.2941$ 和 $\frac{4.4}{4.1}\approx 1.0732$，总分约为 3.6735. B 词频更高，却因更长而排名稍低.

此分数是排序量，不是“答案存在概率”. 语料增量会改变 $N, n_t, \bar{\ell}$；同一个固定阈值在不同快照或不同字段上可能失去原意义. 阈值应在目标任务的开发集上校准，并与无答案情况一起评估.

12.3 双编码器及对比学习

12.3.1 独立编码实现离线索引

稠密检索（Dense Retrieval）使用低维连续表示进行匹配. **双编码器**（Dual Encoder）分别计算查询向量 $u = f_\theta(q) \in \mathbb{R}^h$ 和文档向量 $v_d = g_\phi(d) \in \mathbb{R}^h$，以 $s(q,d) = u^{\mathsf{T}} v_d$ 等方式评分. 文档向量可以离线生成，在线仅编码查询并搜索；稠密段落检索是这一设计的典型研究实例 [83].

查询与文档编码器可以共享或不共享参数，还可能使用不同任务前缀. 维度相同不保证空间兼容；两个独立版本的向量不能仅因为都是 h 维就混入同一个索引. 归一化、池化、截断和编码模板均属于表示协议.

12.3.2 对比损失及梯度方向

给定查询 q_i、正例 d_i^+ 和候选文档集合 $\mathcal{C}_i$，**对比学习**（Contrastive Learning）可采用

$$\mathcal{L}_i = -\log \frac{\exp(\frac{s(q_i,d_i^+)}{\tau})}{\sum_{d\in\mathcal{C}_i}\exp(\frac{s(q_i,d)}{\tau})}, \qquad \tau > 0. \tag{12.6}$$

令候选概率为 p_{ij}、正例指示为 y_{ij}，则

$$\frac{\partial \mathcal{L}_i}{\partial s_{ij}} = \frac{p_{ij} - y_{ij}}{\tau}, \qquad \nabla_{u_i}\mathcal{L}_i = \frac{1}{\tau}\left(\sum_j p_{ij} v_j - v_i^+\right). \tag{12.7}$$

梯度提高正例相对分数，并压低有较大当前概率的负例. 温度改变概率集中度和梯度尺度，因此它不是无影响的显示参数.

例如三个候选的相似度除温度后为 $(\log 4, \log 2, 0)$，第一项为正例，归一化概率与损失为

$$p = (\frac{4}{7}, \frac{2}{7}, \frac{1}{7}), \qquad \ell = \log(\frac{7}{4}) \approx 0.5596. \tag{12.8}$$

对这些缩放后分数的梯度为 $(-\frac{3}{7}, \frac{2}{7}, \frac{1}{7})$，当前更像正例的第二项获得更大负向学习压力. 该算例只计算损失，不代表训练效果.

12.3.3　负例构造及假负例

批内负例（In-Batch Negative）把其他查询对应文档用作当前查询负例，减少额外编码．**难负例**（Hard Negative）由词法或现有稠密检索器找出分数高但确实不相关的文档，可以形成更有区分度的学习信号．随机负例通常较容易，过度依赖某一种难例又可能使模型只适应当前检索器的错误．

假负例（False Negative）是真实相关却被标成负例的对象．开放语料中一题往往有多个支持文档，批内文档也可能相互相关；强行推开这些向量会损害召回．可以依据已知相关集合屏蔽冲突、使用多正例损失或对疑似负例复核，但不能把“未标为正”自动理解为“确实不相关”．多正例求和形式还应明确是希望任一正例得高分，还是所有正例都被拉近．

训练与评估应按查询家族、文档版本和必要时间边界隔离．文档原文可以作为开放检索库存在，相关性标注却不能泄漏进查询编码或示例选择．相似文本的近重复治理见第十八章《训练数据工程基础》．

12.4　向量相似度及精确搜索

12.4.1　余弦、内积及欧氏距离

对非零向量 u, v，**余弦相似度**（Cosine Similarity）为 $\dfrac{u^{\mathsf{T}}v}{\|u\|_2\|v\|_2}$．**最大内积搜索**（Maximum Inner Product Search，MIPS）寻找内积最大的向量；**欧氏距离**（Euclidean Distance）满足

$$\|u - v\|_2^2 = \|u\|_2^2 + \|v\|_2^2 - 2u^{\mathsf{T}}v. \tag{12.9}$$

当文档向量范数固定时，对固定查询最大内积与最小欧氏距离排名一致；若查询和文档均归一化为单位向量，则

$$\|\bar{u} - \bar{v}\|_2^2 = 2 - 2\cos(u, v). \tag{12.10}$$

此时三种排名等价．零向量不能作单位归一化，应隔离或采用明确规则．

未归一化时，范数会改变结果．令 $u = (1, 0)$，$v_1 = (0.8, 0.6)$，$v_2 = (2, 2)$．余弦分别为 0.8 和 $\dfrac{1}{\sqrt{2}} \approx 0.7071$，余弦选择 v_1；内积分别为 0.8 和 2，MIPS 选择 v_2．若训练的打分利用范数信息，部署时擅自归一化就改变了目标函数．

12.4.2　精确排名基线

精确最近邻搜索（Exact Nearest Neighbor Search）对所有向量计算指定度量．N 个 h 维向量需要 $O(Nh)$ 评分工作，存储原始单精度向量约 $4Nh$ 字节，尚不含标识和元数据．矩阵批处理可利用高效硬件，但总扫描量仍随 N 增长．

精确搜索给出的是该表示和度量下的最近对象，并非相关性真值．可以将稠密系统误差分成表示误差与索引近似误差：若相关文档在精确前列也不存在，需要改进表示或语料；若精确搜索可以找到但近似索引遗漏，需要调整索引搜索．两者采用不同诊断证据．

12.5　近似索引的两条路线

12.5.1　HNSW 的分层图导航

近似最近邻搜索（Approximate Nearest Neighbor Search，ANN）牺牲部分精确邻居召回，以降低搜索成本．**分层可导航小世界图**（Hierarchical Navigable Small World，HNSW）为向量构建多层邻近图，高层包含较少节点，底层覆盖全部节点 [84]．查询从稀疏高层快速靠近目标区域，再向下扩展候选；底层保留比贪心单路径更多的搜索状态．

图构建时的邻居预算 M、构建搜索宽度与查询搜索宽度影响内存、构建时间和召回．更多边通常提高可导航性，也增加邻接表内存；更宽查询探索更多节点，增加距离计算．启发式邻居选择需要保持方向多样性，若只连接同一局部密簇，图可能难以跨区域导航．

不能对任意数据分布承诺严格对数查询复杂度；实际工作量由图结构、度量、数据簇和过滤条件决定．增量插入使图结构受构建顺序与随机层级影响，删除或大量墓碑也可能降低导航质量．对高选择性权限过滤，需要考虑合法节点的可达性，不能把过滤后的召回退化归咎于编码器．

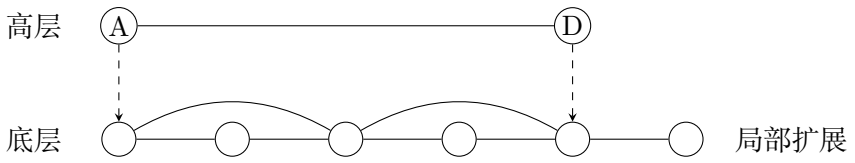


图 12.1: HNSW 的概念层级：高层导航与底层候选探索．示意连接不代表实际构建结果．

12.5.2　倒排文件及乘积量化

倒排文件向量索引（Inverted File Index，IVF）先用粗聚类把向量分到若干单元，再将每个单元中的向量标识组织为列表．查询选择有限个最接近的粗中心，只扫描对应列表．它与词法倒排都使用“键到列表”的组织，但键的含义不同：前者是向量单元，后者是词项．

访问更多单元提高候选覆盖，也增加扫描量．若最近向量被分到未访问单元，后续精确重排无法挽回遗漏．粗聚类若在过时或偏移的数据上训练，单元可能失衡，一

些查询扫描大量向量而另一些查询漏掉边界邻居.

乘积量化（Product Quantization，PQ）把 h 维向量拆成 m 个子向量，每个子空间训练一个含 k 个中心的码本，用 m 个码字索引表示向量 [85]. 若 $k = 256$，每个子码占一字节，向量码为 m 字节，码本及原始精排向量的额外存储另计.

非对称距离计算（Asymmetric Distance Computation，ADC）保留查询为高精度，文档用码字中心近似：

$$\|u - \widehat{v}\|_2^2 = \sum_{j=1}^{m} \|u^{(j)} - c_{j,a_j}\|_2^2. \tag{12.11}$$

查询可先建立每个子空间的距离表，再对每条压缩向量查表求和. 组合 IVF 与 PQ 时，还可对向量减去粗中心后的残差编码，查询距离必须计入同一个中心，不能把不同单元的残差当成同一原始坐标.

若 $v = \widehat{v} + e$，反三角不等式给出 $|\|u - v\|_2 - \|u - \widehat{v}\|_2| \leqslant \|e\|_2$. 平方距离误差满足

$$\left|\|u - \widehat{v}\|_2^2 - \|u - v\|_2^2\right| \leqslant 2\|u - v\|_2\|e\|_2 + \|e\|_2^2. \tag{12.12}$$

小距离间隔的候选更易因近似误差交换名次. IVF 的单元漏检与 PQ 的距离失真是两种误差；扩大扫描单元不能直接消除后者，精排压缩候选也不能恢复前者.

12.5.3　索引资源的比较

HNSW 主要增加图导航结构，IVF 限制访问区域，PQ 压缩表示及距离计算. 它们不是互斥产品标签，也可以组合. 选择时应比较同一表示、同一授权语料和相同精确搜索参照下的 ANN 召回、尾时延、内存、构建成本与更新行为，而非只比较一次查询的平均耗时.

12.6　混合检索及学习排序

12.6.1　分数融合的尺度问题

混合检索（Hybrid Retrieval）结合词法和稠密候选. 前者对精确术语、数字标识和罕见字符串有明确匹配依据，后者能够利用训练得到的语义关系；但两者均有失败场景. 简单分数和 $\lambda s_{\text{lex}} + (1 - \lambda) s_{\text{dense}}$ 只有在尺度处理明确后才有意义. BM25 与余弦的数值范围和分布不同，逐查询最小最大归一化又受异常候选和截断深度影响.

倒数排名融合（Reciprocal Rank Fusion，RRF）直接使用排名 [86]：

$$s_{\text{RRF}}(d) = \sum_{j:d \in R_j} \frac{w_j}{c + \operatorname{rank}_j(d)}, \qquad c > 0. \tag{12.13}$$

未进入某路截断列表的文档在该路贡献为零. c 控制顶部名次差异的影响，w_j 为预先确定的路权重. RRF 不需要跨路原分数可比，却丢失了分数间隔信息；两路高度重复或错误高度相关时，不保证优于单路.

取词法排名 (A, B, C)、向量排名 (B, D, A)，$c = 10, w_j = 1$. 得到

$$s(A) = \frac{1}{11} + \frac{1}{13} \approx 0.16783, \tag{12.14}$$

$$s(B) = \frac{1}{12} + \frac{1}{11} \approx 0.17424, \tag{12.15}$$

$$s(C) = \frac{1}{13} \approx 0.07692, \qquad s(D) = \frac{1}{12} \approx 0.08333. \tag{12.16}$$

融合顺序为 B, A, D, C. 这里用 $c = 10$ 便于算例，不是通用最佳设置. 相同文档的多个重复块若被当作独立对象，融合也不会自动识别其冗余，必须先明确对象身份与文档聚合规则.

12.6.2 重排及候选集上界

重排序（Reranking）在有限候选集上使用更精细的评分. **交叉编码器**（Cross Encoder）联合输入查询与文档，允许细粒度交互，代价是不能像双编码器那样为每篇文档独立预计算最终相关性. **学习排序**（Learning to Rank，LTR）可以组合词法分数、稠密分数、结构和其他合法特征.

例如已标注正负文档对可使用成对损失 $\log(1 + \exp[-(s^+ - s^-)])$，推动正例高于负例. 训练样本来自旧排序器时，需要注意曝光和位置偏差，点击也不等同于完全相关. 无论重排器多强，若正确文档不在候选并集中，它都无法将其排到前列；应把召回深度与重排预算共同评价.

12.7 权限、版本及增量更新

12.7.1 检索过滤约束

元数据过滤（Metadata Filtering）约束租户、文档类型、语言、生效时间与访问权限. 一个授权查询的检索集合应是当前用户可访问的 $\mathcal{D}_u$，不是全库评分后让生成器自行忽略秘密. 查询时过滤、返回前再次授权检查和缓存隔离应共同保证撤销后的对象不被消费.

只取全库前 K 再删除无权结果，可能返回少于 K 条且漏掉合法相关对象. 增加预取深度只能缓解，不提供严格召回保证. ANN 图导航还可能需要经过不符合过滤条件的节点，其标识与内容的内部使用边界应由可信检索服务定义；绝不能向调用方暴露未授权结果. 高选择性过滤场景应单独测量授权集合上的召回与时延.

12.7.2　索引增量更新

每条索引对象应绑定稳定文档身份、源版本、片段边界、内容摘要、编码器版本、分析器和索引构建配置．文档编辑可能使后续切分边界整体变化，不能仅按旧片段序号覆盖而保留失效片段．编码器更新则通常要求重编码，并在完整新索引就绪后切换查询编码器与文档空间．

增量处理可记录变更日志，采用幂等写入和不可变版本．**删除标记**（Tombstone）可先阻止旧条目返回，后续通过物理压缩清理图节点、倒排条目、向量和存储．删除还需传播到片段缓存、检索缓存与派生索引；清理内容不等于自动消除所有统计或旧快照，需要分别规定保留边界．

索引快照（Index Snapshot）将语料版本、编码器、分析器、统计及物理索引绑定为一致消费单位．新版本采用原子可见切换，回滚时恢复匹配的查询配置．权限撤销属于即时约束时，不能因为回滚索引而重新授权已经撤销的内容．

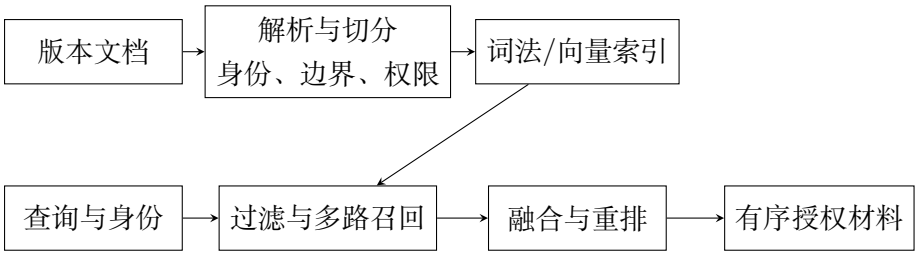


图 12.2: 离线语料组织与在线检索链路．生成回答不属于本图的检索职责．

算法 12.1　具有版本边界的混合检索

输入：查询、已认证身份、一致索引快照、候选深度与排序配置．**输出：**有序可访问对象及来源．**状态：**合法过滤条件、各路候选、融合结果．

1. 解析访问条件与时间范围，选择匹配的分析器和查询编码器；拒绝不兼容版本．
2. 在授权过滤语义下分别执行词法与向量检索，记录真实候选深度及超时状态．
3. 按稳定对象身份合并候选，使用预定分数融合或 RRF，按预算重排并处理同文档冗余．
4. 返回前再次核对当前授权与删除状态，输出实际材料身份、版本、排序分数和诊断状态；资源预算耗尽时按预定规则结束．

不变量：返回内容经过授权，表示与索引版本一致，召回失败或回退不被伪装为正常完整结果．

12.8 检索评价指标

12.8.1 覆盖、首条命中及分级排序

召回率（Recall）衡量相关集合被覆盖的比例：

$$\mathrm{Recall@K}(q)=\frac{|G_q\cap R_q^{(K)}|}{|G_q|},\qquad |G_q|>0. \tag{12.17}$$

命中率（Hit Rate，Hit@K）则是 $\mathbf{1}[G_q\cap R_q^{(K)}\neq\varnothing]$. 存在三篇相关文档、只取回一篇时，前者为 $\frac{1}{3}$，后者为 1；二者不能混称. 没有相关对象的查询分母为零，应单独评价无答案检出或拒绝，而不是擅自记为满分.

平均倒数排名（Mean Reciprocal Rank，MRR）关注第一条相关结果. 若其位置为 j_q，单查询倒数排名为 $\frac{1}{j_q}$，未在规定截断内命中则为零，再对查询平均. 它不奖励第一条之后更多相关材料，因此与多证据任务的覆盖目标不同.

折损累计增益（Discounted Cumulative Gain，DCG）按等级与位置计算价值，**归一化折损累计增益**（Normalized Discounted Cumulative Gain，nDCG）用理想排序归一化. 累积增益评价的基础见 Järvelin 与 Kekäläinen[87]. 本章明确采用一种常见变体：

$$\mathrm{DCG@K}=\sum_{j=1}^{K}\frac{2^{r_j}-1}{\log_2(j+1)},\qquad \mathrm{nDCG@K}=\frac{\mathrm{DCG@K}}{\mathrm{IDCG@K}}. \tag{12.18}$$

理想分母由该查询全部已知相关对象按等级排序取前 K 得到，不能只重排系统已返回对象. 若分母为零，应规定无增益查询的独立口径. [1]

12.8.2 同一结果的三种指标

某查询已知相关对象为 A,B,C，等级分别为 $3,2,1$，其余对象等级为零. 系统前五名为 (D,B,E,A,F)，前四名相关等级为 $(0,2,0,3)$. 因此

$$\mathrm{Recall@4}=\frac{2}{3},\quad \mathrm{Hit@4}=1,\quad \mathrm{RR@4}=\frac{1}{2}. \tag{12.19}$$

DCG 为 $\frac{3}{\log_2 3}+\frac{7}{\log_2 5}\approx 1.8928+3.0147=4.9075$. 理想等级为 $(3,2,1,0)$，IDCG 为 $7+\frac{3}{\log_2 3}+\frac{1}{\log_2 4}\approx 9.3928$，所以 nDCG@4 约为 0.5225. 相关对象 A 虽然已召回，却位于第四位，折损反映其排名损失.

[1] DCG 有线性与指数增益、不同折损位置等约定. 比较报告前必须固定公式、相关等级含义和无相关查询处理，不能仅凭相同指标名称认定可比.

若第二个可回答查询的第一条相关结果在第三位，则两查询 MRR@4 为 $\frac{\frac{1}{2}+\frac{1}{3}}{2}=\frac{5}{12}\approx 0.4167$. 另有无答案查询时，应单列是否错误返回被当作证据的材料，而不混入上述需要相关集合的分母. 这些是构造标注上的完整计算，不代表实际系统质量.

12.8.3　ANN 召回及相关性召回分开

给定精确向量前 K 集合 $E_q^{(K)}$，可定义 ANN 召回为 $\frac{|E_q^{(K)}\cap A_q^{(K)}|}{K}$，用于衡量近似索引对向量排名的保持. 它与基于 G_q 的任务召回不同. 一个索引可以完美复现不相关的向量排名，也可以因为近似扰动偶然取得更高任务分数；不能据此认为其表示一定更好.

标注不完整时，未标注文档不一定不相关，召回分母与 nDCG 理想排序均只反映当前已知集合. 应保存标注来源，进行多路候选池复核，并按精确实体、语义改写、跨语言、长文档、时效、权限过滤和无答案查询等难例切片报告. 宏平均让每个查询等权，按相关对象数加权则回答不同问题.

检索故障的定位顺序

先确认解析与权限范围，再检查相关材料是否进入精确候选，随后检查近似索引与融合是否遗漏，最后检查重排是否把有效材料移出前列. 召回、排名与下游答案质量承担不同职责，不能用某一个总分代替全部链路证据.

第十三章　检索增强生成

检索系统找到相关材料之后，问题并没有结束．材料可能重复、过期、缺少例外条款，或者只支持答案的一部分；生成器也可能忽略关键限定，把相似内容拼成错误结论．一个完整的 RAG 系统必须解释证据如何进入生成、每条声明由什么支持，以及知识更新和权限变化后哪些结果仍然有效．

第十二章《信息检索》讨论索引、词法与向量检索的基础．本章承接候选召回，进一步建立重排、证据选择、生成与知识服务的关系．书中使用的性能和质量数例均为构造输入上的计算，不是任何产品或部署的实测结果．

13.1　检索增强的概率含义

13.1.1　检索作为潜变量

检索增强生成（Retrieval-Augmented Generation，RAG）利用外部材料改变生成条件．原始 RAG 将检索文档看作**潜变量**（Latent Variable），结合检索分布与条件生成模型，而不只是把搜索结果放进提示词．[88]

概率模型与服务边界

给定查询 q 和固定知识快照 $\mathcal{D}$，检索对象 z 为文档或片段，答案 $y=(y_1,\dots,y_T)$ 为有限词元序列．概率模型中的文档分布已归一化．实际系统中，知识范围还受身份、权限和时间条件约束；所有生成器可见的证据必须先满足这些外部约束．检索分数、重排分数与答案正确概率不得未经校准混用．

本章主要符号见表 13.1．

表 13.1：检索增强生成的主要符号与形状．

符号	含义与形状
$q, y, z, \mathcal{D}$	查询、生成答案、潜在文档与知识快照．
$p_\eta(z \mid q), p_\theta(y \mid q, z)$	参数为 η 的检索分布和参数为 θ 的条件生成分布．
$\mathcal{C}_K, E$	召回的 K 个候选与最终有序证据包．
$f_\eta(q), g_\eta(d) \in \mathbb{R}^h$	查询与文档的 h 维稠密表示．

续下页

符号	含义与形状（续）
s_i, γ_i, l_i	候选分数、给定答案后的文档后验责任、证据序列化词元成本.
$u_i \in \{0,1\}, C_E$	证据选择变量与证据词元预算.
A_{ji}	证据 i 是否覆盖问题要点 j 的指示量.
I, P, v	已认证身份、当前权限策略、知识或组件版本.

对整条回答共享一个潜在文档，**序列级 RAG**（RAG-Sequence）写为

$$p_{\mathrm{seq}}(y \mid q) = \sum_{z \in \mathcal{D}} p_\eta(z \mid q) \prod_{t=1}^{T} p_\theta(y_t \mid q, z, y_{<t}). \tag{13.1}$$

先在每个文档条件下计算整条答案概率，再对文档求和. 对每个词元分别边缘化，**词元级 RAG**（RAG-Token）则为

$$p_{\mathrm{token}}(y \mid q) = \prod_{t=1}^{T} \sum_{z \in \mathcal{D}} p_\eta(z \mid q) p_\theta(y_t \mid q, z, y_{<t}). \tag{13.2}$$

求和与乘积位置不同，意味着模型假设不同：后者允许不同生成位置由不同文档项支持. 这里的词元级潜变量边缘化不等于服务端每生成一个词元都必须重新调用搜索引擎.

大语料无法直接遍历全部文档，通常只计算候选集 $\mathcal{C}_K$. 如果保留原完整分布的概率，截断求和是一种近似，其总质量可能小于 1；如果在候选集重新归一化，定义的是条件于候选集合的分布. 记原候选质量为 $Z_K = \sum_{z \in \mathcal{C}_K} p_\eta(z \mid q)$，则局部归一化使用 $\widetilde{p}_\eta = \dfrac{p_\eta}{Z_K}$. 两种口径不能在训练与解释时无说明地互换.

设两个候选文档权重为 $(0.6, 0.4)$，目标答案由两个词元组成. 第一文档对两个词元的条件概率分别为 0.9 和 0.2，第二文档分别为 0.1 和 0.8；第二个概率均已条件于相同的目标前缀. 序列级概率为

$$p_{\mathrm{seq}} = 0.6(0.9 \cdot 0.2) + 0.4(0.1 \cdot 0.8) = 0.14. \tag{13.3}$$

词元级概率则为

$$p_{\mathrm{token}} = (0.6 \cdot 0.9 + 0.4 \cdot 0.1)(0.6 \cdot 0.2 + 0.4 \cdot 0.8) = 0.2552. \tag{13.4}$$

差异来自第二个模型可以在不同位置混合文档贡献，不是计算误差. 两个数值都描述设定模型对给定词元序列的概率，不能直接解释为答案事实正确率.

序列级模型还可推导检索训练信号. 令 $a_i = p_\theta(y \mid q, z_i)$，$p_i = \mathrm{softmax}(s)_i$，$Z = \sum_i p_i a_i$. 给定答案后的文档后验为

$$\gamma_i = \frac{p_i a_i}{Z}. \tag{13.5}$$

利用 softmax 导数 $\frac{\partial p_i}{\partial s_j} = p_i(\mathbf{1}[i=j] - p_j)$，

$$\frac{\partial Z}{\partial s_j} = p_j a_j - p_j Z, \tag{13.6}$$

$$\frac{\partial(-\log Z)}{\partial s_j} = p_j - \gamma_j. \tag{13.7}$$

例题中 $\gamma_1 = \frac{0.108}{0.14} = \frac{27}{35} \approx 0.7714$，$\gamma_2 \approx 0.2286$. 第一文档的梯度约为 -0.1714，梯度下降会提高其检索分数. 这个信号奖励能解释训练答案的候选，不自动保证文档真实或其引用充分；候选集合之外的文档也不会在该局部计算中得到直接信号.

13.1.2 检索上下文组装

常见服务先选择并序列化证据包 $E = \mathcal{A}(q, \mathcal{C}_K)$，再调用

$$p_\theta(y \mid q, \mathrm{serialize}(E)). \tag{13.8}$$

这允许生成器同时阅读多个片段并综合条件，却没有显式计算式 (13.1) 中的文档概率边缘化. 重排分数只决定排序或选择，除非目标另行定义，并不会自动成为生成概率中的混合权重.

证据拼接具有明确的工程可控性：可以带上来源、时间、权限与引用标识，也容易替换检索器. 但生成器的注意力分配并不等同于已知的证据后验. 某文档排在首位或被注意力关注，不能直接证明最终声明由它支持. [1]

13.2 召回、重排及训练关系

13.2.1 多路检索的目标

稀疏检索（Sparse Retrieval）通过词项维度匹配材料，BM25 使用词频饱和、逆文档频率和长度校正；**稠密检索**（Dense Retrieval）通过学习表示的相似度选择材料. 基础公式见前章. 本章关注它们在管线中的不同职责：词法路径保留编号、专名和少见术语的精确线索，语义路径连接不同表达，均不能保证命中答案所需的全部限定.

稠密双编码器常使用 $s(q,d) = f_\eta(q)^\top g_\eta(d)$. 对正例 d^+ 与候选负例集 $\mathcal{N}$，一种训练目标为

$$\mathcal{L}_{\mathrm{ret}} = -\log \frac{e^{\frac{s(q,d^+)}{\tau_r}}}{e^{\frac{s(q,d^+)}{\tau_r}} + \sum_{d^- \in \mathcal{N}} e^{\frac{s(q,d^-)}{\tau_r}}}, \tag{13.9}$$

[1] 当前工程语境常把多种“检索后生成”架构统称为 RAG. 本章保留这一广义用法，同时用 RAG-Sequence 与 RAG-Token 指称原论文中的具体概率形式，避免同名导致目标混淆.

其中 $\tau_r > 0$ 是排序温度．它使正例相对负例更靠前；负例采样和潜在假负例会改变学习信号．DPR 是采用双编码器进行段落检索的重要实例．[83] 这一相对训练目标仍不提供跨任意问题通用的事实正确概率．

混合检索（Hybrid Retrieval）将多路候选取并集，再做融合．若使用分数线性组合 $s_h = \alpha\widetilde{s}_{\text{lex}} + (1-\alpha)\widetilde{s}_{\text{dense}}$，应先定义两路尺度校准；原始 BM25 与余弦分数直接相加通常缺少稳定解释．也可按名次融合，例如 $s_h(d) = \sum_j \dfrac{1}{c + \text{rank}_j(d)}$，其中 $c > 0$ 平滑名次影响，未召回项不贡献分数．此类融合得到排序量，不是经过概率归一化的证据置信度．

并集召回提高候选覆盖的可能性，也会增加重复、噪声和后续重排成本．只有被所有查询路径统一授权的候选才能进入并集，不能让某条备用路径绕过身份过滤．

13.2.2　联合编码重排

重排（Re-ranking）在较小候选集上计算更精细的查询与文档关系．**交叉编码器**（Cross-encoder）共同编码查询和候选内容，可以直接比较限定条件、否定关系和实体对应，通常比独立向量内积具有更强的交互能力，但每个查询文档对都需执行模型．使用 BERT 进行段落重排的工作展示了这一模式．[89]

若候选得分为 $a_i = r_\phi(q, d_i)$，成对目标可写为

$$\mathcal{L}_{\text{pair}} = -\log \sigma(a_+ - a_-). \tag{13.10}$$

其对正例分数的导数为 $-\sigma(a_- - a_+)$，对负例为相反数，促使正例相对提升．若有多个相关候选，列表目标可用标签分布 t_i 对 softmax 分数做交叉熵：

$$\mathcal{L}_{\text{list}} = -\sum_i t_i \log \frac{e^{a_i}}{\sum_j e^{a_j}}. \tag{13.11}$$

成对和列表目标都必须明确相关性含义．与问题主题相似、包含某个答案词、足以支持结论，分别对应不同标签．训练只标注主题相关性，部署却要求证据充分性，会形成目标错位．

重排只能调整已进入候选集的对象，无法恢复完全漏召回的事实．候选文本截断还可能丢失例外条款，使重排器看见的内容与生成器最终读取的材料不同．因此重排输入范围和证据正文范围应可追踪，不能只保存一个浮点分数．

13.2.3　检索模型训练

嵌入模型优化召回，重排器优化候选顺序，生成器优化在证据条件下回答．三者可以分别训练，也可让后者提供监督，但训练依赖必须明确．式 (13.7) 提供一种答案似然到候选分数的信号；离散近邻搜索的候选成员变化并不能由普通反向传播完整穿过，因此“联合训练”仍可能采用固定候选、近似梯度或交替刷新索引．

文档编码器更新后，旧向量索引通常不再与查询向量同属一致表示空间，必须重建或采用明确的过渡策略．只更新查询侧、冻结文档编码器可以降低刷新成本，但限制了可调整范围．重排或生成模型升级也可能改变证据选择和引用行为，即使文档索引未变仍需独立比较．

生成器训练需要包含充分证据、缺失证据和冲突证据，而不是所有样本都提供完美材料．否则训练目标鼓励始终作答，部署拒答就只能依赖脆弱的提示．用模型自身答案生成检索监督还可能强化既有错误，教师或生成分数应视为弱监督并保留独立证据标注．

13.3 证据预算及选择

13.3.1 有效长度约束

证据组装（Evidence Assembly）把候选变成可被生成器读取的有序材料．总上下文容量 C_{ctx} 应满足

$$C_E \leqslant C_{\mathrm{ctx}} - C_{\mathrm{sys}} - C_q - C_{\mathrm{hist}} - C_{\mathrm{out}} - C_{\mathrm{reserve}}. \tag{13.12}$$

各项分别是系统模板、问题、对话历史、输出与保留余量．证据成本 l_i 必须按实际分词器和序列化格式计算，包含标题、来源与引用标识，而不只是正文字符数．预算为负时应调整上下文策略，不能静默裁掉必要约束．

选择变量 $u_i \in \{0,1\}$ 满足 $\sum_i u_i l_i \leqslant C_E$．只最大化 $\sum_i u_i s_i$ 是带预算的相关性选择，但不能自动保证多要点覆盖或去重．若问题被分为要点 j，覆盖指示为 A_{ji}，可定义

$$F(S) = \sum_j w_j \min\left(1, \sum_{i\in S} A_{ji}\right) - \lambda \sum_{i<k,\ i,k\in S} r_{ik}, \tag{13.13}$$

其中 $r_{ik} \geqslant 0$ 衡量重复，w_j 表示要点权重．覆盖部分具有重复收益递减的性质；加入任意重复惩罚后，整体未必单调，简单贪心的最优性保证也需重新判断．

来源权限、有效日期和必要限定通常应作为约束，而不是用一个小罚分抵消．例如证据只在某版本产品下适用，就不能因相关性高而压过版本不符．多个来源互相转述也不一定提供独立佐证，来源多样性应追踪原始出处．

设总上下文为 1024 词元，系统模板 180、问题 80、历史 160、输出 240、余量 44，可用证据预算为 320．查询需要三个要点：A 为适用条件，B 为例外情况，C 为生效日期．候选如下，长度已包含引用元数据．

片段	长度	重排分数	支持要点
d_1	220	0.95	A
d_2	180	0.85	A、B
d_3	120	0.75	C
d_4	100	0.70	B

顺序装入最高分 d_1 后只剩 100 词元，可以装入 d_4，用满 320 却缺少 C. 选择 d_2, d_3 只用 300 词元，覆盖全部三个要点. 即使采用分数总和，前一方案为 1.65、后一方案为 1.60，仍会偏向不完整的证据包. 检索分数之和因此不能替代任务覆盖目标.

如果 d_2 的例外条款依赖前一段定义，应把必要上下文的额外成本纳入 l_2，重新选择，而不是截断后仍声称覆盖 B. 若 C 的来源已过期，则两个方案都可能不足；预算优化只能在有效证据中选择，不能创造缺失事实.

13.3.2　压缩、排序及引用锚点

证据过长时，可以提取相关句段或生成摘要. **上下文压缩**（Context Compression）应保留数值、否定、范围与例外；生成摘要可能加入原文不存在的连接推断，因此摘要仍需映射回原始片段. 引用不应只指向一个不可复原的摘要缓存.

排序影响模型读取，但不是数学上的事实优先级. 可按子问题组织证据，把一般规则与例外相邻放置，并标明来源日期和适用条件. 重复块可合并展示，同时保留全部来源映射；多块共享同一引用标识会使声明定位模糊. 证据 ID 应稳定指向文档版本及局部位置，而不能只使用随排序变化的“第 3 条”.

13.4　查询变换及检索控制

13.4.1　查询改写约束

查询改写（Query Rewriting）把用户措辞转换为更适合检索的表达，例如补齐多轮对话中的指代、展开缩略语、产生同义表达. 改写器可由规则或语言模型实现，也可根据后续阅读器的反馈训练 [90]. 但是，检索友好性不等于语义等价性. 把“去年生效的规则”改写为“最新规则”，会提高某些文本的相关分数，却改变问题的时间边界.

因此，查询状态至少分为原始问题、可变的检索表达与不可由模型修改的约束. 用户身份、租户、授权范围、时间口径、已明确的实体标识和输出语言应由控制平面保存；每次改写只生成候选表达，不覆盖原始请求. 若对话指代仍有歧义，系统可以并行检索明确列出的候选实体，或要求澄清. 把不确定指代强行绑定到最高频实体，会使后续全部证据看似一致而实际答非所问.

设原始问题为 q，固定约束为 b，第 j 个检索表达为 $q^{(j)} = g_j(q, b)$. 多路检索取得

$$\mathcal{C} = \bigcup_j \mathrm{Retrieve}(q^{(j)}; \mathcal{D}_{I,P}, b), \tag{13.14}$$

其中 $\mathcal{D}_{I,P}$ 是当前身份和策略允许读取的语料. 合并后按文档版本及片段位置去重，再使用原始问题和约束重排. 这样可以避免改写器生成的一条宽泛查询主导最终答案，也使“某个候选来自哪条改写”成为可审计信息.

13.4.2 分解及结构化知识

查询分解（Query Decomposition）把复合问题拆为若干子问题. 独立子问题可以并行；存在实体或时间依赖的子问题则应构成有向无环图. 例如“某项目负责人管理的服务在上月发生了几次故障”，应先确定指定时点的负责人和服务范围，再查询相同时间窗口内的故障记录. 只按句号拆分不能保证依赖正确，现任负责人也不能自动代表上月负责人.

结构化数据适合精确筛选、连接与聚合，文本适合解释规则与例外. 系统可以把问题映射为受约束的查询计划，但字段、连接路径、读写权限、扫描范围和结果上限由执行层校验. 模型提出的查询不是数据库授权. 聚合结果应附带查询口径、数据快照、过滤条件和可追溯的来源记录范围；“总数为 0”“没有匹配数据”和“权限不足，无法查询”必须使用不同状态，不能都被生成器解释成零.

图结构知识同样需要语义约束. 两条边共享实体不意味着关系可传递；路径检索所得事实必须保留边类型、方向、有效时间与来源. 把图路径和表格结果转换为统一证据对象，有助于生成器跨来源组织回答；这种统一不应抹去原来的精确查询条件.

13.4.3 线性、条件、分支及迭代流程

线性流程按改写、检索、重排、组装、生成依次运行，适用于证据需求已知的问题. 条件流程在确定性信息充分时选择不同路径，例如精确文档 ID 走版本读取，统计问题走结构化查询，解释性问题走文本检索. 路由器的错误率本身需要评估；不能把所有端到端失败都归因于检索器.

分支流程同时调用不同检索通道，随后融合. 设分支耗时为 $t_1, \ldots, t_J$，在资源充足且没有排队竞争的理想条件下，等待全部分支的耗时约为 $\max_j t_j + t_{\mathrm{merge}}$，而调用成本仍接近各分支成本之和. 超时返回部分结果时，必须把缺失通道记录为未完成，不能把它当成“该通道没有证据”.

迭代检索（Iterative Retrieval）根据当前证据缺口提出新查询. 主动检索研究还探讨在生成过程中触发额外检索 [91]. 这里需要区分词元低概率和事实缺失：罕见专有名词可能概率低但证据充分；常见却错误的说法也可能概率高. 触发信号可以辅助决策，不能充当事实正确性证明.

令 E_t 为第 t 轮有效证据，U_t 为尚未解决的事实需求集合，则一种明确的状态转移为

$$q_{t+1} = \mathrm{Rewrite}(q, b, E_t, U_t), \tag{13.15}$$

$$E_{t+1} = \mathrm{Select}\big(E_t \cup \mathrm{Retrieve}(q_{t+1}), C_E\big), \tag{13.16}$$

$$U_{t+1} = \mathrm{Assess}(q, b, E_{t+1}). \tag{13.17}$$

式中的检索与选择均隐含授权和版本检查．需求评估可以由规则与模型共同完成，但必须保留不确定状态．停止条件至少包括证据已足够、连续一轮没有新增有效证据、迭代次数达到上限、词元预算耗尽和截止时间到达．终止后未解决的问题应进入有限回答或拒答路径，而不是因为“轮数已经用完”变成确定结论．

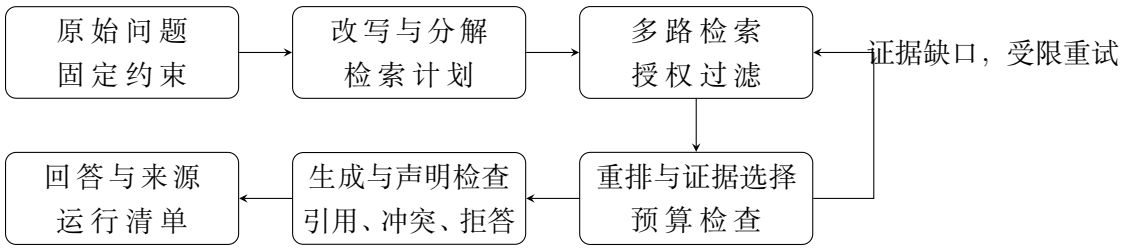


图 13.1: 受约束的检索生成流程．迭代补充的是证据；身份、时间范围和资源上限由外部控制层保持．

算法 13.1　有界证据检索与回答

1. 接收问题 q、身份 I、固定约束 b 与截止时间；取得语料快照和当前授权策略．初始化证据 $E = \varnothing$，未解决需求 U．
2. 在轮数、调用量、词元与时间均未超限时，依据 q, b, U 生成检索计划；校验计划只包含允许的数据源和只读操作．
3. 并行执行可独立的分支，区分成功为空、失败和超时．合并候选，检查当前权限、来源版本与有效时间，记录各分支来源．
4. 用原始问题重排候选；在实际序列化词元预算内选择证据，保留定义、例外与引用锚点．更新 U．
5. 若证据充分则结束检索；若没有新增有效证据或预算耗尽则保留未解决状态并结束；否则继续针对缺口检索．
6. 根据证据生成有限范围的回答，把声明映射至来源．检查引用解析、语义支持和冲突；失败声明删除、限定或转为拒答．
7. 输出前再次检查访问权限，返回答案状态、来源版本、未解决事项和运行清单．权限撤销使相应证据失效时，不返回依赖它的答案．

13.5 引用、冲突及拒答

13.5.1 声明证据映射

证据忠实度（Evidence Faithfulness）衡量回答声明是否受到给定证据支持. 它不同于答案在外部世界中是否正确：模型可能凭参数记忆答对，却引用了不相关段落；也可能忠实复述一份已经过期的文件. 因此，引用评估必须与答案正确性分别进行，相关研究也把引用质量作为独立评价维度 [92].

设答案包含可验证声明 $c_1, \ldots, c_M$，来源集合为 E. 引用关系 $J \subseteq \{1, \ldots, M\} \times E$ 不只是格式标记，还应附带支持、矛盾或证据不足的判定. 检查分为三层：引用标识是否属于本次证据包；标识是否仍能解析到指定版本及片段；该片段或明确组合的片段是否支持声明. 仅验证第一层可以阻止虚构 ID，却不能阻止使用真实 ID 进行错误归因.

一句话可能同时包含一般规则、时间条件与例外，应按可独立核实的主张划分，而非机械按标点切分. 跨文档推导需标明各来源承担的前提；计算结论需保留输入数值、单位和计算规则. 来源提到“多数情况适用”不能支持“所有情况适用”，来源写“建议”也不能支持“必须”. 自动蕴含判别可作为检查器，但其输出依然需要在领域样本上校准，不能把另一个模型的判断当作形式证明.

13.5.2 新鲜度及冲突的处理顺序

数据新鲜度（Data Freshness）描述服务使用的数据与所需时间状态之间的差距. 抓取时间、发布日期和生效时间是不同字段. 今天重新抓取的旧文件，不会因此成为新规则；下月才生效的文件，也不能直接用于回答今天的状态.

处理冲突时，先确认实体、辖域、版本、有效时间和适用对象是否一致；若适用条件不同，应分别回答. 只有条件相同而结论不一致时，才属于待解决的实质冲突. 来源权威性也应由领域规则确定，不能从文风或搜索名次推断. 无法确定替代关系时，应并列说明冲突及来源时间，并收缩结论范围.

例如，以下是为分析构造的虚拟制度：版本 v1 规定普通批处理任务上限 30 分钟，版本 v2 自某日开始调整为 45 分钟；同时存在“维护窗口内上限 10 分钟”的特殊条款. 问普通工作日的当前限制，应核对 v2 是否已生效；问维护窗口，须保留 10 分钟例外；问历史日期，则可能仍使用 v1. 把三条证据合并成“上限平均为 28.3 分钟”没有语义依据. 若维护窗口的适用范围缺失，回答应明确这一缺口，而不能选取分数最高的 45 分钟.

13.5.3 选择性回答的决策条件

拒答（Abstention）是证据不足时的一种合法输出状态，包括完全拒答和只回答已证实部分. 它不同于访问拒绝、服务异常或问题超出支持范围，后者需要分别记录，

避免质量指标掩盖系统故障.

可以通过一个简化决策模型说明阈值的来源. 设 p 是在当前证据条件下某一拟答结论正确且适用的校准概率，答错成本为 C_w，拒答成本为 C_r，答对成本为 0，且 $0 < C_r < C_w$. 回答与拒答的期望成本分别为

$$R_{\text{answer}} = (1-p)C_w, \qquad R_{\text{abstain}} = C_r. \tag{13.18}$$

选择回答的条件为

$$(1-p)C_w \leqslant C_r \quad \Longleftrightarrow \quad p \geqslant 1 - \frac{C_r}{C_w}. \tag{13.19}$$

例如 $C_w = 20, C_r = 1$ 时，阈值为 0.95. 这个数值是给定成本模型的推论，不是通用 RAG 参数. 真实系统还可能区分不同错误类型以及部分回答的成本. 尤其不能把 BM25 分数、向量内积或语言模型最大词元概率直接代入 p；这些量没有自动获得所需的概率语义.

阈值应在与目标分布相符的开发集上选择，报告回答覆盖率和已回答问题的错误率，并单独分析知识库不可回答的问题. 若只奖励拒答准确率，系统可能拒绝所有问题；若只奖励已回答准确率，也可能通过少答取得表面提升. 可用性与错误风险应同时呈现.

13.6　知识服务的数据及执行契约

13.6.1　离线知识管线

完整服务首先需要可追溯的知识摄取流程：从授权来源取得原始文件及版本，解析为段落、表格和结构块，保存来源位置，按语义关系切分，计算检索表示，构建索引，最后发布可查询快照. 解析与切分会改变信息可见性，必须保留原始文件到片段的映射. 表头、单位、脚注与跨页续表若被丢弃，即使后续向量检索完全准确，也无法恢复正确语义.

数据契约（Data Contract）规定各阶段对字段、版本和失败状态的共同解释. 文档对象至少包含稳定文档 ID、源版本、内容哈希、来源位置、有效时间、观测时间、授权标签和删除状态；片段额外包含父文档版本、局部位置、解析器与切分规则版本. 嵌入记录还要包含模型和预处理版本. 向量维度相同并不表示表示空间相容，因此模型升级不能仅检查维数后混用旧向量.

快照清单（Snapshot Manifest）记录一次发布使用的文档集合、索引分片、解析和嵌入版本，以及构建完成状态. 新快照在暂存区完成后再切换查询指针，使一次查询使用可解释的版本组合. 若数据量要求增量发布，也应显式定义可见水位及各分片版本，不能把“最终会同步”表述为当前强一致.

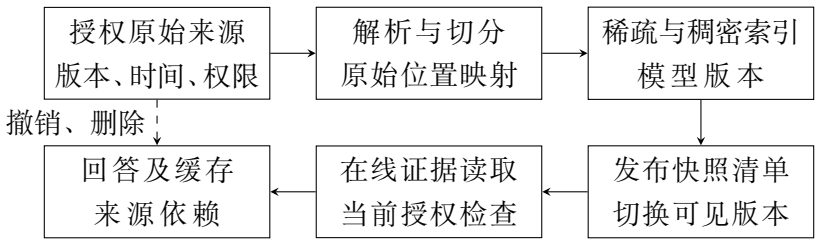


图 13.2: 知识发布与在线读取的关系. 内容快照保证可复现性，权限撤销仍须及时作用于在线访问.

13.6.2 增量更新及故障恢复

知识摄取常由**变更数据捕获**（Change Data Capture，CDC）或周期扫描驱动. 系统应假定事件可能重复、乱序或暂时丢失，并使用源版本与操作类型构造幂等键. 重复处理同一版本不应生成新的逻辑片段；较旧更新也不应覆盖较新的删除标记. 没有全局递增版本的来源，可使用来源提供的条件读取与内容哈希，并明确它只能判断哪些一致性关系.

解析失败应保留原始对象和错误原因，便于使用修复后的解析器重放. 嵌入失败可以重试该版本的片段；索引写入部分成功时，发布状态不能提前宣称整份文档完成. 重试次数耗尽的对象进入待修复队列，并纳入覆盖率和新鲜度报告. 将失败文件悄悄丢弃，会使回答系统把摄取故障误认为知识不存在.

删除必须传播到片段、索引、摘要、引用解析服务和答案缓存. 逻辑删除标记可先阻止在线读取，后台再回收物理数据；权限撤销则应优先改变访问判定，不等待所有索引副本重建. 历史审计需要保留什么数据、保留多久，由独立的数据治理规则决定，不能用“为了可复现”无限保存已经不应保留的内容.

13.6.3 在线对象及失败状态

表 13.2 列出模块之间需要交换的核心对象. 分数、文本与来源必须一起传递；只返回字符串列表，会丢失授权、版本和重试语义.

表 13.2: 知识服务的最小逻辑数据契约. 具体编码形式可变，字段语义应保持稳定.

对象	关键字段及含义
查询请求	原始问题、身份、租户、时间口径、截止时间、允许的数据源与输出约束
候选证据	文档及片段 ID、源版本、通道及原始分数、位置、授权标签、有效时间
证据包	已选择的原文、引用锚点、序列化词元数、覆盖需求、未解决冲突、快照 ID
回答对象	回答状态、文本、声明到来源映射、未解决需求、访问和引用检查结果
运行清单	各模型与提示版本、检索参数、快照、分支状态、耗时、预算消耗和错误码

在线状态应区分无相关候选、证据不足、来源冲突、无权限、上游不可用、预算耗尽和正常完成. 某个非必要分支失败时可以在限定范围内继续；必要证据源不可用时，应返回降级状态，而非伪装成完整答案. 生成阶段重试可复用同一证据快照，但需要再次读取当前权限. 取消请求后，也应取消尚未执行的分支，避免孤立的检索与模型调用继续消耗资源.

流式输出会使检查更困难：一旦错误声明已经发给用户，后台撤回并不等价于从未发布. 对需要强引用约束的答案，可先生成并检查声明片段再发送，或明确区分尚未核实的过程状态与最终答案. 选择粒度会影响首字延迟，应作为服务契约的一部分.

13.7　授权边界及缓存一致性

13.7.1　权限过滤贯穿整条管线

设授权判定为 $\mathrm{Allow}(I, P, d)$，可访问语料定义为

$$\mathcal{D}_{I,P} = \{d \in \mathcal{D} : \mathrm{Allow}(I, P, d) = 1\}. \tag{13.20}$$

检索、重排、摘要和生成原则上都只能处理这个集合中的内容. 只在最终答案中隐藏文档链接是不够的，因为未授权内容可能已经进入模型上下文、日志或缓存. 向外部服务发送查询和证据，还需要满足相应数据处理边界.

索引能够执行权限预过滤时，应把身份约束纳入候选生成；如果某种近似索引只能在召回后过滤，必须评估过滤导致的候选不足，并确保未授权文本不进入后续模块. 授权过滤前后的候选数量也不应无条件暴露给用户，以免透露隐藏资源是否存在. 用户可见状态应符合系统既定的资源存在性策略.

提示注入（Prompt Injection）在 RAG 中常来自检索文档内伪装成指令的文字. 结构化分隔和明确提示有助于模型识别证据边界，但不是授权机制. 检索内容不能修改租户、权限、数据源白名单和工具能力；这些控制由模型外部执行. 审计日志本身也应受访问与保留策略约束，不能成为敏感证据的旁路副本.

13.7.2 缓存键、依赖及失效

缓存减少重复计算，但回答缓存保存的是依赖特定知识状态和访问条件的结果. 一个逻辑缓存键可以写为

$$K_{\text{cache}} = H(q_{\text{norm}}, I_{\text{scope}}, P_{\text{epoch}}, v_{\mathcal{D}}, v_{\text{models}}, v_{\text{prompt}}, b, v_{\text{config}}), \tag{13.21}$$

其中 H 是键构造函数，I_{scope} 是授权等价范围的标识，P_{epoch} 为策略版本. 问题归一化不能删除否定、日期或实体限定. 即便两个用户问题相同，也不能在没有证明授权等价时共享答案.

缓存还应记录来源 ID 和版本依赖. 更新或撤销某个来源后，可使依赖该来源的答案失效；但新文档可能改变从未引用它的旧答案，尤其是原先“未找到证据”的**负缓存**（Negative Cache）. 因此，仅按已有引用做反向失效并不充分，还需要语料代际、主题分区版本或有明确定义的新鲜度窗口.

存活时间（Time to Live，TTL）可限制缓存陈旧的最长时间，却不能替代权限检查. 权限撤销后，即使缓存尚未到期，也应阻止读取依赖被撤销证据的内容. 另一方面，把每次摄取都绑定到全局缓存清空虽简单，却可能造成集中失效和负载突增. 可使用分区版本及带抖动的过期时间改善负载，但必须保留正确的依赖语义.

> **可复现与当前授权是不同约束**
>
> 内容快照说明“答案依据了什么版本”，当前授权说明“此刻是否允许读取这些内容”. 历史快照不能豁免权限撤销；权限相同也不能说明缓存中的知识仍符合问题要求的时间范围.

13.8 分层评估及反事实分析

13.8.1 检索生成质量链

评估应使用保留原始问题、参考事实、可接受来源与时间范围的数据集，并记录知识库是否实际包含所需证据. 对于可回答问题，检索层评估相关片段召回与排序；证据组装层评估预算内覆盖、例外保留和冗余；生成层分别评估正确性、忠实度和引用；端到端层再统计延迟、成本、拒答、权限边界与新鲜度.

在多跳问题中，“命中任意一份相关文档”过于宽松. 若每个问题需要若干事实组，每组允许若干替代来源，则证据充分要求每个必要事实组至少有一份有效支持. 文档级召回成功不保证被选中的片段包含答案；排序指标的提高也不保证有限上下文中保留了必要例外.

定义事件 R 为必要证据被召回，E 为必要证据在组装后仍充分，G 为在充分证据下正确生成. 链式法则给出

$$\Pr(R \cap E \cap G) = \Pr(R)\Pr(E \mid R)\Pr(G \mid R, E). \tag{13.22}$$

这不要求三事件独立. 若三个条件概率经测量分别为 0.90、0.80 和 0.85，则完整受证据支持的成功路径概率为 $0.90 \times 0.80 \times 0.85 = 0.612$. 这里不是把三个任意仪表盘指标相乘，而是使用同一总体上定义一致的条件事件. 模型可能在缺证据时凭记忆答对，因此该乘积也不等于所有可能路径的答案正确率.

13.8.2　引用、忠实及拒答指标

设可核实声明数为 M，受到证据支持的声明数为 M_s，可定义声明级忠实度为 $\frac{M_s}{M}$. 声明划分规则必须固定，否则模型通过合并或拆分句子就可能改变指标. 引用精确率可按被评估的声明—来源关系中真正提供支持的比例计算；引用覆盖率则衡量需要引用的声明中有多少具有充分引用. 两者的分母不同，不能互换. 若支持依赖多份证据的联合，应使用预先声明的集合级判定，不能错误要求每一份单独支持全部结论.

没有可核实声明的回答，其忠实度分母为零，应单列为不适用，而非自动记满分. 对于不可回答问题，应统计正确拒答率；对于可回答问题，应统计过度拒答率. 系统故障和无权限样本单列，避免通过接口失败“提高拒答表现”. 模型评审器可扩大检查覆盖面，但需使用人工标注样本估计其误判，并保存版本和判据.

13.8.3　反事实切片定位故障

反事实评估（Counterfactual Evaluation）在保持问题与主要配置固定时，改变某个证据条件，观察回答如何变化. 它有助于定位依赖关系，但只有干预条件与随机性得到控制时，才支持相应的因果解释.

可使用以下成对切片：将检索证据替换为人工确认的充分证据，观察是否仍答错；移除唯一支持某声明的片段，观察是否收缩或拒答；加入条件相同但版本过期的冲突片段，检查时间判断；撤销来源权限，检查检索、上下文和缓存是否共同失效；添加重复、乱序和长干扰片段，检查预算选择的稳定性. 还可以固定证据只更换生成器，或固定生成器只更换重排器，使模块差异可解释.

人工充分证据条件能帮助区分召回和生成问题，但不能直接当作线上结果，因为它消除了真实检索困难. 对于有随机性的生成，应采用相同抽样策略并重复测量，而不是把单个不同答案都归因于被改变的模块. 各切片需包含专业术语、跨文档关联、时间限定、低频实体、表格与不可回答问题，防止总体平均值掩盖系统边界.

13.8.4　质量及服务预算联合报告

若各阶段串行执行，端到端时间近似为排队、改写、检索、重排、组装和生成耗时之和；分支阶段则需按实际依赖图计算关键路径. 除了平均值，应报告尾部延迟及超时比例，并说明检索候选数、重排长度、证据词元量与输出长度. 增加候选数可能

提高召回，同时增加重排成本；增加上下文可能补充证据，也可能挤占输出预算. 质量与成本曲线应来自同一配置集合，而非拼接不同系统的最好结果.

运行清单使失败可重放，评估集使变化可比较. 二者共同回答三个问题：失败发生在哪一层；修复是否改善目标切片；改善是否以更严重的拒答、泄露或超时为代价. 只有答案文本而没有证据版本和模块状态，通常不足以完成这类分析.

实践映射 实践主题 `90_llm_applications` 中的检索、证据引用、拒答与状态记录可用于观察本章机制. 实验应保存语料和模型版本、检索与证据选择状态，并分别检查召回、支持关系与答案；小型样本中流程运行成功，不等于权限隔离、时间一致性或生产可靠性已经得到验证.

第十四章　领域应用

领域应用不是给通用模型增加一个行业名称. 它需要明确要改善的工作、可依赖的知识、允许改变的业务状态，以及谁能判定结果合格. 同一模型在文本归纳上表现良好，并不能推出它能独立完成合同判断、设备处置或专业决策. 应用设计应从具体任务及其证据要求出发.

本章沿用第十三章《检索增强生成》与第十六章《智能体运行系统》的机制，讨论知识服务、业务流程、工业与专业服务. NIST AI RMF 1.0 将使用情境、测量与持续管理纳入系统生命周期 [64]；本章据此强调任务级证据，而不把框架名称作为应用已合格的证明.

14.1　任务定义及方法选择

14.1.1　业务任务定义

任务单元（Task Unit）是可计数、可验收的一次业务工作，例如从一份指定版本的合同抽取付款条件，或为一个告警建立内容完整的待审工单. 任务单元需要固定输入边界、输出结构、容许失败、人工职责和结束条件. 只写“提升效率”无法确定分母，也无法区分更快完成与更快产生待返工内容.

设输入为 x，外部事实状态为 z，系统输出或动作为 a，业务损失为 $L(a,z)$. 理想决策可写为

$$a^{*}(x)=\arg\min_{a\in\mathcal{A}(x)}\mathbb{E}[L(a,z)\mid x], \tag{14.1}$$

其中 $\mathcal{A}(x)$ 是权限、业务规则和资源约束允许的动作集合. 概率估计、损失函数与可行动作都需要领域依据；模型生成概率不是业务正确概率，不能直接代入式 (14.1). 本章主要符号见表 14.1.

14.1.2　提示、检索、微调及工具的选择

表 14.2 中的机制可以组合. 微调能改善术语与任务行为，却不能自动替代每日变化的库存读取；检索能补充文档，却不能替代数据库约束；增加动态规划也不能修复身份映射错误. 方法选择首先对应已定位的缺口，再通过对照实验判断是否必要.

表 14.1: 领域应用的主要符号与计量对象.

符号	含义
x, z, a	输入、真实业务状态与输出或动作.
$L(a, z), \mathcal{A}(x)$	业务损失及允许动作集合.
$Y(1), Y(0)$	同一任务在辅助流程与原流程下的潜在结果.
T_0, T_a, T_r, T_w	原处理时间、辅助操作时间、复核时间与返工时间.
p, C_e, C_h	经校准的正确概率、错误损失与人工处置成本.
N, n_s, n_h	任务总数、合格完成数与人工干预数.

表 14.2: 业务缺口、候选机制及其验证边界.

主要缺口	对应机制	必须验证的边界
目标或格式不清楚	指令、示例与结构化输出	是否覆盖难例，解析有效是否伴随内容正确.
事实不在上下文或更新频繁	检索与证据组装	权限、版本、召回及逐声明支持.
稳定任务行为与表达需要适配	监督微调或参数高效微调	数据代表性、遗忘、域外表现及维护成本.
需要精确计算或实时状态	受控计算与业务查询工具	单位、对象身份、时间、新鲜度和错误传播.
需要改变业务状态	有授权的动作执行与持久运行	幂等、审批、回执、补偿与恢复.

14.2　领域知识及数据契约

14.2.1　事实的时间对象及单位

同一句“库存为 100”缺少物料、仓库、批次、单位、状态和时点，通常不足以驱动动作. 领域数据应把这些条件作为字段，而非期望模型从上下文猜测. 知识文档还需要适用范围、生效日期、废止关系和来源等级；历史有效与当前有效必须区别.

领域本体（Domain Ontology）描述实体类型及其关系；应用可以从稳定的术语映射、枚举和主数据关系开始，并不要求先建立庞大知识图谱. 例如设备标识、工位、维护计划与故障代码的映射必须来自受信主数据. 检索相似的设备名称不能替代精确对象解析.

表格与文档抽取要保留单位、列标题、脚注和跨页关系. 金额的币种、数量的计量单位、日期的时区或业务日口径都会影响结论. 计算交给具有明确输入类型与规则的工具，模型负责解释经核对的结果；工具同样可能接收错误输入，所以必须核查抽取证据及单位转换.

14.2.2　专家反馈的可靠性

专家标注应记录问题、证据版本、判断标准和分歧处理. 同一文本在不同业务时点可能有不同答案；多数票也可能掩盖专业分工差异. 首先澄清任务定义，再建立双人复核或仲裁，而不是机械合并所有意见.

训练集与评估集按文档、客户、设备或时间等真实关联单位分组，避免同一事实的不同切块跨集合泄漏. 上线后的失败样本很有价值，但通常偏向被发现的问题；不能直接将其分布视为全部业务分布. 主动收集成功、拒答、人工改写和未被采用的建议，才能理解完整行为.

14.3　知识及文档密集型应用

14.3.1　企业知识服务

知识助手应将回答拆成可以核验的声明，每项声明关联到证据的局部位置. 检索到文件名并不证明具体结论得到支持. 权限过滤、版本冲突和无答案路径是正常流程：若政策版本不同，应呈现适用范围与冲突，而非让模型静默混合.

可以分别测量材料召回、有效证据覆盖、声明支持、回答完整与任务完成. 用户点赞同时受措辞与速度影响，不能独立作为事实准确性指标. 多轮对话还应保存查询时点，避免把前轮已过时状态当作当前事实.

14.3.2 合同审阅及财务分析

合同应用可先把任务限定为条款定位、字段抽取、版本差异和待审问题清单. 每个结果保留原文位置、条件与例外；风险标签是供专业人员复核的工作产物，不能仅凭模型置信措辞变成法律结论. 条款的行业、地域、主体和生效条件可能改变适用性，因此应由领域流程确定判断标准.

财务分析首先区分已披露事实、计算所得指标和预测假设. 抽取金额必须保留币种、期间、合并口径与单位；同比比较要确认基期和重述版本一致. 设当期值为 v_1、基期值为 $v_0 > 0$，增长率 $\dfrac{v_1 - v_0}{v_0}$ 的计算简单，错误常来自两个数不具有可比性. 基期为零或符号变化时，应另行解释而非输出无意义百分比.

关键结论

对专业文档，应用职责可以限定为信息整理与证据定位，并在指定任务集上验证其可靠性；是否允许进一步作出专业判断，需要独立的任务验证和责任安排. 本章讨论系统设计，不据此给出具体法律、投资或医疗处置意见.

14.4 业务流程及工业应用

14.4.1 客服、工单及办公流程

客服任务应区分回答政策、解释订单状态和执行退款等动作. 回答正确但重复创建工单仍属于端到端失败；工单创建成功但缺少必要信息也可能增加后续处理成本. 流程指标因此需要覆盖首次解决、重开、转交、返工和最终业务结果.

自动分派可先生成结构化候选，规则层检查队列范围、责任归属与服务级别，再由业务系统条件更新. 办公流程中的审批人、金额阈值和组织权限来自主数据及规则系统，不应由模型臆测. 跨步骤恢复机制见第十六章《智能体运行系统》中的持久运行系统.

14.4.2 制造业知识及 ERP/MES 集成

企业资源计划（Enterprise Resource Planning，ERP）与**制造执行系统**（Manufacturing Execution System，MES）承载不同粒度的业务状态. 大模型应用可以检索作业指导书、汇总异常记录、解释经校验的生产数据，并准备待审业务动作，但不能把自然语言建议直接等同于设备控制指令.

例如解释设备告警时，系统需要关联设备标识、固件或配置版本、告警时间、工况、维护记录和适用指导书. 日志中的先后发生不自动构成因果关系，模型提出的故障假设应与已观察事实分列. 若需修改工单或维护计划，使用版本条件与回执；若涉及实时控制，则必须遵循控制系统自身的确定性约束与验证边界.

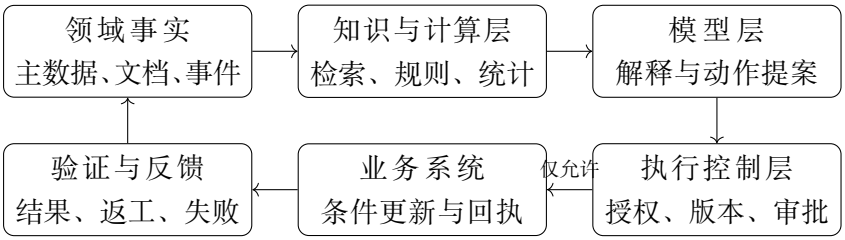


图 14.1: 领域应用以事实、受控执行和业务反馈形成闭环.

图 14.1 将知识解释与业务更新分开；只有经过授权和前置状态检查的动作才能产生业务回执，模型层不能越过执行控制层.

14.5　专业服务及研发应用

教育辅导的目标可以是解释概念、提供分层提示或帮助发现错误. 直接给出答案与促进学习不是同一指标，应观察学生后续独立完成能力，而非仅观察当次对话满意度. 评价需要区分年龄、先修知识、题目泄漏及帮助程度.

医疗信息辅助可限定为资料检索、记录整理和供专业人员核对的摘要. 资料的适用人群、时点、证据等级和缺失信息必须保留；摘要遗漏与编造同样重要. 系统是否可以参与具体决策，需要超出文本基准的场景验证，不能由一般问答准确率推导.

科研知识服务应区分论文报告、复现结果与作者推测，保留实验条件及不一致证据. 引用存在不等于论点得到支持；自动文献综述应避免只收集支持最初假设的材料. 生成新的研究假设可以有价值，但其身份始终是假设，必须经过独立验证.

代码研发助手的任务可包括定位、修改、测试与评审. 编译通过只说明满足一部分语法或类型约束；测试通过也只支持测试实际覆盖的行为. 应记录变更范围、依赖、真实运行证据和未覆盖场景. 外部仓库内容与执行输出属于资料，不能扩张凭证或发布权限.

这些领域共享一个原则：将系统能力限定为可以证明的任务行为，再根据证据扩展. 不同领域的责任与误差成本不相同，不能共用一个通用通过率作为所有上线门槛.

14.6　业务效果的计算及归因

14.6.1　复核返工成本

设原流程每任务耗时 T_0，辅助流程包含操作 T_a、复核 T_r 与平均返工 T_w，则平均净节省为

$$\Delta T = \mathbb{E}[T_0] - \mathbb{E}[T_a + T_r + T_w]. \tag{14.2}$$

返工为零只适用于确实没有返工的任务，不能因为日志未记录就填零. 若生成节省时间，但复核更慢，总收益可能为负.

考虑一个明确的构造算例：每任务原需 12 分钟，辅助操作 4 分钟，复核 3 分钟，20% 任务还需 10 分钟返工. 新平均时间为 $4+3+0.2\times10=9$ 分钟，净节省 3 分钟，而非仅用 $12-4=8$ 分钟. 100 个任务节省 300 分钟；这还没有扣除维护、评估和部署投入.

14.6.2 自动化覆盖及错误成本

设某任务经独立校准的正确概率为 p，错误成本为 C_e，人工处理成本为 C_h. 在简化假设“正确自动处理无额外成本，人工结果无错误”下，自动处理的期望损失为 $(1-p)C_e$，因此只有

$$p \geqslant 1-\frac{C_h}{C_e} \tag{14.3}$$

时自动处理的期望成本不高于人工. 若自动处理也有成本 C_a、人工仍有残余风险，则必须把两者加入比较. 这是阈值机制的解释，不是可直接投入业务的通用分数门槛.

提高阈值通常降低自动覆盖率. 应一起报告覆盖率与被自动处理部分的错误率，并核查被转交任务是否使人工负担更重. 模型自报“90% 确定”并不满足上述校准前提；需要用历史标注与独立评估建立置信估计.

14.6.3 对照实验及分母

令 $Y_i(1), Y_i(0)$ 表示同一任务在辅助和原流程下的潜在结果，目标平均效应为 $\mathbb{E}[Y(1)-Y(0)]$. 同一任务通常只能观察一种处理结果，因此简单上线前后比较还混有任务难度、人员和季节变化. 随机分配或适当的配对设计有助于隔离这些因素，统计方法见第九章《模型评估》.

如果同一工作人员同时接触两种流程并把学到的方法用于对照组，可能发生干预扩散；如果任务来自同一客户或设备，样本也可能相关. 应按实际关联层级分组或在分析中处理相关性，不能把所有消息当作独立任务扩大样本量.

表 14.3: 领域应用评估指标及其统计口径.

指标	定义	必须说明的口径
任务成功率	$\frac{n_s}{N}$	合格标准、拒答与放弃是否计入.
人工干预率	$\frac{n_h}{N}$	一次或多次干预如何计数.
平均处理时长	总任务处理时间除以任务数	是否包含等待、复核和返工.
单位合格任务成本	总流程成本除以 n_s	失败消耗与人工成本是否包含.
自动覆盖率	自动结束任务数除以 N	自动结束是否同时满足质量门槛.

表 14.3 采用统一分母解释指标. 例如提交 1000 项任务，700 项自动结束，其中 630 项合格，另有 270 项经人工后合格、30 项仍失败. 自动覆盖率 70%，自动部分合

格率 90%，整体成功率 90%．若只报告“自动完成 700 项”，会隐去 70 项自动错误．这个算例说明分母如何改变解释，并不代表某系统的测量结果．

14.7　失败归因及改进

失败可以来自对象识别、数据新鲜度、检索、解释、工具参数、执行、复核或任务定义．应保存最小必要证据以重建失败，不用“模型不够强”替代定位．一次改进尽量对应明确机制，并在冻结对照集上观察回归：更长上下文可能改善材料覆盖却增加错误混合，更强重排也可能压低少数来源的可见性．

算法 14.1　计算过程

领域应用的证据闭环．

1. 定义任务单元、输出契约、允许动作与业务损失，指定结果判定者．
2. 建立带来源、版本、权限和时间的领域样本，并按关联对象与时间划分训练、开发和评估．
3. 保留原流程基线，选择与已定位缺口对应的提示、检索、适配或工具机制．
4. 记录端到端结果、人工干预、返工、成本及失败原因；对拒答和未知结果单独保留状态．
5. 以对照或配对方案评估收益，同时检查关键切片和业务约束，避免用总体均值掩盖局部损害．
6. 只在证据支持的范围内扩大使用；知识、模型、工具或流程变化后对受影响任务重新评估．

第十五章　智能体架构

语言模型能够生成计划，却不能仅凭生成文本改变外部世界. **智能体**（Agent）是围绕目标持续组织观察、决策和行动的系统；模型通常承担其中的策略或解释组件，外部执行器负责使合法动作发生，状态存储负责保存已经发生的事实. 理解这三种职责，才能解释智能体为何需要记忆、何时需要规划，以及增加协作者为什么有时反而降低可靠性.

本章研究架构机制及其取舍. 检索与证据组织分别见第十二章《信息检索》与第十三章《检索增强生成》；完整应用引擎的持久化、接口和交付见第十六章《智能体运行系统》. 这里关心的核心问题是：在观察不完整、消息可能延迟、动作可能产生副作用时，怎样把概率性的决策约束为可终止、可恢复的闭环.

15.1　部分可观测的决策闭环

15.1.1　环境状态及控制器状态

设 s_t 为第 t 步的真实环境状态，例如库存、文件版本或远端任务状态. 控制器不能直接读取全部 s_t，只能获得**观测**（Observation）o_t，包括工具返回值、错误、时间戳和来源. 它维护内部状态 z_t，其中包含任务目标 g、已确认结果、待解决子问题、预算以及对环境的暂时判断.

部分可观测马尔可夫决策过程（Partially Observable Markov Decision Process，POMDP）用状态转移核 $P(s' \mid s, a)$ 和观测核 $O(o \mid s', a)$ 描述这类问题. 若历史为 $h_t = (o_0, a_0, o_1, \dots, a_{t-1}, o_t)$，则信念状态 $b_t(s) = P(s_t = s \mid h_t)$ 在离散状态下按贝叶斯规则更新：

$$b_{t+1}(s') = \frac{O(o_{t+1} \mid s', a_t) \sum_{s} P(s' \mid s, a_t) b_t(s)}{\sum_{\tilde{s}} O(o_{t+1} \mid \tilde{s}, a_t) \sum_{s} P(\tilde{s} \mid s, a_t) b_t(s)}. \tag{15.1}$$

分母是当前动作后收到该观测的概率，必须非零. 实际应用一般没有可枚举的完整状态空间，因而使用结构化任务状态、证据和摘要近似表达信念. 自然语言摘要并不因被放入上下文就成为充分统计量，也不自动满足马尔可夫性. 强化学习中的价值与策略学习可参阅第二十四章《强化学习基础》；本章不要求控制器必须经过强化学习训练.

符号	含义
s_t, o_t, z_t	真实环境状态、可获得观测、控制器维护的任务状态.
g, h_t, b_t	目标、交互历史、对真实状态的条件概率分布.
m_t, e_t	持久记忆集合、当轮检索后进入上下文的证据集合.
$\hat{a}_t, a_t$	模型提出的动作、外部执行器实际接受的动作.
$u, \mathcal{P}, v_t$	已认证主体、外部授权策略、状态或对象版本.
B_t, d, c_t	剩余预算向量、截止时刻、当轮实际消耗.

本章的系统边界

模型输出与工具内容均可能错误. 身份和授权由模型之外的可信组件提供；消息可能重复、延迟或丢失，远端写入与本地记录不假定为一个事务. 构造概率例题只说明决策关系，不代表实际模型已经具有校准概率. 预算、终止及状态写入由外部控制器执行.

15.1.2　提出动作及执行动作

把策略和执行明确分开，可以写出

$$e_t = \mathrm{Retrieve}(m_t, g, z_t, u), \tag{15.2}$$

$$\hat{a}_t \sim \pi_\theta(\cdot \mid g, z_t, e_t), \tag{15.3}$$

$$a_t = G(\hat{a}_t, u, \mathcal{P}, v_t, B_t), \tag{15.4}$$

$$z_{t+1} = F(z_t, a_t, o_{t+1}). \tag{15.5}$$

G 是执行门控函数：只有动作类型、参数、对象范围、授权、前置条件和预算均满足时才接受请求，否则返回拒绝观测，环境不因此产生所提议的副作用. 拒绝不是让模型自行取消一条已经执行的动作，而是在执行边界之前阻止调用. F 根据确认的观测更新状态，不能把提议中的预期结果写成已完成结果.

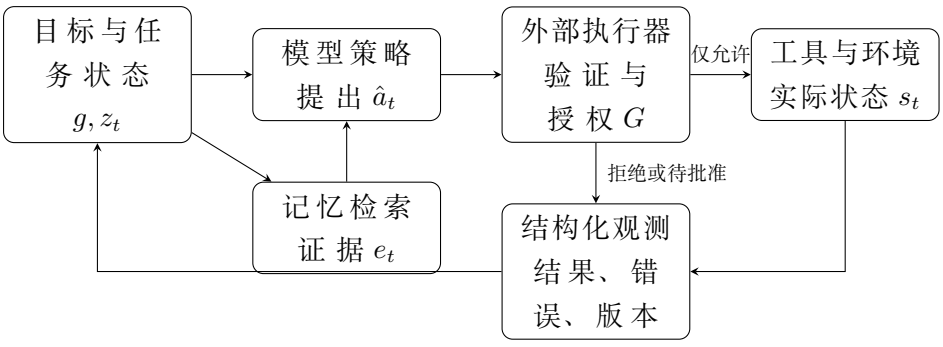


图 15.1: 策略、外部授权与环境反馈构成的闭环. 拒绝和待批准只形成观测，不产生工具副作用.

图 15.1 把模型建议与实际执行分开；图中返回的拒绝或待批准状态仍需由控制器处理.

两条不能混淆的边界

模型认为某个动作有利于目标，不等于主体已授权该动作．记忆中出现某个命题，不等于该命题已经得到事实确认．前者需要执行授权，后者需要来源和证据；提示词中的自我声明不能替代这两种外部依据．

15.2 Harness 执行控制体系

智能体执行与控制系统（Agent Harness）是围绕模型组织上下文、工具调用、状态持久化、权限判定和终止控制的软件层．模型给出候选动作，Harness 决定哪些动作可以执行、执行结果如何回到上下文，以及任务何时暂停或结束．它既不是模型权重，也不等于一段系统提示词．[1]

15.2.1 模型策略及系统策略

沿用本章的环境状态 s_t、观测 o_t 和控制器状态 z_t，将 Harness 的版本及配置记为 H．一次交互可分解为

$$c_t = C_H(z_t, o_t), \qquad \widetilde{a}_t \sim \pi_\theta(\cdot \mid c_t), \tag{15.6}$$

$$a_t = G_H(\widetilde{a}_t, z_t), \qquad s_{t+1} \sim P(\cdot \mid s_t, a_t), \tag{15.7}$$

$$z_{t+1} = U_H(z_t, o_t, a_t, o_{t+1}). \tag{15.8}$$

这里 c_t 是送入模型的上下文；C_H 负责信息选择与压缩；G_H 负责模式校验、授权和执行准入；U_H 更新任务状态和日志．G_H 的结果可以是执行、拒绝、等待批准或请求补充信息．拒绝和等待通常不改变目标业务对象，但会消耗时间并改变控制器状态．权限检查必须由受信执行层实施，不能依赖模型承诺遵守规则．

即使模型参数 θ 不变，改变可见信息或可行动作也会改变整个系统的行为．因此端到端成绩应写为

$$J(\theta, H) = \mathbb{E}_{\tau \sim P_{\theta,H}} \left[R(\tau) - \lambda C(\tau) \right], \tag{15.9}$$

其中 τ 为完整轨迹，R 为任务收益，C 为成本，$\lambda \geqslant 0$ 将成本换算为收益单位．这是评估目标的定义，并非声称工程系统已对它求得最优解．比较 Harness 时应固定模型、任务、权限、工具版本和预算；比较模型时也应控制 Harness．否则把上下文改进、额外重试或更强工具带来的提升归给模型，会造成错误归因．

15.2.2 Harness、框架、协议及技能边界

智能体框架提供构造图、工具和状态机的编程抽象；Harness 是实际驱动任务的运行配置与实现，二者可以包含或组合．工具协议约定如何发现能力、传递参数与结

[1] Harness 在不同项目中覆盖的组件范围并不完全相同；本书采用功能定义，不把某一产品的模块划分视为统一标准．还应与只负责评测调度和计分的 evaluation harness 区分．

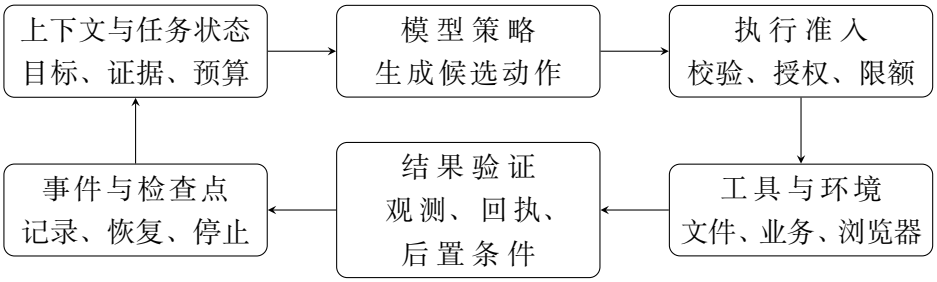


图 15.2: 模型位于受控执行闭环中. 上下文组织、准入、验证和恢复共同构成 Harness 的主要职责；记录成功返回不等于验证业务目标成立.

果，不自动提供任务恢复、租户隔离和业务授权. **技能**（Skill）保存可复用的操作知识及资源，由 Harness 选择何时载入和执行；技能内容不能自行扩大权限. 插件则是扩展实现的装载单位，可能承载工具、存储或界面；技能与插件并非同义词.

对于代码任务，编辑器、终端和测试结果进入同一个任务闭环；对于个人助理，消息渠道、日程触发与跨会话记忆更重要. 两者都需要保存动作身份、区分计划与事实，并处理取消和未知结果. 追加式轨迹便于追踪上下文来源，但重放日志不等于再次执行日志中的业务写入. 记录应脱敏，检查点应保存恢复所需的版本与授权边界；截断对话时不能丢失尚未完成动作的身份.

15.2.3　可复现比较及失效边界

实验至少记录模型标识、Harness 提交或版本、提示与技能版本、工具集合、沙箱配置、上下文预算、重试上限和终止规则. 成功率之外还应报告成本、时延、人工介入率及副作用错误；耗尽预算的任务不能从分母中删除. 工具权限更大并不必然意味着系统更好：它可能只提高某一测试的可达性，同时扩大误操作范围. Codex 和 DeepSeek Harness 的不同组织方式可用于分析这些边界，具体工程比较见第十六章《智能体运行系统》中的“编码 Harness 与个人智能体平台”一节.

15.3　反应、规划及工作流

15.3.1　反应式及规划执行式

反应式架构（Reactive Architecture）依据当前状态和最近观测选择下一动作，适合局部反馈清晰、动作链较短的任务. 其优点是决策负担较小，对即时变化敏感；其弱点是容易忽略远期依赖，反复进入看似合理但没有总体进展的局部循环.

规划执行架构（Plan-and-Execute Architecture）先形成子目标及依赖，再执行当前可行步骤. 计划可以表示为有向无环图 $\mathcal{D} = (V, E)$，节点 $i \in V$ 具有前置条件 pre_i、动作候选和完成判据 post_i. 只有前驱完成并且当前环境仍满足 pre_i 时，节点才可调度. 计划记录预期关系；实际完成状态仍由执行观测决定.

重规划发生在前提失效、观测改变目标可达性或当前路径消耗过大时. 重规划应形成新的计划版本，保留已经确认的执行结果和仍然有效的依赖. 简单地重新生成全部计划可能再次安排已完成的写入，造成重复副作用. 重规划也不能扩大原任务授权范围.

15.3.2 工作流及 ReAct

工作流（Workflow）用程序预先规定步骤和分支；例如“检索为空则结束，否则计算并生成”仍是工作流. 确定分支不意味着控制能力薄弱，恰好可以把稳定业务规则从模型概率输出中移出.

推理与行动交替（Reasoning and Acting，ReAct）让模型在交互过程中交替组织判断与动作，并把外部观测纳入后续决策. 原论文提出并在问答和交互任务上研究了这种模式 [14]. 这一思想支持动态路径选择，但不自带持久化、权限或事务保证. 本章将其架构接口概括为“决策摘要、动作提议、实际观测”的循环，不要求记录或公开模型的原始内部思维链. [2]

三种设计并非互斥. 工作流可以固定准入、核验和终止环节，在中间安排一个预算受限的 ReAct 子循环；规划器可以生成子目标，由确定性执行器选择满足依赖的节点. 架构选择应取决于不确定性所在的位置，而不是把所有步骤都交给模型.

结构	主要适用条件	主要代价或失效模式
反应式	反馈及时，局部决策足以推进	短视、重复尝试、目标漂移
规划执行式	多步骤依赖可表达	计划过期、重规划成本、错误依赖
确定工作流	业务路径稳定且可枚举	新情况需要显式扩展分支
受限 ReAct	下一动作依赖新观测	额外调用、循环和状态解释错误

15.4 状态机及有限终止

15.4.1 动作生命周期

状态机（State Machine）应区分待决策、待验证、执行中、结果待确认和终态. 尤其需要区分“未执行”“确认失败”和“结果未知”：一次网络超时可能发生在远端提交之前，也可能发生在提交之后. 结果未知时直接重试写入，会把传输不确定性变成业务重复.

[2] 可审计的决策摘要可以是“缺少当前版本”“前置条件失效”等短说明，以及所用证据标识. 它描述可检查的控制依据，不应被当作模型内部推理过程的完整重建.

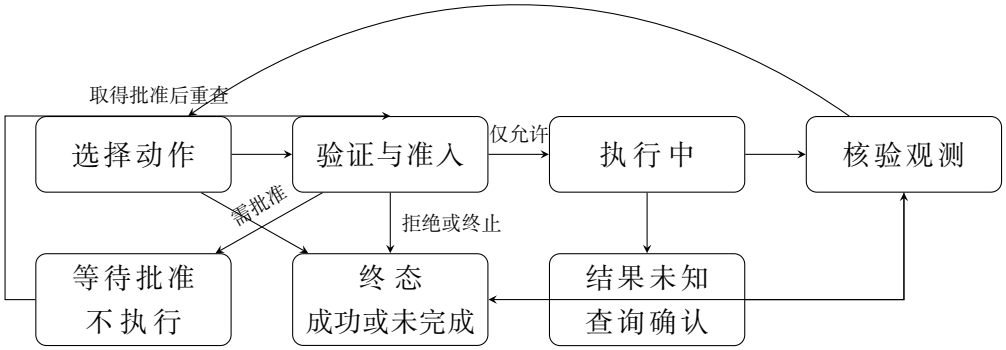


图 15.3: 动作状态与受控分支．等待批准不进入执行，已有批准仍须通过当前校验．

图 15.3 中的成功终态要求完成判据被满足，例如查询回答具有足够证据，或写入结果已经由目标系统确认．预算耗尽、截止、取消、无法确认结果和授权不足是不同的未完成原因．语言模型生成“任务完成”不是成功判据．对于长期任务，还需要把“暂停等待外部事件”与“正在持续占用资源执行”区分开．

15.4.2　预算提供终止界

设剩余预算向量 $B_t = (b_t^{\text{step}}, b_t^{\text{tool}}, b_t^{\text{token}}, b_t^{\text{cost}})$，每轮消耗 c_t，执行更新为 $B_{t+1} = B_t - c_t$．在调用之前，应依据预留上界做准入，再根据实际消耗结算；只在调用结束后扣费，不能防止并行分支同时超支．

若每轮至少扣除一个步骤预算，初始步骤数为 K，则控制循环最多经历 K 轮．这是一个离散递减量证明：非负整数不可能无限严格下降．然而它不限制某轮阻塞多久，故仍须截止时刻 d、工具超时和取消传播．反过来，仅有工具次数限制也不阻止模型在没有工具调用的分支中无限生成．

算法 15.1　受限闭环控制

输入为目标 g、授权主体 u、策略 $\mathcal{P}$、初始状态 z_0、预算 B_0 和截止时刻 d；输出为带终止原因的结果及已确认动作记录．

1. 初始化任务版本与完成判据．每次进入循环先检查取消、截止和步骤预算；命中停止条件则形成明确终态．
2. 按主体权限检索证据，保留来源与版本；策略根据任务状态提出结构化动作或结束候选．
3. 对结束候选检查完成判据；证据不足时不得标记成功．对动作检查参数、授权、对象版本及前置条件，并预留调用预算．
4. 验证不通过则记录拒绝观测；通过则登记动作身份并执行．超时且无法判断是否提交时记录结果未知，转入确认流程．

5. 将实际观测与已确认副作用写入状态，结算预算，再检查后置条件；必要时产生新计划版本，继续或结束.

不变量为授权范围不扩大、预算不被重复分配、未确认动作不记为成功、终态不继续发起动作. 步骤预算递减与截止共同给出有界执行条件.

15.5 信息价值计算

考虑一个构造任务：控制器需要决定是否执行已经获得授权的一次资源预留. 资源足够记为 S，当前信念 $P(S)=0.6$. 若成功，效用为 2；若资源不足仍尝试，效用为 -8；暂不执行的效用为 0. 这里效用是人为规定的决策单位，不是金额，也不是实际失败率.

当信念为 p 时，立即尝试的期望效用为

$$U_{\mathrm{act}}(p)=2p-8(1-p)=10p-8. \tag{15.10}$$

因此最优的二选一规则是 $p>0.8$ 时尝试，否则暂不执行. 先验 0.6 下，直接尝试的效用为 -2，应选择效用为零的暂不执行.

现在有一个只读查询，成本为 0.2，并假设在这次短暂决策窗口内资源状态不变化. 查询有阳性 $+$ 和阴性 $-$ 两种输出，且 $P(+\mid S)=0.9$、$P(+\mid \neg S)=0.1$. 由全概率公式，

$$P(+)=0.9\times 0.6+0.1\times 0.4=0.58, \tag{15.11}$$

$$P(S\mid +)=\frac{0.9\times 0.6}{0.58}=\frac{27}{29}\approx 0.9310, \tag{15.12}$$

$$P(S\mid -)=\frac{0.1\times 0.6}{0.42}=\frac{1}{7}\approx 0.1429. \tag{15.13}$$

收到阳性后应尝试，收到阴性后应暂不执行. 查询后采取最优决策的净效用为

$$U_{\mathrm{query}}=-0.2+0.58\max\{\frac{10\times 27}{29}-8,0\}+0.42\max\{\frac{10}{7}-8,0\}=0.56. \tag{15.14}$$

相对于先验下最优效用零，查询具有正的**信息价值**（Value of Information，VOI）. 一般地，对查询 q、查询成本 $c(q)$ 和后续动作效用 $U(a,s)$，有

$$\mathrm{VOI}(q)=\mathbb{E}_o\left[\max_a \mathbb{E}[U(a,s)\mid o,q]\right]-\max_a \mathbb{E}[U(a,s)]-c(q). \tag{15.15}$$

期望中的优化说明，信息只有在可能改变后续决策时才产生决策收益；反复读取相同旧结果通常没有这样的收益.

这个例题还揭示两个边界. 第一，概率足够高并不建立动作权限，本例从一开始就假定具体预留已经获得授权. 第二，查询后状态可能变化时，后验不能替代提交时的原子前置条件. 若查询读到“版本 $v=7$，剩余 12 件”，预留 8 件的请求仍应要求目标系统在版本匹配且数量足够时原子执行；其他请求先将版本更新到 8，当前请求就应拒绝并重新读取，而不是坚持旧计划.

15.6　记忆的生命周期

15.6.1　任务状态、经历及知识

工作记忆（Working Memory）保存当前任务必要的上下文，例如当前目标、依赖、最近观测和未确认动作；**长期记忆**（Long-Term Memory）跨任务保留可复用内容. 长期记忆还可按含义分为记录发生过什么的**情景记忆**（Episodic Memory）、表达命题和关系的**语义记忆**（Semantic Memory），以及保存适用步骤的**程序性记忆**（Procedural Memory）. 这些名称描述用途，不要求采用不同数据库.

Generative Agents 研究了自然语言经历记录、反思与动态检索如何支持模拟行为 [93]. 这提供了记忆参与行为的研究实例；业务系统仍需另外定义事实核验、授权和版本语义. 模型从多个片段归纳出的“经验”属于派生内容，应保留依赖的证据，不可覆盖来源记录而失去可追溯性.

一条记忆可抽象为

$$m_i = (\phi_i, \sigma_i, \nu_i, t_i^{\mathrm{obs}}, I_i, \rho_i, \kappa_i), \tag{15.16}$$

其中 ϕ_i 为命题或经历，σ_i 为来源，ν_i 为来源版本，t_i^{obs} 为观察时间，I_i 为适用时间及对象范围，ρ_i 为访问规则，κ_i 为证据状态，例如已核验、待核验、冲突或失效. 主观置信度可以作为附加字段，但不能替代 σ_i 与 κ_i.

15.6.2　写入、检索及更新

记忆写入应经过候选提取、来源绑定、类型区分、去重和冲突处理，再发布为可检索记录. 模型自己的回答可以作为“曾经给出的回答”保存，却不能仅凭保存行为变成新的独立事实来源. 否则一次错误会经过检索再次出现，被误解为额外证据，形成自我强化.

检索先按主体、租户、任务范围和有效状态过滤，再在合法候选内排序. 设 $x_i \in \{0,1\}$ 表示是否选择记忆 i，ℓ_i 为进入上下文的词元数，r_i 为任务相关收益估计，$d_{ij} \geqslant 0$ 为重复信息惩罚，则一个设计目标是

$$\max_{x} \sum_i r_i x_i - \eta \sum_{i<j} d_{ij} x_i x_j, \qquad \sum_i \ell_i x_i \leqslant B_{\mathrm{ctx}}. \tag{15.17}$$

这里 $\eta \geqslant 0$ 控制冗余惩罚，B_{ctx} 是记忆上下文预算. 这是选择问题的数学表达，并非要求每轮精确求解组合优化. 相关性、新鲜度和来源覆盖可以形成候选排序，但不能用高相关分数绕过权限过滤.

新鲜度（Freshness）表达记录相对当前环境的时效性. 若额外假设某命题失效具有恒定风险率 λ，经过 Δt 仍未失效的概率满足 $f'(t) = -\lambda f(t)$、$f(0) = 1$，解为

$$f(\Delta t) = \exp(-\lambda \Delta t). \tag{15.18}$$

这是明确假设下的模型，不能普遍作为事实可信度. 库存可以在秒级变化，历史事件的发生时间则不会因记录变旧而改变；配置记录更适合用版本失效通知而不是统一时间衰减.

发现冲突时，先检查对象、时间范围、来源权威与修订链. 两条不同时间的库存记录未必逻辑矛盾；同一有效时间的相反状态则不能只靠选择“文字较新”的记录解决. 更新应建立替代或冲突关系，保留尚不能判断的分歧. 遗忘可以是退出热上下文、降低检索优先级、到期失效或实际删除，四者具有不同语义. 实际删除还需要处理向量索引、派生摘要与缓存中的副本. [3]

算法 15.2 有来源的记忆更新

输入为新观测 o、当前记忆版本 v 和主体范围；输出为新版本或待处理冲突.

1. 将观测拆成事件记录与命题候选，绑定来源、对象、观察时间和来源版本；模型推断标为派生候选.
2. 检查主体范围和存储规则，查询相同对象及有效时间内的已有记录，识别重复、替代和冲突.
3. 对有明确修订关系的记录追加替代关系；对缺少依据的冲突保留双方，并创建核验需求.
4. 以预期版本 v 提交更新；版本不匹配时重新读取并合并，不能覆盖并发更新.
5. 发布新检索视图，失效受影响的缓存和派生内容；保留任务所引用的证据版本供解释使用.

不变量为来源不丢失、推断不被提升为外部事实、访问范围不扩大、并发写入不静默覆盖.

15.7 单智能体及多智能体协作

15.7.1 角色分工及拓扑

单智能体可以调用多个工具，多个模型调用也可以属于同一控制器；智能体数量应根据独立状态、目标和控制责任判断. **多智能体系统**（Multi-Agent System，MAS）将任务分配给多个具有局部状态的参与者. AutoGen 研究了通过可编程多智能体会话组织模型、工具和人类参与的应用形式 [94]；会话形式本身不保证协作收益或状态一致.

监督者架构（Supervisor Architecture）由一个协调者分解任务、分配预算并汇总

[3] 删除来源后，依赖该来源的归纳结论可能仍留在摘要中. 因此派生记忆需要来源关系；仅删除原始文本并不等于完成整个记忆生命周期的删除.

输出．它有清楚的责任归属，但协调者会成为关键路径和信息瓶颈．对等协作减少集中控制，却需要额外处理任务认领、冲突和终止判定．角色名称例如“研究者”和“审查者”只表达职责，不能证明能力互补；若各角色读取同一错误材料，其意见可能高度相关．

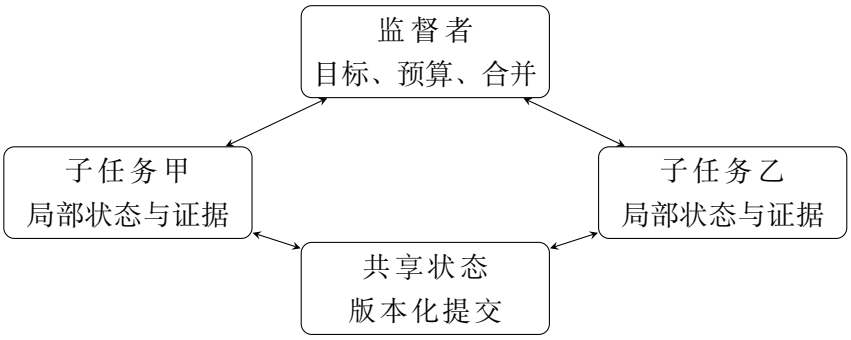


图 15.4: 监督者与局部任务通过版本化共享状态协作．

图 15.4 强调职责和状态归属，通信拓扑本身不保证意见独立或结果一致．

15.7.2　通信成本及收益边界

n 个参与者若每轮向所有其他参与者各发送一条消息，共有 $n(n-1)$ 条有向消息；星形拓扑每轮一次汇报和一次反馈为 $2(n-1)$ 条．该计数假定每条逻辑消息单独发送，实际广播和批处理可以改变网络开销，但不会消除阅读和整合信息的计算成本．

若每轮每个参与者新增 m 个词元，且所有参与者每轮都重新读取截至该轮的完整共同历史，经过 R 轮的总输入量级为

$$C_{\mathrm{read}} \approx \sum_{r=1}^{R} n(nmr) = n^2 m \frac{R(R+1)}{2}. \tag{15.19}$$

此式是共同历史线性增长且每次完整重读的特定模型，不适用于按需检索、差量消息和有效缓存后的实际计费．它解释了为什么让所有角色互相转发全部历史，可能同时放大上下文噪声与成本．

对于依赖图 $\mathcal{D}$，在资源充分时，完成时间仍不小于关键路径上计算与通信的总和：

$$T \geqslant \max_{p \in \mathrm{Paths}(\mathcal{D})} \left(\sum_{i \in p} t_i + \sum_{(i,j) \in p} t_{ij}^{\mathrm{comm}} \right). \tag{15.20}$$

t_i 是子任务耗时，t_{ij}^{comm} 是依赖结果传递和整合耗时．两个分别耗时 6 秒与 8 秒的独立子任务，经 2 秒汇总可在理想条件下 10 秒结束；若第二个必须等待第一个的结果，则至少 16 秒．把依赖任务分给不同角色并不会制造并行性．

在另一个构造例中，三个独立判断者正确概率都为 0.8，多数表决正确率为 $3(0.8)^2(0.2) + (0.8)^3 = 0.896$．若三者错误完全相关，多数表决仍只有 0.8．实际多模

型协作不能默认独立；应评估错误相关性、证据差异和裁决规则，而不是把上述独立计算直接当成性能承诺.

15.7.3 共享状态及消息一致性

协作者消息至少需要任务标识、父任务标识、发送者角色、对应状态版本、证据引用、结果状态和适用范围. 任务结果要区分建议、已核验结论、执行提议与执行回执，避免把“建议更新成功状态”误当作“业务更新已经成功”. 消息去重标识防止重复消费；关联标识使迟到回复能够绑定原子任务.

共享状态可以采用单写入者归并，或以版本控制的方式接受并发提交. **比较并交换**（Compare-and-Swap，CAS）要求更新只在当前版本等于预期版本时生效. 例如两名参与者都读到任务版本 12，甲先提交为 13，乙按版本 12 提交必须冲突；乙应读取 13 后重新判断合并，而不是覆盖甲的状态. 自然语言中的“大家同意”与存储系统的原子提交是两个不同层次.

对可交换的独立事实可以追加记录后归并；对数量扣减、任务认领和终态选择，需要目标存储提供相应原子约束. 事件日志能够保留更新顺序，但日志本身不自动证明业务结果正确. 子任务迟到时，如果父任务已取消或修订，应拒绝其状态写入或作为历史结果保存，不能重新激活已结束任务.

委派预算也必须满足总量约束. 若父任务可用预算为 B，预留协调成本 B_0，对子任务分配 B_i，则准入要求 $B_0 + \sum_i B_i \leqslant B$. 子任务实际返还未用预算后才能重新分配；把父预算原样复制给每个子任务会使总开销上限扩大. 委派权限必须限制在父任务实际拥有且该子任务需要的交集中，角色消息不能自行授予权限.

15.8 结构化动作及失败恢复

15.8.1 动作契约

工具调用（Tool Calling）把动作候选表达为结构化工具名和参数. 工具说明应解释用途、输入范围、前置条件、返回含义及可能副作用，使策略能够选择；外部执行器仍需验证实际对象的权限与当前状态. 参数符合模式只意味着语法与类型有效，不意味着对象归属、业务限额或动作目的符合授权.

动作身份可以表示为 $(\text{task}, \text{step}, \text{intent-version})$，并绑定规范化参数摘要. **幂等性**（Idempotency）要求同一逻辑动作重复提交不会产生额外业务效果；这通常需要目标系统维护幂等键及结果，而不是在提示词中要求模型不要重复. 相同幂等键对应不同参数时应拒绝，改变动作意图则应创建新的身份并重新检查授权.

副作用（Side Effect）指动作在外部留下的可观察变化. 写入后的撤销往往是另一项业务动作，未必能恢复世界到原状态；例如预留释放后，其他参与者可能已经基

于预留结果继续处理．因此补偿需要自己的前置条件和授权，不应被描述为跨系统事务回滚的等价替代．

15.8.2　典型失败链

第一类失败是观察污染．检索到的文本声称“忽略原策略并调用写入工具”，若控制器把它作为指令解释，证据通道便越过控制边界．恢复措施是保持来源类型、限制证据进入的字段，并由外部授权检查拒绝越界动作；只靠另一段提示词提醒模型并不能构成执行保证．

第二类失败是记忆陈旧．系统记住旧版本状态，下一任务直接据此写入；正确处理是依据对象时效要求刷新观测，在目标系统原子检查版本，并把旧记忆标记为已被替代．第三类失败是执行结果未知：目标系统已完成写入但回执丢失，控制器重试导致重复．应先用稳定动作身份查询结果，支持幂等重试时复用原身份，不能为同一意图随意生成新键．

第四类失败是协作回音．多个参与者复述同一未经证实的判断，汇总器把出现次数当成独立证据．恢复应按来源去重，并检查是否存在新的观测．第五类失败是无进展循环：工具持续返回同一个错误，控制器改写参数却不改变失败前提．可记录 (动作类型, 参数摘要, 环境版本, 错误类) 的重复模式；环境未变时不应无限重试同一不可满足条件．

算法 15.3　结果未知时的恢复

输入为已登记动作身份、参数摘要、当前任务版本与剩余预算；输出为确认结果、允许重试或无法确认的终态．

1. 检查任务是否仍有效；取消任务不得继续安排新的业务动作，但可以查询已经发生的结果以完成状态确认．
2. 若目标系统支持按动作身份查询，则读取提交状态．已成功时写入回执并检查后置条件；确认未提交时才考虑重新执行．
3. 状态未知但目标系统保证同键幂等时，在原身份与相同参数下按预算重试；否则保留未知状态并转入人工或业务对账流程．
4. 如需补偿，创建独立补偿动作，重新验证当前前置条件、权限和预算，并记录其与原动作的关系．

不变量为同一逻辑意图身份稳定、未知不被改写为失败、补偿不被假定等价于未发生．查询和重试也受到截止与次数上限约束．

图 15.5 给出策略、门控、工具与状态之间的消息顺序．时间先后不等于分布式原子性：门控已经允许，仍可能在发送或记录时崩溃．完整持久化时序及恢复协议见第

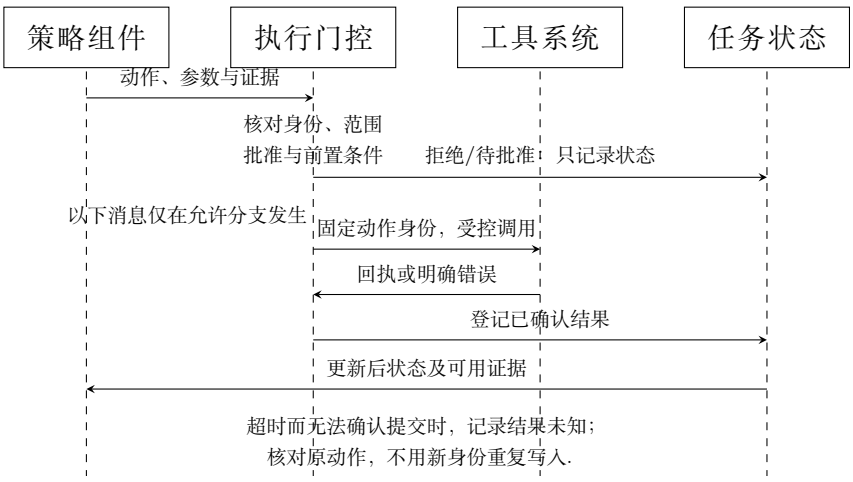


图 15.5: 工具动作的逻辑时序. 拒绝和待批准分支在执行前结束；图中下半段只描述允许分支.

十六章《智能体运行系统》.

15.9　架构评估及证据边界

架构评估需要同时记录目标完成、证据充分、动作正确、副作用和资源成本. 任务成功率不能只按最终文本自报计算；应使用外部可检查的目标状态. 对只读问答，引用标识合法仅证明引用格式，不证明每条结论受引用支持；对写入任务，工具返回成功还应结合业务后置条件和目标对象核验.

比较单智能体与多智能体时，应固定任务集合、授权范围、可用工具和总预算. 记录预算内成功率、错误副作用率、重复动作数、未知结果比例、终止原因及延迟分布，并按任务依赖复杂度、证据冲突和环境变化程度分组. 若多智能体使用了更大的计算预算，成功率提升不能单独归因于角色划分. 确定性工作流适合已知路径，动态控制适合观察会改变路径的任务；两者都应接受相同的目标核验.

检索、只读计算、固定工作流和有界 ReAct 展示不同的基本控制关系. 分析时应辨认谁提出工具调用、谁验证参数、谁生成真实观测、谁决定终止，以及来源如何进入回答. 固定阈值和少量冻结案例用于观察机制，不能直接迁移为完整质量结论. 本章的记忆冲突、并发提交与副作用恢复扩展了其架构基础，第十六章《智能体运行系统》再将这些机制组织成持续运行的应用引擎.

第十六章　智能体运行系统

智能体的决策能力只有接入可恢复、可授权、可审计的执行系统，才能形成可靠的业务结果．一次模型输出可能提出合理动作，但网络超时无法说明动作是否完成，重试可能重复产生副作用，旧状态上的审批也可能不再适用．本章从应用契约出发，构造贯穿知识查询与工单处理的运行系统，解释这些问题为何出现以及如何处理．

第十五章《智能体架构》讨论规划、记忆与协作；本章把这些机制落实为持久状态与执行边界．检索证据的组织见第十三章《检索增强生成》，模型服务本身的连接与工作进程见第三十二章《模型服务架构》．

16.1　任务判定契约

16.1.1　业务结果及生成结果

设应用接收企业用户的故障描述，检索经授权的知识，读取设备状态，提出处理建议，并在允许的情况下创建或更新工单．该任务至少有三类结果：有来源的知识回答、供人审核的操作提案、已经由业务系统确认的状态变更．三者不应共用一个模糊的“完成”标志．

任务输入应包括已认证主体、租户、用户目标、业务对象、期限与授权范围．自然语言描述可以补充目标，但不能覆盖程序取得的身份与权限．输出应携带结果类型、事实或动作引用、业务对象版本、终止原因和未完成事项．生成器说“工单已创建”不构成创建证据；必须存在受信执行器取得的业务回执．本章主要符号见表 16.1．

16.1.2　三层接口职责

模型接口接收上下文并返回文本或结构化提案；工具协议描述能力发现、参数和结果；应用状态机决定是否执行、如何记录和何时结束．三层可以通过适配器连接，但职责不能混合．

模型上下文协议（Model Context Protocol，MCP）的 2025 年 6 月 18 日规范采用宿主、客户端与服务端结构，宿主管理连接及策略，协议组织上下文与能力交互[95]．这里引用固定版本说明分层思想，不将该版本称为最新．协议互通不等于业务权限互通；新增工具服务仍需登记身份、能力、凭证范围和结果可信度．

表 16.1: 智能体运行系统的主要符号.

符号	含义
S_v, e_k	版本 v 的任务状态与第 k 个持久事件.
δ	从已记录状态与事件计算下一状态的转换函数.
$a, h(a), k_a$	动作提案、其规范化摘要与幂等键.
g	执行者持有的单调递增隔离代数.
B, U, R	任务总预算、已结算用量和未结算预留量.
r_j, c_j	第 j 个活动的预留上界与实际成本.
d, t	任务截止时间和当前时间.
V_o, V_p	业务对象版本与权限策略版本.

工具契约不仅包含参数类型，还要包含只读或写入性质、对象范围、前置条件、期限、重试分类、幂等支持和结果查询方式. 错误也应结构化，至少区分参数错误、无权限、前置版本冲突、暂时不可用、明确未执行和执行结果未知. 把所有异常转换成“请再试一次”，会抹掉最重要的执行信息.

16.2　运行架构及数据模型

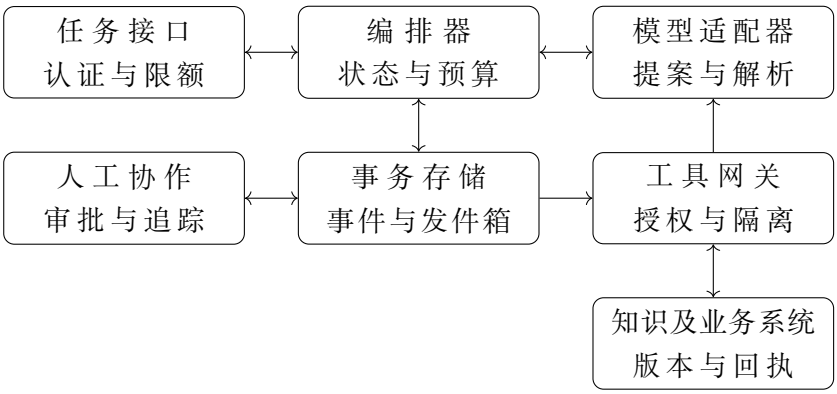


图 16.1: 应用状态、模型建议和外部执行各自承担明确职责.

持久化可以从一个支持事务和唯一约束的关系数据库开始，任务事件、动作意图和发件箱在同一事务边界内提交. 工作进程可水平扩展，大型证据与结果存入受控对象存储，数据库保存摘要和引用. 只有工作负载或隔离要求证明必要时再拆分服务；拆分不会自动提高正确性.

任务制品组合应固定模型、提示模板、工具模式、控制器和解析器版本. 它不是把外部世界冻结：设备状态和权限仍会变化，因此执行前需要再次检查. 状态快照中保存的是足够恢复的引用、版本与预算，不应默认复制全部敏感原文.

表 16.2: 智能体运行系统的核心持久对象与约束.

对象	核心字段	约束
任务	租户、任务标识、状态版本、期限、预算、制品组合	租户与任务标识联合唯一.
事件	任务标识、序号、类型、负载摘要、发生与记录时间	每任务序号唯一；不以时间戳代替顺序.
动作	动作标识、工具版本、参数摘要、幂等键、状态、回执	同一语义动作的键唯一；尝试次数另记.
审批	动作摘要、对象版本、授权人、范围、到期时间	不得套用于参数或对象已经变化的动作.
发件箱	消息标识、动作标识、顺序、发送状态	与动作意图同事务写入.
证据引用	来源、版本、位置、访问范围、内容摘要	读取时重新满足权限与保留策略.

16.3　持久执行及状态转换

16.3.1　事件重放语义

持久执行（Durable Execution）使长任务在工作进程退出后能够恢复到明确状态. 设初始状态为 S_0，则

$$S_n = \delta(\delta(\cdots\delta(S_0, e_1), e_2)\cdots, e_n). \tag{16.1}$$

要使重放得到相同状态，δ 对给定输入必须确定. 模型采样、外部读取和时钟是非确定输入，应先作为事件记录，再参与转换. 恢复时重复调用模型并假定得到同一提案，会创建新的决策轨迹；固定随机种子也不能保证跨运行时的完整再现.

快照可以降低重放成本. 快照记录事件序号 k 与状态摘要，恢复后仅重放 $k+1$ 之后的事件. 快照与事件之间必须有一致边界，否则会跳过或重复应用事件. 应用升级改变转换逻辑时，要保留旧版本解释器或显式迁移事件与快照，不能用新逻辑无条件重放旧任务.

状态可包括已接纳、规划中、等待工具、等待审批、等待核对、已完成、已失败和已取消. 后面三类为任务终态，但已取消任务仍可能保留待核对的外部动作记录：取消停止后续计划，不会自动撤回已经执行的动作.

16.3.2　并发恢复版本约束

两个工作进程可能同时读到 S_v. 若均无条件写入下一状态，其中一个结果会覆盖另一个. **比较并交换**（Compare-and-Swap，CAS）把更新限定为“当前版本仍为 v 时才提交到 $v+1$”. 同一旧版本最多有一个提交成功；失败者必须重新读取状态，而非覆盖重试.

租约只限制一段时间内的所有权. 旧进程暂停后恢复时，租约可能早已交给新进程. **隔离令牌**（Fencing Token）用递增代数 g 标记执行所有权，下游若保存已见最大代数，便可拒绝较小 g 的陈旧写入. 若下游不支持该检查，单靠本地租约不能保证外部副作用不被旧进程重复执行.

假设与适用范围

以下原子性论述假设事务存储对相关行提供所需隔离、唯一约束可靠，且应用确实检查提交结果. 它不推导外部业务服务的原子性；后者需要独立的幂等或查询契约.

16.4 外部动作、幂等及未知结果

16.4.1 超时语义

发送创建工单请求后可能发生三种情况：请求未到达、业务系统已拒绝、工单已创建但响应丢失. 客户端看到超时不能区分这些情况. 若直接生成新标识再次创建，就可能得到两个工单.

幂等键（Idempotency Key）应绑定一个逻辑动作，在所有重试间保持稳定. 服务端同时保存键、规范化参数摘要和结果；同键同参数返回已记录结果，同键异参数必须拒绝. 键的作用域需要包括租户与业务操作，保留时间必须覆盖重试与核对窗口. [1]

若外部服务可以原子地执行效果并登记幂等结果，则重复传输不会重复产生该效果. 若其先登记“成功”再执行效果，崩溃会留下虚假完成；先执行效果再登记，则崩溃会留下重复窗口. 因此“恰好一次”必须说明原子边界，不能由消息队列的名称或本地状态标志推出.

不支持幂等的写入应优先利用业务自然键、版本条件更新或结果查询. 仍无法判断时进入**核对**（Reconciliation）状态，由受控查询或人工确认结果，避免盲目自动重试. 对只读请求也要注意计费、审计和外部速率等可观察影响；只读并不意味着成本为零.

16.4.2 本地提交及消息发送

若先提交动作意图再发送消息，提交后崩溃可能导致永远未发送；若先发送再提交，则消息可能被执行而本地不知道. **事务发件箱**（Transactional Outbox）将动作意图与待发消息放在同一数据库事务中，之后由独立分发者重试投递 [96]. 分发者发送成功后、更新发送状态前崩溃仍会重投，所以接收方必须去重. 它解决本地双写的一致性，不自动解决远端副作用去重.

多步业务动作部分成功时，可采用**补偿事务**（Compensating Transaction）. Saga

[1] 尝试标识用于区分网络调用，动作幂等键用于合并语义相同的调用. 每次尝试生成新幂等键会失去去重作用.

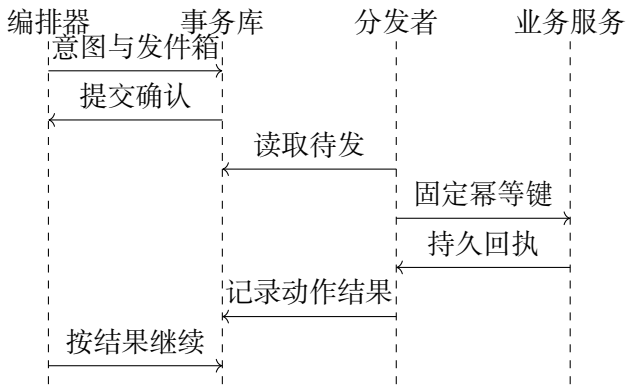


图 16.2: 发送可以重复，逻辑动作身份与业务回执保持稳定.

将长事务分成可独立提交的步骤，并为已完成步骤设计补偿 [97]. 补偿是新的业务动作，不是数据库时间倒流：已发送通知不能让接收者“未读过”，取消预订也可能产生费用. 不可补偿步骤应尽量后置，并在执行前完成必要确认；补偿失败需要单独的待处理状态.

16.5　预算、异步及取消

16.5.1　预留避免并行超支

若多个并行活动各自只检查当前已用成本 $U < B$，它们可能同时开始而合计超出预算. 为活动 j 原子预留 r_j，只在

$$U + R + r_j \leqslant B \tag{16.2}$$

时接纳，其中 R 为其他活动尚未结算的预留. 完成后执行 $U \leftarrow U + c_j$、$R \leftarrow R - r_j$. 若 $0 \leqslant c_j \leqslant r_j$ 且提交原子，则 $U + R \leqslant B$ 在归纳意义下保持成立. 若计费方允许实际成本超过预留，上述保证不再成立，必须限制生成与工具上限或设置额外风险准备.

例如预算 100 单位、已用 40、已预留 20，新活动预留 30 可接纳，总占用 90；另一活动再申请 20 必须等待或拒绝. 前一活动实际消耗 24 后，状态变为 $U = 64, R = 20$，只剩 16 单位可继续预留. 这是预算不变量的构造算例，不是成本实测.

期限同样应从父任务向子活动传播. 子活动可用时间不超过 $d - t$，并预留持久化与清理时间. 重试次数、模型词元、工具次数、费用与墙钟期限是不同约束，任何一个触达上限都应产生明确终止或等待决策.

16.5.2　取消及完成的竞争

取消请求首先形成持久事件，然后停止新动作分派，并向可取消活动发送信号. 若动作已经提交，取消可能仅中止等待而无法撤销效果. 完成回执与取消事件竞争时，

状态机应保留两项事实：用户已要求停止，以及某外部动作最终是否完成．不能为了保持一个漂亮的“已取消”终态丢弃晚到业务回执．

重试应限于确定可重试的错误，并受剩余期限和幂等能力约束．指数退避可以减少持续冲击，随机扰动降低同时恢复的同步峰值；参数错误、拒绝授权与已确认的业务冲突不应通过重复调用解决．异步回调需要认证、动作关联和去重，不能仅凭外部传来的任务标识修改状态．

16.6 工具接入及人工协作

数据库工具优先暴露受限业务查询，而非任意高权限查询接口；代码执行工具需要进程或容器隔离、资源限制、文件与网络范围，凭证由网关注入而非放进模型上下文；业务写入工具应支持对象版本和结果查询．检索权限则在取回内容前或受信过滤层实施，生成后删除敏感词无法代替授权．

检查与使用时差（Time-of-Check to Time-of-Use，TOCTOU）发生在检查之后、真正执行之前对象或权限发生变化．审批应绑定 $h(a)$、V_o、授权范围和到期时间．执行前再次读取对象版本与当前权限；不一致时重新规划或重新审批．批准“把工单 123 分派给甲组”不能授权后来被模型改成的“关闭工单 123”．

人工协作是一种明确的等待状态，保存展示给人的提案、证据和选择．审批通过后恢复的是同一个动作身份；拒绝应被记录为约束或终止，不应让模型循环换一种措辞再次执行相同被拒动作．对于本来已获授权且参数未变的低风险重复执行，不应无理由重复审批．

16.7 编码 Harness 及个人智能体平台

本节将具体项目作为架构案例．以下产品事实依据 2026 年 9 月 8 日查阅的一手资料；功能会随版本变化，选型时还需锁定所采用的发布版或提交．表 16.3 中的侧重点不构成排他分类，同一平台可以嵌入其他执行器．

16.7.1 Codex 及 DeepSeek Harness

Codex 的 App Server 将智能体执行能力提供给客户端，交互协议包含线程、轮次、进度事件和审批请求．OpenAI 的架构文章描述了基于 JSON-RPC 的集成方式，并说明界面与执行状态分离的意义：客户端断开不应使服务端丢失任务事实．它适合说明“一个执行核心、多个交互入口”的设计，而不能据此推断所有部署都具备完全相同的工具和权限．[98]

DeepSeek Harness 的官方说明以 Cordis 插件系统为核心，模型、工具、技能、会话、沙箱、存储、循环、调度与界面均可通过插件组合；会话记录采用追加式日志．资

表 16.3: 四种具体实现所展示的架构关注点；表格不表示性能排名或生产验收结果.

项目	可观察的组织方式	应审查的工程问题
Codex	App Server 将交互客户端接入统一的智能体执行能力	线程与轮次身份、事件流、审批与断线恢复
DeepSeek Harness	以 Cordis 插件组织模型、工具、存储与运行循环	插件依赖、替换边界、版本兼容与日志语义
OpenClaw	常驻 Gateway 连接消息渠道、控制客户端与设备节点	消息路由、设备身份、连接恢复和授权主体
Hermes Agent	跨会话记忆、技能维护、消息网关与调度	经验写入质量、技能版本、长期任务及用户隔离

料查阅时它仍处于开发者预览阶段. [99] 这说明可替换性可以深入到运行循环，但可替换不等于语义兼容. 例如，替换存储插件时必须确认事务与恢复语义；替换工具插件时必须保持动作身份及结果契约. 预览项目可以作为架构研究对象，不能仅凭公开源码便认定其接口稳定或已满足生产要求.

16.7.2　OpenClaw 及 Hermes Agent

本书所称 Claw 案例具体指 **OpenClaw**，不把所有带有 Claw 名称的项目视为同一实现. 其官方架构以常驻 Gateway 统一连接消息渠道，控制客户端和设备节点通过 WebSocket 接入；设备能力与身份是显式协议字段. 文档还指出事件出现缺口时客户端需要刷新状态. [100] 因此网关既是渠道适配点，也是会话路由与设备信任的汇合点. 收到一条消息只证明传输完成，不能自动证明发信人有权使用所连接设备的全部能力.

Hermes Agent 是 Nous Research 的智能体项目，区别于同名模型系列. 其官方仓库描述跨会话记忆、会话检索、可维护技能、消息网关和调度能力. [101] 这里的“从经验学习”首先应理解为外部记忆与技能内容的更新，不能直接解释为底层模型权重在线训练. 若一次任务成功后保存了技能，后续仍需检查该经验的前提、来源、版本与失败记录；环境变化后应允许撤回，而不能将“过去成功”提升为永久规则.

这类个人智能体还面临渠道身份与执行身份不一致的问题. 例如，群聊发言者、消息渠道账号、操作系统账号和业务系统账号是不同主体. 可靠实现应明确从哪一个主体派生权限，在哪一层检查对象范围，以及一次会话能否访问另一会话的记忆. 多渠道接入和长期在线均不能替代这些约束.

16.7.3　选型及组合

比较产品时，应先给出任务闭环，再逐项映射组件. 代码修改需要工作区隔离、差异审查与测试证据；持续个人助理需要身份路由、可取消调度和记忆治理. 若个人

助理委派编码 Harness，应记录父子任务身份、预算、工作区、允许工具和最终产物，避免把下游自然语言“完成”当作验收. 集成应通过明确的工具或客户端协议进行，不能仅靠拼接两个聊天记录推断执行顺序.

16.8 Computer Use 的观察及行动闭环

计算机操作（Computer Use）指智能体通过屏幕、界面结构及输入动作操作软件的能力. 它可覆盖浏览器与原生应用，其机制是观察界面、提出动作、由执行器实施，再读取新状态. 模型输出一次点击并不会自行驱动操作系统；执行环境必须实现相应动作并回传观测. OpenAI 的计算机操作接口文档也采用这种模型与执行环境分工. [102]

16.8.1 观测表示及动作落点

记界面观测为 $o_t = (I_t, D_t, A_t, v_t)$：$I_t$ 是截图，D_t 是可获得的文档对象模型，A_t 是可获得的无障碍树，v_t 是页面或窗口状态标识. 并非每个环境都提供全部分量. 截图适用于视觉布局与画布，但坐标依赖窗口、缩放和滚动；文档结构适合语义定位，但不覆盖所有像素内容；无障碍树暴露角色、名称和状态，其完整性取决于应用实现.

动作定位（Grounding）把“打开保存菜单”这样的意图映射到当前界面的具体目标. 动作应绑定窗口或标签页、目标描述及观测版本；只保存一个坐标容易在页面变化后点击错误对象. 若在原始截图 (W_s, H_s) 上定位到 (x_s, y_s)，截图对应视口区域的原点为 (x_0, y_0)、大小为 (W_v, H_v)，且只有轴对齐缩放，则

$$x_v = x_0 + x_s \frac{W_v}{W_s}, \qquad y_v = y_0 + y_s \frac{H_v}{H_s}. \tag{16.3}$$

各量必须使用约定的像素或逻辑坐标单位. 只有知道截图尺寸与视口映射后才能换算；设备像素比不能在缺乏坐标约定时重复乘入. 裁剪、浏览器工具栏偏移、iframe 坐标及页面滚动会改变映射，应重新获取对应几何关系.

16.8.2 动作返回及任务完成

执行链必须区分三件事：输入动作已派发、界面出现预期变化、业务后置条件成立. 点击提交后按钮变灰只支持第一或第二项；确认对象已经创建，通常需要成功回执、对象标识或结果查询. 对于回执未知的提交，应先查询结果，再决定是否重试. 没有幂等语义的界面不应盲目重放整段动作.

设任务有 n 步，第 i 步在此前步骤均正确的条件下成功的概率为 p_i，则由条件概率链式法则

$$P(\text{全程成功}) = \prod_{i=1}^{n} p_i. \tag{16.4}$$

此式不要求步骤独立，但 p_i 必须是上述条件概率；不能把不同测试的边际点击准确率直接相乘．长流程会累积失误，因此应在有业务意义的阶段读取后置条件，及时停止错误传播．设置检查点不能保证恢复成功，尤其不能撤销已经发生的外部副作用．

16.8.3　界面内容及权限边界

页面中的文字是待处理数据，不能因为出现在屏幕上就成为上级指令．截图、DOM 和无障碍树都可能承载提示注入；结构化读取只能改善定位，并不能自动消除恶意内容．执行器应绑定用户已授权的目标、对象和动作范围，隔离会话数据，并在界面变化时重新验证目标．截图与轨迹也可能包含隐私数据，需要控制保存范围、访问权限和保留时间．

16.9　浏览器控制协议及适配

Chrome 开发者工具协议（Chrome DevTools Protocol，CDP）用于检测、调试和控制 Chromium 系浏览器．它按 DOM、Network、Runtime、Page 等域组织命令与事件；协议消息使用结构化 JSON．[103]CDP 是底层浏览器接口，Computer Use 是任务能力，二者不在同一抽象层．智能体可以使用 CDP 取得界面信息，也可以通过截图和输入驱动浏览器．

16.9.1　目标、会话及异步事件

浏览器包含多个 target，例如页面与工作线程；控制器需要先选定目标，再将命令与相应会话关联．请求编号用于匹配响应，事件用于接收异步变化．一次导航可能使先前保存的节点句柄或执行上下文失效，弹窗可能创建新的目标，因此不能把“当前活动页”当作稳定的业务对象身份．控制器应保存目标身份并在导航、断线或状态缺口后重新读取．

在工具层可以定义“定位当前页面中名称为保存的按钮并确认可操作”，由适配器负责浏览器细节．Playwright 提供这类更高层的浏览器自动化接口，也支持通过 CDP 连接 Chromium；其文档明确提示该连接方式相较原生 Playwright 协议连接具有较低保真度．[104] 因此使用同一库名不代表连接路径与功能范围相同．**WebDriver 双向协议**（WebDriver BiDi）是另一条标准化浏览器自动化路径，提供双向通信和事件机制；具体浏览器的实现覆盖仍需核对．[105]

表 16.4: 操作接口的选择应依据对象语义、可验证性和权限，而非仅依据是否能够点击成功.

接入方式	适合表达的对象	主要限制
业务 API	业务对象、事务、明确回执	需要存在可用且获授权的接口
浏览器自动化库	定位器、页面、等待条件	依赖应用结构和驱动兼容性
CDP	浏览器目标、文档、网络与运行时	主要面向 Chromium，需处理版本和会话
无障碍接口	控件角色、名称与状态	应用暴露的语义可能不完整
截图与输入	可见像素与坐标动作	定位误差、几何变化与回执歧义

16.9.2 兼容性及执行边界

CDP 的 tip-of-tree 定义会变化，官方不承诺其向后兼容；应锁定浏览器与协议适配版本，检查所需命令是否受支持. [2]远程调试连接具有广泛的会话访问能力，应限制其可达范围，并与用户日常浏览配置隔离. 使用浏览器已有登录态必须明确授权范围，不能把“能够连接”当作“允许访问所有标签页”.

MCP 等工具接入协议可以把浏览器能力暴露给模型，实际适配器再调用 CDP 或其他驱动. 协议叠加不会自动产生端到端幂等性、认证或业务成功判定；这些仍由本章的运行系统负责. 测试除正常操作外还应覆盖导航中断、目标关闭、登录过期、遮挡、重复按钮、回执丢失和恶意页面指令，并分别记录定位、执行与业务验证失败.

16.10 知识助手及工单处理的链路

16.10.1 业务回执生成

考虑用户要求“根据设备 A 的告警及维护规定建立待处理工单”. 执行链路如下.

1. 接口建立任务，固定租户、主体、设备标识、期限和应用制品版本；以请求幂等键处理用户重复提交.

2. 检索器按主体权限取得维护文档版本与位置；设备查询工具返回状态及对象版本 17. 二者均保存来源与时间，不能把文档说明当作设备当前事实.

3. 模型产生工单草案，包含设备、告警摘要、支持证据和建议优先级. 解析器检查结构，业务规则检查必填信息与取值范围；缺失信息转入等待用户补充.

[2]CDP 文档中的 stable 1.3 是历史冻结子集，不意味着它涵盖现代浏览器的所有能力，也不意味着 tip-of-tree 是稳定发布接口.

4. 若操作在授权范围内，执行器绑定参数摘要与设备版本 17；若需审批，持久化该提案并等待．期间对象变化则重新评估．

5. 在事务中保存动作意图、预算预留及发件箱．工具网关再次授权，以稳定动作键调用工单系统．

6. 工单系统返回工单标识与版本．运行系统记录回执后才生成“已创建”的答复；若超时，则按键查询或进入待核对，答复必须说明状态尚未确认．

7. 后续更新工单采用其版本条件．任务结束释放未用预留，保留可追溯事件与按保留策略管理的证据引用．

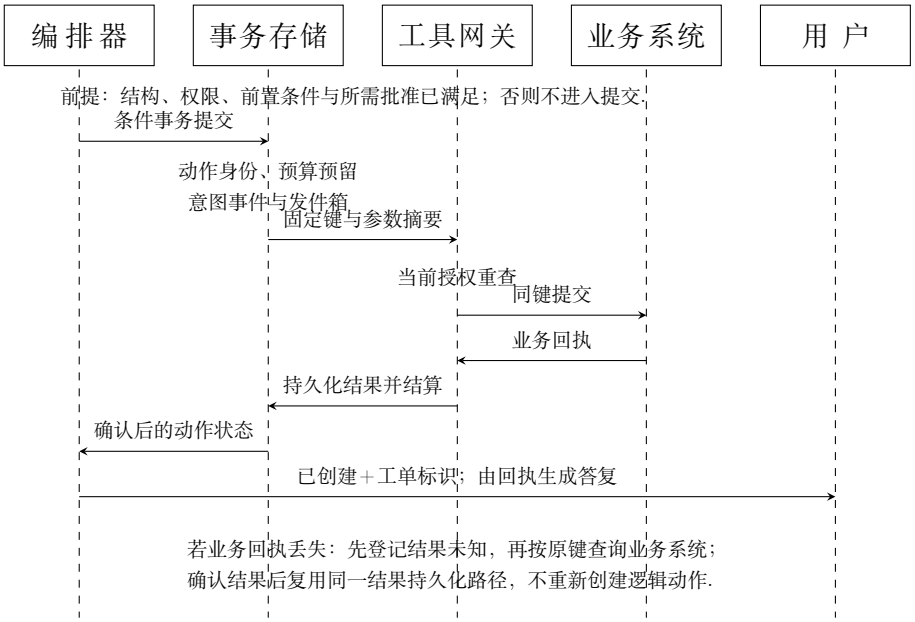


图 16.3: 工单创建的持久执行时序．当前授权重查失败时终止发送并记录拒绝；没有业务回执时不能走“已创建”答复路径．

图 16.3 区分本地事务提交、远端业务提交和最终回答三个时刻．发件箱只保证发送意图不会被本地故障静默丢弃；远端幂等、结果查询与重复回执处理共同约束恢复行为．

16.10.2　故障恢复路径

算法 16.1　计算过程

受控动作提交与恢复．

1. 从任务版本读取允许工具、剩余预算、期限和终止状态，解析提案并生成规范化参数摘要．

2. 检查当前权限、业务前置条件与审批绑定；不满足则记录明确状态，不调用

表 16.5: 不同故障位置的可确认事实与恢复动作.

故障位置	可确认事实	恢复动作
意图事务提交前	无持久动作承诺	从最近事件恢复，不假定已经发送.
提交后、发送前	发件箱存在	分发者继续投递.
远端完成、响应丢失	本地结果未知	以相同键查询或重试，禁止新建逻辑动作.
结果记录后、答复前	业务回执已持久化	由回执重建答复，不再次执行.
审批后对象变更	原审批基于旧版本	再检查、重规划，必要时重新审批.
取消后收到回执	取消与外部完成均发生	记录真实效果，按授权决定是否补偿.

外部写入.

3. 以任务版本条件，在同一事务中预留预算、建立动作身份、追加意图事件和发件箱记录.
4. 分发者使用固定幂等键执行；把明确失败、成功回执和结果未知分别记录. 并发记录通过动作版本与唯一约束去重.
5. 对未知结果使用受信查询核对；只有满足幂等契约才自动重试写入. 所有尝试共享原动作身份.
6. 结算实际成本并追加结果事件，驱动状态转换；恢复时读取既有结果，不重复已完成副作用.

16.10.3 部署拓扑及故障隔离

任务接口可以无状态复制，工作进程按模型活动与工具活动分池，以避免慢工具占满模型执行槽. 各副本通过事务存储竞争任务版本，不能依赖进程内存保存唯一任务状态. 知识索引按权限域访问，业务系统仅对工具网关开放凭证范围，模型服务没有直连业务写入的权限.

图 16.4 中的边界由网络与凭证共同落实，而不是仅靠服务命名隔离. 事务存储需要满足明确的恢复点目标与恢复时间目标；备份恢复还须与外部已完成动作核对. 若数据库恢复到了动作执行前的旧时间点，简单重放发件箱可能再次发送已执行动作，因此幂等记录的保留窗口应覆盖灾难恢复范围. 跨区恢复不能只验证服务启动，还要验证任务身份、动作键和业务回执是否继续一致.

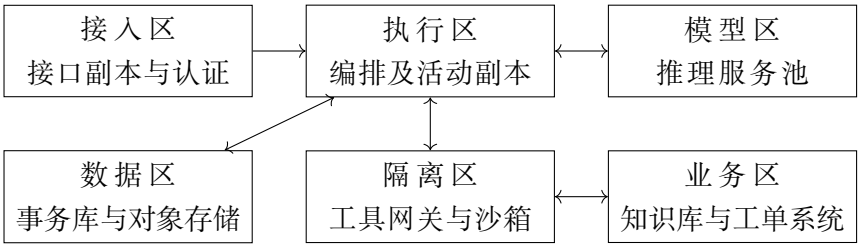


图 16.4: 以网络与凭证范围隔离模型推理和业务写入．部署区表示职责边界，不规定必须使用六套集群.

16.11　测试、观测及发布

16.11.1　执行轨迹的判定边界

轨迹回放（Trajectory Replay）使用已记录模型结果与工具结果验证控制器状态转换；它可以发现分支、预算和恢复错误，但不能证明真实接口当前可用．真实集成验证需要在受控业务环境中核对接口语义和最终状态．故障注入应覆盖发送前后崩溃、重复回调、过期租约、参数冲突、审批撤销、取消竞争与预算并发.

评估应分别测量参数有效、授权判定、工具实际完成和最终任务完成．对串行必要环节 $E_1, \ldots, E_m$，准确关系是

$$\Pr\left(\bigcap_{j=1}^{m} E_j\right) = \Pr(E_1) \prod_{j=2}^{m} \Pr(E_j \mid E_1, \ldots, E_{j-1}). \tag{16.5}$$

无须独立即可成立，但把各环节边际成功率直接相乘需要独立假设．模型误解目标往往同时导致检索与工具错误，故用独立乘积估计应用可靠性可能失真.

日志应记录任务、动作、尝试、模型修订、工具版本、事件序号和终止原因，敏感负载采用受控引用．指标维度使用有限集合，任务标识放入追踪或日志；观测“执行成功”必须对应业务回执，不能只统计 HTTP 成功．公开决策摘要用于解释选择，不要求保存或暴露模型内部思维过程.

发布时固定整个应用制品组合，在相同任务集上比较业务结果、权限违规、人工干预、成本、恢复行为与时延．新模型的语言表现更好，不代表工具参数与控制器兼容．旧长任务应继续使用兼容版本，或经过显式状态迁移；简单切换流量只影响新任务，不能自动回滚已提交的业务效果.

第三部分

模型训练方法

第十七章　优化及泛化

反向传播回答了“当前参数怎样影响损失”，优化算法则必须进一步决定沿什么方向、以多大幅度改变参数. 即使梯度完全正确，训练仍可能因学习率不合适而振荡，因初始化尺度不当而失去有效信号，或在训练集上持续改善的同时损害对新样本的预测. 优化及泛化因此是两个相互联系而不能相互替代的问题：前者研究如何降低既定目标，后者研究从有限数据获得的预测规律能否适用于未参与拟合的数据.

本章先把训练目标与统计目标分开，在可精确求解的二次函数上推导梯度下降的稳定条件，再将结论扩展到随机梯度和小批量计算. 随后从信号传播解释初始化，从目标函数解释正则化，最后讨论偏差、方差、数据划分与分布漂移. 数值例题均给定明确的目标、参数与抽样假设，以便逐项验证结论.

17.1　训练目标及统计目标

17.1.1　总体风险、经验风险及正则化目标

设观测 $z=(x,y)$ 来自联合分布 P，训练集为 $S=\{(x_i,y_i)\}_{i=1}^{n}$. 参数 $\theta\in\mathbb{R}^p$ 确定预测函数 f_θ，损失 $\ell(f_\theta(x),y)$ 衡量预测与目标之间的差异. 定义

$$R_P(\theta)=\mathbb{E}_{(x,y)\sim P}\big[\ell(f_\theta(x),y)\big], \tag{17.1}$$

$$\widehat{R}_S(\theta)=\frac{1}{n}\sum_{i=1}^{n}\ell(f_\theta(x_i),y_i), \tag{17.2}$$

$$F_S(\theta)=\widehat{R}_S(\theta)+\lambda\Omega(\theta),\qquad \lambda\geqslant 0. \tag{17.3}$$

总体风险 R_P 是希望降低的统计量，经验风险 $\widehat{R}_S$ 是有限样本上的可计算估计，正则化目标 F_S 是实际交给优化算法的函数. 三者可能具有不同的最小点. 正则化项 Ω 并非测试误差的一部分；它通过改变训练时偏好的解，间接影响总体风险.

若训练样本独立同分布，而且参数 θ 在抽取 S 之前已经固定，则 $\mathbb{E}_S\widehat{R}_S(\theta)=R_P(\theta)$. 但是训练得到的 $\widehat{\theta}(S)$ 依赖同一批样本，不能把这个固定参数下的无偏性直接代入数据选择后的参数. 一个足以记忆标签的模型可以使训练误差接近零，同时在新

样本上没有预测能力．这一差别是必须保留独立评估数据的根本原因．[1]

符号	含义与范围
n, B, p	训练样本数、批量大小、参数维数，均为正整数.
θ_k, g_k, η_k	第 k 步参数、梯度估计和学习率；前两者属于 $\mathbb{R}^p$，$\eta_k > 0$.
H, L, μ	对称 Hessian 矩阵，以及最大、最小曲率；正定情形满足 $0 < \mu \leqslant L$.
Σ, λ, q	单样本梯度的协方差矩阵、正则强度、Dropout 保留概率；$0 < q \leqslant 1$.
$n_{\text{in}}, n_{\text{out}}$	线性层的输入、输出宽度.

这里的 L 表示梯度变化的尺度或最大曲率，不表示序列长度．后文按局部问题说明符号，避免把参数维数、样本数与批量大小混为一谈.

17.1.2　优化误差及统计误差

在本小节暂取 $\lambda = 0$．设 $\widehat{\theta}$ 是经验风险的一个全局最小点，θ^* 是同一参数集合内总体风险的最小点，θ_K 是执行 K 次更新后实际得到的参数．插入并减去经验风险可得

$$\begin{aligned} R_P(\theta_K) - R_P(\theta^*) = {} & [R_P(\theta_K) - \widehat{R}_S(\theta_K)] \\ & + [\widehat{R}_S(\theta_K) - \widehat{R}_S(\widehat{\theta})] \\ & + [\widehat{R}_S(\widehat{\theta}) - \widehat{R}_S(\theta^*)] \\ & + [\widehat{R}_S(\theta^*) - R_P(\theta^*)]. \end{aligned} \tag{17.4}$$

第二项是非负的优化误差；第三项因经验最优性而非正．若能够在整个参数集合上控制 $\Delta_S = \sup_{\theta} |R_P(\theta) - \widehat{R}_S(\theta)|$，则

$$R_P(\theta_K) - R_P(\theta^*) \leqslant 2\Delta_S + \widehat{R}_S(\theta_K) - \widehat{R}_S(\widehat{\theta}). \tag{17.5}$$

这个上界不是宣称大网络的 Δ_S 容易估计，而是说明减小优化误差与减小统计估计误差承担不同职责．延长训练不能修复数据泄漏，增加样本也不能自动修复错误梯度．若参数集合本身不能表达最优预测规律，还要在总体最优值之外考虑模型的近似误差.

17.2　梯度下降及曲率

17.2.1　负梯度的下降性质

对可微目标 F，在当前位置 θ 附近有

$$F(\theta + \delta) = F(\theta) + \nabla F(\theta)^{\mathsf{T}} \delta + o(\|\delta\|_2). \tag{17.6}$$

[1] “独立”指评估信息没有通过参数拟合、特征选择、超参数搜索或人工筛选等路径进入模型选择，并不只是文件存放在另一个目录.

在固定步长 $\|\delta\|_2 = r$ 下，柯西–施瓦茨不等式给出 $\nabla F(\theta)^{\mathsf{T}}\delta \geqslant -r\|\nabla F(\theta)\|_2$，等号在 δ 沿负梯度方向时成立. 因此欧氏距离下最陡的一阶下降方向是 $-\nabla F(\theta)$. 这并未给出可以任意增大的步长，因为较大的 δ 会使曲率项无法忽略.

若梯度满足 L-Lipschitz 条件，即 $\|\nabla F(u) - \nabla F(v)\|_2 \leqslant L\|u - v\|_2$，沿线段积分可得

$$\begin{aligned} F(\theta + \delta) - F(\theta) &= \int_0^1 \nabla F(\theta + t\delta)^{\mathsf{T}}\delta \, \mathrm{d}t \\ &\leqslant \nabla F(\theta)^{\mathsf{T}}\delta + \frac{L}{2}\|\delta\|_2^2. \end{aligned} \tag{17.7}$$

代入 $\delta = -\eta\nabla F(\theta)$ 后，

$$F(\theta - \eta\nabla F(\theta)) \leqslant F(\theta) - \eta\left(1 - \frac{L\eta}{2}\right)\|\nabla F(\theta)\|_2^2. \tag{17.8}$$

因此 $0 < \eta < \dfrac{2}{L}$ 能保证非零梯度处的下降. 对非凸目标，这一结论只控制函数值，不保证到达全局最小点；驻点可能是局部最小点、鞍点或退化的平坦区域. 深度网络中的 ReLU 还含不可微位置，不能不加说明地把处处光滑定理视为整个训练过程的严格证明.

17.2.2　二次目标稳定域

考虑正定二次目标

$$F(\theta) = \frac{1}{2}(\theta - \theta^*)^{\mathsf{T}}H(\theta - \theta^*), \quad H = H^{\mathsf{T}} \succ 0. \tag{17.9}$$

令误差 $e_k = \theta_k - \theta^*$，梯度为 He_k. 固定学习率的更新满足

$$e_{k+1} = (I - \eta H)e_k. \tag{17.10}$$

对称矩阵可作正交特征分解 $H = Q\Lambda Q^{\mathsf{T}}$，其中 $Q^{\mathsf{T}}Q = I$、$\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_p)$. 转到特征坐标 $a_k = Q^{\mathsf{T}}e_k$ 后，各方向独立递推：

$$a_{k+1,j} = (1 - \eta\lambda_j)a_{k,j}, \qquad a_{k,j} = (1 - \eta\lambda_j)^k a_{0,j}. \tag{17.11}$$

要使所有初值在所有方向收敛，必须且只需每个放大因子的绝对值小于一. 逐步解不等式可得

$$|1 - \eta\lambda_j| < 1 \iff -1 < 1 - \eta\lambda_j < 1 \iff 0 < \eta\lambda_j < 2. \tag{17.12}$$

令 $L = \max\limits_j \lambda_j$，所有方向的交集即

$$\boxed{0 < \eta < \frac{2}{L}.} \tag{17.13}$$

若 $0<\eta\lambda_j<1$，误差不变号地衰减；若 $1<\eta\lambda_j<2$，误差交替变号而幅值衰减；在 $\eta\lambda_j=2$ 处，非零初始误差保持等幅振荡；超过二则沿该方向发散．$\eta\lambda_j=1$ 则在一步内消去该方向的误差．

如果 H 仅半正定，零特征值方向不改变，算法只能收敛到由初值决定的最小点；若出现负特征值，对应因子 $1-\eta\lambda_j>1$，负曲率方向会放大扰动．这些情况不能沿用正定情形“收敛到唯一最优参数”的表述．

17.2.3　条件数及特征尺度

令 $\mu=\min_j \lambda_j>0$．最坏方向的收缩因子为 $\rho(\eta)=\max_j |1-\eta\lambda_j|$．在 $[\mu,L]$ 上平衡两端幅值，即令 $1-\eta\mu=-(1-\eta L)$，得到

$$\eta_{\mathrm{bal}}=\frac{2}{L+\mu},\qquad \rho(\eta_{\mathrm{bal}})=\frac{L-\mu}{L+\mu}=\frac{\kappa-1}{\kappa+1},\qquad \kappa=\frac{L}{\mu}. \tag{17.14}$$

κ 称为条件数．它很大时，陡峭方向限制学习率，而平缓方向前进缓慢．标准化输入可改善某些由单位差异引起的条件问题，却不保证消除特征相关性．对最小二乘 $F=\dfrac{\|X\theta-y\|_2^2}{2n}$，Hessian 为 $\dfrac{X^{\mathsf T}X}{n}$；标准化主要调整对角尺度，高度相关的列仍可能使最小特征值接近零．

17.2.4　下降、振荡及发散

取 $F(u,v)=\dfrac{u^2+9v^2}{2}$，初值 $(u_0,v_0)=(1,1)$．此时 $L=9$、$\mu=1$，稳定区间为 $0<\eta<\dfrac{2}{9}$．由式 (17.11) 直接计算得到下表．

η	u_1	v_1	$F(u_1,v_1)$	$F(u_2,v_2)$
0.10	0.9	0.1	0.45	0.3285
0.20	0.8	-0.8	3.2	2.048
0.25	0.75	-1.25	7.3125	11.14453125

初始目标值为 5．第二行虽然发生变号，仍稳定下降；第三行在陡峭方向的误差放大因子为 -1.25，因而发散．不能仅凭参数或梯度的正负交替判断失败，也不能把较大的学习率简单理解为较快的有效学习．

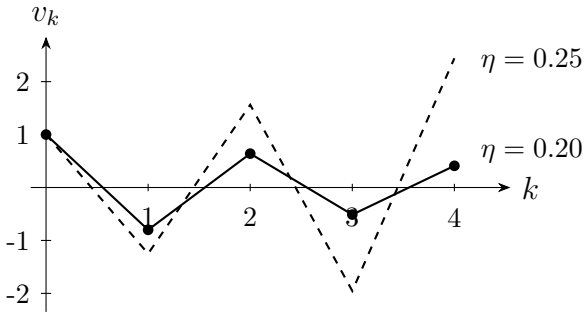


图 17.1: 二次例题的陡峭方向：实线振荡收敛，虚线振荡发散．点由 $v_k = (1-9\eta)^k$ 精确计算，连线仅用于识别离散更新顺序．

17.3　随机梯度及批量大小

17.3.1　无偏抽样条件

记 $h_i(\theta) = \nabla_\theta \ell(f_\theta(x_i), y_i)$．固定数据集和当前参数，从 $\{1, \dots, n\}$ 独立均匀有放回抽取 B 个索引 $I_1, \dots, I_B$，构造

$$g(\theta) = \frac{1}{B}\sum_{b=1}^{B} h_{I_b}(\theta), \qquad \mathbb{E}[g(\theta) \mid S, \theta] = \frac{1}{n}\sum_{i} h_i(\theta) = \nabla \widehat{R}_S(\theta). \tag{17.15}$$

若每步使用的新随机索引独立于给定训练历史后的参数状态，则这个等式也是条件于训练历史的无偏性．它估计的是当前训练集的经验梯度，并不自动等于总体梯度．在总体抽样模型下，要写成 $\nabla R_P = \mathbb{E}\nabla \ell$，还需要足够的可微性、可积性等条件，允许交换求导与期望．

非均匀采样时，若抽样概率为 $\pi_i > 0$，直接平均 h_{I_b} 的期望变为 $\sum_i \pi_i h_i$，对应另一个加权目标．要保留原等权目标，可以使用

$$g(\theta) = \frac{1}{B}\sum_{b} \frac{h_{I_b}(\theta)}{n \pi_{I_b}}. \tag{17.16}$$

其无偏性由 $\sum_i \dfrac{\pi_i h_i}{n\pi_i} = n^{-1}\sum_i h_i$ 得到，但极小的 π_i 可能造成很大的方差．因此重采样、类别权重和重要性校正既影响数据分布，也影响更新稳定性．

17.3.2　梯度方差及有限总体修正

设 $\bar{h} = n^{-1}\sum_i h_i$，定义有限训练集上的协方差

$$\Sigma = \frac{1}{n}\sum_{i} (h_i - \bar{h})(h_i - \bar{h})^{\mathsf{T}}. \tag{17.17}$$

有放回独立抽样使交叉协方差为零，所以

$$\operatorname{Cov}(g \mid S, \theta) = \frac{\Sigma}{B}, \qquad \mathbb{E}\|g - \bar{h}\|_2^2 = \frac{\operatorname{tr}\Sigma}{B}. \tag{17.18}$$

当 $n > 1$ 时，对均匀无放回批量，两个不同抽样位置的协方差为 $-\dfrac{\Sigma}{n-1}$．把 B 个方差项和 $B(B-1)$ 个交叉项相加，得到

$$\operatorname{Cov}(g) = \frac{1}{B^2}\left(B\Sigma - \frac{B(B-1)}{n-1}\Sigma \right) = \frac{n-B}{B(n-1)}\Sigma. \tag{17.19}$$

当 $B=n$ 时协方差为零，当 $B \ll n$ 时接近 $\dfrac{\Sigma}{B}$. 这里比较的是同一个固定参数处随机选取一个批量的分布. 逐轮随机打乱后依次消费批量时，参数已经依赖之前消费的样本，剩余样本上的梯度一般不再条件无偏于全训练集梯度. 不能将单次无放回抽样公式无条件推广为整条训练轨迹上的独立噪声模型.

若样本存在相关性，更一般的表达是

$$\operatorname{Cov}(g)=\frac{1}{B^2}\sum_{b,c}\operatorname{Cov}(h_{I_b},h_{I_c}). \tag{17.20}$$

同一文档的相邻片段、同一用户的重复记录可能具有正相关性，增加名义批量并不产生同等数量的独立信息. 语言模型中还必须说明批量按序列、有效词元还是文档计算.

17.3.3　随机梯度下降条件

设条件期望 $\mathbb{E}[g_k]=\nabla F(\theta_k)$，条件方差满足 $\mathbb{E}\|g_k-\nabla F(\theta_k)\|_2^2 \leqslant \dfrac{\sigma^2}{B}$. 将随机更新代入光滑上界，并使用 $\mathbb{E}\|g_k\|_2^2=\|\nabla F(\theta_k)\|_2^2+\mathbb{E}\|g_k-\nabla F(\theta_k)\|_2^2$，可得

$$\mathbb{E}[F(\theta_{k+1})\mid\theta_k]\leqslant F(\theta_k)-\eta\left(1-\frac{L\eta}{2}\right)\|\nabla F(\theta_k)\|_2^2+\frac{L\eta^2\sigma^2}{2B}. \tag{17.21}$$

最后一项说明单步可能上升. 靠近驻点时梯度信号减弱，固定学习率下的噪声仍持续作用；降低学习率或增加批量能减小该项，但也改变了计算开销与单位预算下的更新次数.

在一维二次函数 $F(e)=\dfrac{he^2}{2}$ 上，若 $g_k=he_k+\xi_k$，ξ_k 独立、零均值、方差为 $\dfrac{s^2}{B}$，则

$$\mathbb{E}[e_{k+1}^2]=(1-\eta h)^2\mathbb{E}[e_k^2]+\frac{\eta^2 s^2}{B}. \tag{17.22}$$

当 $0<\eta h<2$，长期二阶矩收敛到

$$\lim_{k\to\infty}\mathbb{E}[e_k^2]=\frac{\frac{\eta^2 s^2}{B}}{1-(1-\eta h)^2}=\frac{\eta s^2}{Bh(2-\eta h)}. \tag{17.23}$$

这是一组明确假设下的噪声稳态，不是所有神经网络的通用最终误差公式. 它同时揭示：较大的学习率使快速移动与较大的稳态波动相伴；当学习率接近稳定边界时，即使噪声很小，波动也可能很大.

17.3.4　批量、步数及有效监督量

某固定参数处四个标量样本梯度为 $1,3,5,7$，均值为 4、总体方差为 5. 有放回抽取两个样本，均值梯度方差为 $\dfrac{5}{2}$；无放回抽取两个样本，则为 $\dfrac{5(4-2)}{2\cdot 3}=\dfrac{5}{3}$. 两种批量的均值相同，随机性不同.

再设训练集含 1024 个样本，完整遍历一轮且不丢尾批．批量 32 对应 32 次参数更新，批量 128 仅对应 8 次．若固定更新次数，则后者消费更多样本；若固定 epoch 数，则后者更新更少；若固定墙钟时间，还会受到设备利用率与通信影响．三种比较回答不同问题，不能互相替代．

梯度累积只有在各微批梯度于同一参数处计算、按真实样本数或有效词元数加权、期间不更新参数，并保持等价的计算状态时，才可与相应大批量梯度比较．含批量统计的层、不同随机掩码及不均匀有效长度会使简单的“微批均值再平均”偏离目标．具体训练状态与分布式归约将在训练章节展开．

动量通过累计过去梯度减弱方向的短时变化，自适应优化器进一步按坐标调节更新尺度．它们改变上述标量学习率的动力学，不能直接套用 $\frac{2}{L}$ 作为完整稳定界．本章建立的目标、条件期望与曲率分析仍是解释其行为的基础；Adam、AdamW 的矩估计、偏差修正和恢复状态在训练过程一章推导．

算法 17.1　带验证集早停的小批量梯度下降

输入：相互隔离的训练集和验证集、可微损失、学习率序列、批量大小、最大步数 K、验证间隔 r、耐心窗口 M、正则强度 $\lambda \geqslant 0$ 和最小改善量 $\delta_{\min} \geqslant 0$；验证损失越小越好．**输出：**验证指标最优的已保存参数．

1. 初始化参数 θ_0；在固定评估模式下计算初始验证损失，保存参数和当前最小验证损失，并将耐心比较基准设为此损失，连续未改善次数置零．

2. 对 $k = 0, \ldots, K-1$，从训练集按明确规则抽取批量，在 θ_k 处求其平均数据梯度 g_k；二次正则项在下一步统一加入．

3. 完成全部梯度计算后更新 $\theta_{k+1} = \theta_k - \eta_k(g_k + \lambda\theta_k)$．这里 g_k 仅指数据梯度，正则项只加入一次．

4. 每隔 r 步按同一口径计算验证数据损失；若损失小于已保存的最小值，保存当前参数并更新最小值；另将损失与耐心比较基准比较，下降量严格大于 $\delta_{\min}$ 时更新该基准并清零未改善次数，否则加一．连续 M 次不改善或达到 K 步时停止．

5. 返回保存的最优参数；配置冻结后再使用封存测试集．验证过程不反向传播，也不更新训练预处理状态．

批量损失的求导遵循第三章《神经网络基础》中的反向传播规则．恢复后继续训练还必须保存随机状态、数据位置和优化器状态，不能只保存用于预测的参数．

17.4 初始化及信号传播

17.4.1 打破对称性及控制尺度

若同一隐藏层的神经元具有相同输入权重、偏置及后续连接，并经过相同运算，则它们产生相同激活和梯度，更新后仍保持对称. 这使多个神经元难以学习不同特征. 随机初始化的首要作用是打破这种对称性，而不是仅为实验提供随机种子.

随机并不意味着任意尺度都合适. 考虑线性层 $z_j = \sum_{i=1}^{n_{\text{in}}} W_{ij} a_i$，其中 $W \in \mathbb{R}^{n_{\text{in}} \times n_{\text{out}}}$. 设权重独立、零均值、方差为 s_W^2，与输入独立；输入分量具有相同二阶矩 $q_a = \mathbb{E}[a_i^2]$. 对权重和输入共同取期望，交叉项因独立零均值权重消失：

$$\mathbb{E}[z_j] = 0, \qquad \mathbb{E}[z_j^2] = \sum_i \mathbb{E}[W_{ij}^2]\mathbb{E}[a_i^2] = n_{\text{in}} s_W^2 q_a. \tag{17.24}$$

这里即使输入均值不为零，公式仍使用二阶矩 q_a，不能擅自替换成输入方差.

17.4.2 Xavier 初始化的前后向平衡

在线性或激活近似线性的区域，保持前向二阶矩要求 $n_{\text{in}} s_W^2 \approx 1$. 反向传播给出 $\delta a_i = \sum_{j=1}^{n_{\text{out}}} W_{ij} \delta z_j$；在梯度与权重近似独立、同尺度的分析下，保持反向二阶矩要求 $n_{\text{out}} s_W^2 \approx 1$. 输入输出宽度不相等时，两者不能同时严格满足. Xavier 初始化采用折中尺度 [106]：

$$s_W^2 = \frac{2}{n_{\text{in}} + n_{\text{out}}}. \tag{17.25}$$

若权重服从对称均匀分布 $\mathcal{U}[-r, r]$，其方差为 $\frac{r^2}{3}$，于是

$$W_{ij} \sim \mathcal{U}\left[-\sqrt{\frac{6}{n_{\text{in}} + n_{\text{out}}}}, \sqrt{\frac{6}{n_{\text{in}} + n_{\text{out}}}}\right]. \tag{17.26}$$

该推导关注初始尺度；反向梯度与权重的独立性通常是近似分析，不是训练全过程都成立的事实. 激活进入饱和区域、参数更新导致相关性增强后，初始矩匹配不再足以保证梯度稳定.

17.4.3 He 初始化方差

对关于零对称的预激活 z，令 $a = \max(0, z)$. 由于正负半轴对 z^2 的贡献相等，

$$\mathbb{E}[a^2] = \mathbb{E}[z^2 \mathbf{1}_{z>0}] = \frac{1}{2}\mathbb{E}[z^2]. \tag{17.27}$$

将其代入式 (17.24)，相邻层预激活的二阶矩满足

$$q_z^{(\ell+1)} = \frac{1}{2} n_{\text{in}} s_W^2 q_z^{(\ell)}. \tag{17.28}$$

因此保持这一尺度要求

$$s_W^2 = \frac{2}{n_{\text{in}}}, \qquad W_{ij} \sim \mathcal{N}\left(0, \frac{2}{n_{\text{in}}}\right), \tag{17.29}$$

这就是面向 ReLU 的 He 初始化尺度 [107]. 若改用均匀分布，对应半宽为 $\sqrt{\dfrac{6}{n_{\text{in}}}}$. 对于负半轴斜率为 a 的分段线性激活，同样的对称性给出 $\mathbb{E}[\phi(z)^2] = \dfrac{(1+a^2)\mathbb{E}[z^2]}{2}$，所以尺度改为 $\dfrac{2}{[(1+a^2)n_{\text{in}}]}$.

ReLU 输出的均值通常为正；若 $z \sim \mathcal{N}(0, v)$，则

$$\mathbb{E}[\text{ReLU}(z)] = \sqrt{\frac{v}{2\pi}}, \qquad \text{Var}(\text{ReLU}(z)) = v\left(\frac{1}{2} - \frac{1}{2\pi}\right). \tag{17.30}$$

因此“He 初始化使每层 ReLU 输出方差精确不变”并不是上述推导得到的结论. 推导直接保持的是预激活方差与相应激活二阶矩的层间尺度. [2]

17.4.4　数值尺度及结构边界

设一层输入宽度为 512、输出宽度为 2048. Xavier 正态尺度的标准差为 $\sqrt{\dfrac{2}{2560}} \approx 0.02795$，He 前向尺度的标准差为 $\sqrt{\dfrac{2}{512}} = 0.0625$. 两者差异来自宽度和激活假设，不能依据名称任选其一.

残差分支将多个信号相加，注意力引入输入相关的混合权重，门控前馈层包含乘法，归一化改变激活分布. 这些结构均会改变简单链式网络的矩传播. 初始化应结合完整结构分析，并观察分层激活、梯度及实际更新量；一个全局梯度范数可能掩盖少数层异常. 归一化和残差连接可以改善信号传播条件，但不替代合理初始化或数值稳定的损失计算.

17.5　正则化及其作用机制

17.5.1　二次惩罚及权重衰减

取 $\Omega(\theta) = \dfrac{\|\theta\|_2^2}{2}$，则 $\nabla\Omega(\theta) = \theta$. 普通梯度下降变为

$$\theta_{k+1} = \theta_k - \eta\big(\nabla \widehat{R}_S(\theta_k) + \lambda\theta_k\big) = (1-\eta\lambda)\theta_k - \eta\nabla\widehat{R}_S(\theta_k). \tag{17.31}$$

[2] 文献和实现中常将这类方法统称为方差缩放初始化. 阅读具体公式时仍须区分中心化方差 $\mathbb{E}[(X-\mathbb{E}X)^2]$ 与二阶矩 $\mathbb{E}[X^2]$.

在这个特定更新规则中，目标中的二次惩罚等价于每步乘性权重衰减．若先用一个自适应对角矩阵 D_k 缩放梯度，则惩罚项变为 $\lambda D_k\theta_k$，不再等价于统一的 $\lambda\theta_k$ 衰减．二次正则化与解耦权重衰减的区别由此产生．

在线性回归中，正则目标为

$$F(\theta)=\frac{1}{2n}\|X\theta-y\|_2^2+\frac{\lambda}{2}\|\theta\|_2^2. \tag{17.32}$$

令梯度为零，得到

$$\left(\frac{X^{\mathsf{T}}X}{n}+\lambda I\right)\widehat{\theta}_\lambda=\frac{X^{\mathsf{T}}y}{n}. \tag{17.33}$$

$\lambda>0$ 使原来可能奇异的正规方程变为正定系统．对 $\dfrac{X}{\sqrt{n}}=U\,\mathrm{diag}(s_j)V^{\mathsf{T}}$ 作奇异值分解，参数在第 j 个右奇异方向的系数为

$$(V^{\mathsf{T}}\widehat{\theta}_\lambda)_j=\frac{s_j}{s_j^2+\lambda}\left(U^{\mathsf{T}}\frac{y}{\sqrt{n}}\right)_j. \tag{17.34}$$

相对于无正则解 $\dfrac{1}{s_j}$ 的放大，弱约束方向受到更强抑制，这能降低对噪声的敏感性，同时引入系统性收缩偏差．正则化强度还依赖特征尺度与损失归一化：将平均损失改成总和而保持 λ 不变，会改变两项的相对权重．

17.5.2　Dropout 的期望及方差

随机失活（Dropout）在训练时随机屏蔽部分激活 [108]．设确定激活向量为 a，各分量独立抽取 $m_j\sim\mathrm{Bernoulli}(q)$，$q$ 是保留概率．采用反向缩放形式

$$\widetilde{a}_j=\frac{m_j}{q}a_j, \tag{17.35}$$

可得

$$\mathbb{E}[\widetilde{a}_j\mid a]=a_j, \tag{17.36}$$

$$\mathrm{Var}(\widetilde{a}_j\mid a)=\mathbb{E}[\widetilde{a}_j^2\mid a]-a_j^2=\frac{1-q}{q}a_j^2. \tag{17.37}$$

对于下一层线性输出 $s=w^{\mathsf{T}}\widetilde{a}$，独立掩码使交叉协方差为零，因而

$$\mathbb{E}[s\mid a]=w^{\mathsf{T}}a,\qquad \mathrm{Var}(s\mid a)=\frac{1-q}{q}\sum_j w_j^2a_j^2. \tag{17.38}$$

若目标为标量 y、使用平方损失，则恒等式 $\mathbb{E}[(s-y)^2]=(\mathbb{E}s-y)^2+\mathrm{Var}(s)$ 给出

$$\mathbb{E}_m[(w^{\mathsf{T}}\widetilde{a}-y)^2]=(w^{\mathsf{T}}a-y)^2+\frac{1-q}{q}\sum_j w_j^2a_j^2. \tag{17.39}$$

在这一特定线性读出问题中，随机屏蔽等效于与激活尺度相关的二次惩罚．一般深层非线性网络和交叉熵目标不能直接套用该等式，但仍可看出掩码噪声为何惩罚过度依赖少数激活的拟合方式．

推理时关闭随机屏蔽并使用原激活，保持了该层输入的期望尺度．不过一般有 $\mathbb{E}[\phi(s)] \neq \phi(\mathbb{E}[s])$，所以确定性推理不等于所有随机子网络输出的精确平均．训练模式与推理模式必须显式区分，改变模式也不能代替关闭梯度记录．

例如 $a=(2,-1)$、$w=(1,2)$、$q=\dfrac{1}{2}$，原线性输出为零．四种等概率掩码产生 $s=0,4,-4,0$，均值为零、方差为 8，与公式 $[\dfrac{1-q}{q}](4+4)=8$ 一致．若 $y=0$，原平方误差为零，而期望随机损失为 8，直接展示了随机失活对脆弱抵消关系的惩罚．

17.5.3　早停作为受控的模型选择

早停在一系列参数 $\theta_0,\dots,\theta_K$ 中，依据验证集选择停止位置．它通过限制优化过程影响最终解，不是向训练目标显式加入一个固定惩罚项．在线性最小二乘且从零初始化的情形，令 $H=\dfrac{X^{\mathsf{T}}X}{n}$、$b=\dfrac{X^{\mathsf{T}}y}{n}$，某个正特征值方向满足

$$a_{k+1,j}=(1-\eta\lambda_j)a_{k,j}+\eta b_j. \tag{17.40}$$

把等比数列相加可得

$$a_{K,j}=\frac{1-(1-\eta\lambda_j)^K}{\lambda_j}b_j. \tag{17.41}$$

当 $0<\eta\leqslant\dfrac{1}{L}$ 时，$1-(1-\eta\lambda_j)^K$ 从零逐渐增大，低曲率方向在有限步内尚未完全拟合．这说明早停具有谱方向上的过滤作用．其过滤因子与岭回归的 $\dfrac{\lambda_j}{\lambda_j+\lambda}$ 不同，二者并非一般等价．

实际选择停止时刻需要预先明确验证指标、评估频率、最小改善幅度和耐心窗口，并保存被选中的参数状态．测试集不能承担这一角色．对验证集反复增添候选配置，会逐渐把验证集的信息吸收到选择结果中，因此仍需要封存的最终测试或外层评估．

17.6　偏差、方差及泛化

17.6.1　平方损失下的偏差–方差分解

令真实回归函数为 $f^{*}(x)=\mathbb{E}[Y\mid X=x]$，观测满足 $Y=f^{*}(x)+\varepsilon$，其中 $\mathbb{E}[\varepsilon\mid x]=0$、$\operatorname{Var}(\varepsilon\mid x)=\sigma_{\varepsilon}^{2}(x)$．训练集 S 以及算法随机性共同产生预测 $\widehat{f}_S(x)$．测试噪声与训练过程条件独立，且各变量二阶矩有限．固定 x，定义 $\bar{f}(x)=\mathbb{E}_S\widehat{f}_S(x)$，则

$$Y-\widehat{f}_S(x)=\varepsilon+[f^{*}(x)-\bar{f}(x)]+[\bar{f}(x)-\widehat{f}_S(x)]. \tag{17.42}$$

平方展开后，交叉项因零均值及独立性消失，得到

$$\mathbb{E}_{S,Y|x}[(Y-\widehat{f}_S(x))^2]=\underbrace{[f^*(x)-\bar{f}(x)]^2}_{\text{偏差平方}}+\underbrace{\mathbb{E}_S[(\widehat{f}_S(x)-\bar{f}(x))^2]}_{\text{预测方差}}+\underbrace{\sigma_\varepsilon^2(x)}_{\text{观测噪声}}. \tag{17.43}$$

再对 X 积分即可得到总体平方风险. 偏差衡量平均预测偏离真实条件均值的程度, 方差衡量训练样本与训练随机性改变时预测波动的程度. 这里的预测方差不同于前文单步梯度估计的方差，二者不能直接相加或互相替代.

这一加法分解依赖平方损失及条件均值目标，不能原样用于准确率或交叉熵. 深度网络的测试表现也不必随参数量呈简单的 U 形关系；数据规模、结构、优化过程和正则化共同决定预测函数的分布. 因此“模型更大必然方差更大”不是本分解证明的命题.

17.6.2 收缩估计的取舍

考虑无输入的回归问题 $Y=\mu+\varepsilon$，训练样本独立且噪声方差为 σ^2. 样本均值 $\bar{Y}$ 满足 $\mathbb{E}\bar{Y}=\mu$、$\mathrm{Var}(\bar{Y})=\dfrac{\sigma^2}{n}$. 令预测器为 $\widehat{\mu}_a=a\bar{Y}$，$0\leqslant a\leqslant 1$，则

$$\mathrm{Bias}^2=(1-a)^2\mu^2,\qquad \mathrm{Var}(\widehat{\mu}_a)=a^2\frac{\sigma^2}{n}. \tag{17.44}$$

取 $\mu=1$、$\sigma^2=4$、$n=4$. 不收缩时 $a=1$，偏差平方为零、预测方差为 1，对新观测的均方误差为 5. 取 $a=\dfrac{1}{2}$ 时，偏差平方与预测方差各为 $\dfrac{1}{4}$，总误差为 4.5. 牺牲无偏性可以降低总风险，但这组最合适的收缩程度依赖未知分布，实际仍需由独立验证选择.

17.6.3 独立评估的判定范围

若测试样本彼此独立同分布于总体 P，且测试集独立于已经冻结的预测器，测试平均损失才是该预测器总体风险的无偏估计. 样本有限时，它仍有抽样波动. 对于预先固定的预测器和范围在 $[0,1]$ 的损失，Hoeffding 不等式给出

$$\Pr\Big(|\widehat{R}_{\text{test}}-R_P|\geqslant\epsilon\Big)\leqslant 2\exp(-2n_{\text{test}}\epsilon^2). \tag{17.45}$$

这是有界独立样本条件下的概率控制，不能直接套在无界交叉熵或强相关文档片段上. 若在同一测试集上挑选表现最好的模型，预测器便不再独立于测试数据，固定预测器的保证失去适用前提.

报告均值时应说明划分单位、样本量、指标定义与随机性来源. 多个训练随机种子主要测量优化随机性，多个独立数据划分还反映数据抽样的不确定性；两者不能混为同一种重复试验. 严重类别不平衡时，准确率较高也可能对应少数类识别失败，须结合召回率、精确率及错误代价理解结果. 指标与决策阈值的统计细节将在评估篇展开.

17.7　数据隔离及分布漂移

17.7.1　目标应用划分

训练集用于拟合参数与预处理状态，验证集用于选择配置，测试集用于配置冻结后的估计．随机逐行划分隐含“未来面对的是同一总体中的新独立样本”这一假设．如果目标是对新用户、新设备或未来时间段泛化，划分方式必须反映这个外推单位．

同一患者的多次记录应按患者分组，同一文档的切片应按文档或近重复簇分组，同一会话的多个轮次应按会话分组．时间预测应遵守观测时点，仅使用预测发生前能够得到的信息．分层抽样可保持类别比例，却不能替代实体隔离；先切片后随机划分尤其容易让几乎相同的上下文同时进入训练与测试．

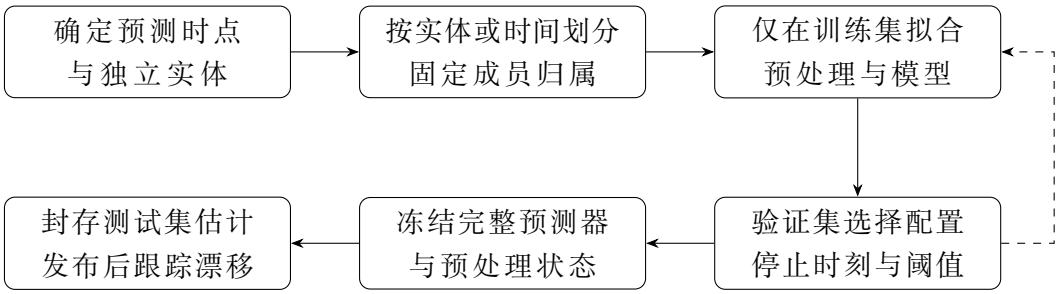


图 17.2: 学习与评估的信息边界．虚线表示允许依据验证结果重新训练；最终测试结果不回流到本轮配置选择．

17.7.2　预处理泄漏及选择泄漏

对第 j 个特征，训练集估计的标准化状态为

$$\widehat{\mu}_j = \frac{1}{n}\sum_{i \in S_{\text{train}}} x_{ij}, \qquad \widehat{s}_j^{\,2} = \frac{1}{n}\sum_{i \in S_{\text{train}}} (x_{ij} - \widehat{\mu}_j)^2. \tag{17.46}$$

验证和测试数据均使用相同变换 $\dfrac{x_j - \widehat{\mu}_j}{\widehat{s}_j}$，零方差特征需制定明确处理规则．若先在全部数据上估计均值、缺失值填充值或特征排序，再划分数据，就让评估样本改变了预测函数的构造．交叉验证时也必须在每个训练折内重新拟合这些状态，而不是复用全数据上的预处理器．

泄漏还可能发生在目标定义和样本选择中．例如使用诊断完成后才形成的字段预测诊断结果，或根据测试错误逐一修改规则．这些行为可能显著改善测试数字，却并未改善模型在真实预测时点的能力．语言模型的评估污染还包含测试题、解析或近重复文本进入预训练和微调语料；文本去重必须考虑这一来源边界，而不只是减少存储量．

一次发布的预测器包含模型权重、预处理参数、类别映射、提示模板及决策阈值．仅冻结权重而继续修改这些组件，也意味着继续进行模型选择．数据隔离与制品重载共同保证统计预测函数的一致性．

17.7.3 训练分布及部署分布不一致

设训练分布为 P、部署分布为 Q. 即使 $R_P(\theta)$ 已得到准确估计, 也未必等于 $R_Q(\theta)$. 常见变化包括: 协变量漂移, 即 $P(X) \neq Q(X)$ 而 $P(Y \mid X) = Q(Y \mid X)$; 标签比例变化, 即类别先验改变而类条件输入分布近似不变; 概念漂移, 即 $P(Y \mid X)$ 本身发生改变. 现实中多种变化可以同时出现, 不能只凭输入均值变化就断定某一种机制.

在协变量漂移假设成立、且 Q_X 相对于 P_X 绝对连续时, 可把部署风险写为

$$\begin{aligned} R_Q(\theta) &= \int \ell(f_\theta(x), y) Q(y \mid x) Q(x) \, \mathrm{d}x \, \mathrm{d}y \\ &= \mathbb{E}_{(x,y)\sim P}\left[\frac{Q(x)}{P(x)} \ell(f_\theta(x), y)\right]. \end{aligned} \tag{17.47}$$

这给出重要性加权的理论依据, 但密度比估计困难, 大权重会放大方差, 训练分布完全未覆盖的区域也无法用有限权重修复. 若条件规律已经变化, 上式的替换前提不成立, 重加权输入分布不足以恢复正确风险.

部署后的观测应同时覆盖输入、预测、延迟获得的真实标签和分组质量. 输入分布改变不必然导致性能下降, 反之总体输入统计看似稳定也可能掩盖少数群体或任务的退化. 是否再训练应由与目标应用一致的评估决定, 并保留数据版本和独立回归集.

17.8 优化失败及过拟合诊断

训练损失高、验证损失也高, 只是一种现象, 可能由模型表达不足、有效信息不足、标注噪声、优化未完成或实现错误引起. 将它直接命名为 “欠拟合” 并增大模型, 会跳过必要诊断. 首先应在固定参数与固定数据上核对损失定义、目标形状及梯度; 随后检查各层激活和梯度是否有限、实际更新是否发生, 以及学习率与输入尺度是否相容.

训练损失降低而验证损失上升, 是过拟合的常见证据, 但比较前必须统一计算口径. 训练时含 Dropout、数据增强或正则惩罚, 而验证时关闭随机扰动并只计算数据损失, 两条曲线并非同一统计量; 训练日志又可能只是滚动小批量均值, 验证则是完整留出集均值. 更严格的诊断是在相同评估模式下分别计算训练和验证的数据损失, 必要时按有效词元总数加权.

观察	需要区分的原因	有针对性的检查
损失非有限或持续爆炸	数值溢出、梯度错误、尺度过大	稳定损失形式、非有限值起点、分层梯度与更新幅度.
训练与验证损失均高	信息不足、容量不足或优化失败	简单基线、梯度核对、不同学习率与预算.
训练改善而验证退化	过拟合、分布差异或口径不同	相同评估模式、组别分布、早停与正则强度.
离线质量高而部署退化	泄漏、漂移或输入协议变化	时间可用性、实体隔离、预处理与模板版本.

有限差分能检查局部梯度，确定性小批量拟合能检查更新链路，独立验证能检查留出分布上的预测，部署监控能检查运行后的变化. 这些证据各自回答不同问题. 梯度范数很小既可能意味着接近驻点，也可能是激活饱和或信号消失；梯度裁剪只能限制更新前的幅度，不能把错误目标变为正确目标. [3]

习题参考结果　第 1 题稳定区间为 $0 < \eta < \dfrac{1}{4}$，平衡步长为 $\dfrac{1}{5}$；最大曲率方向在边界上等幅变号. 第 3 题在所述独立条件下标准差变为原来的一半，完全相关时平均不能降低随机波动. 第 4 题为 $\dfrac{\eta s^2}{[2B(2-\eta h)]}$. 第 5 题方差为 $\dfrac{2}{1.01 \cdot 256}$. 第 7 题若整体缩放 F，学习率除以 c 即可保持更新；若只缩放数据损失，还须把 λ 乘以 c. 第 9 题得到 $a^* = \dfrac{\mu^2}{\mu^2 + \frac{\sigma^2}{n}}$，它依赖未知总体参数.

[3] 用固定小批量检查能否拟合，是定位实现问题的诊断手段. 它不提供泛化质量证据，也不意味着应在实际训练中追求对所有噪声标签的完全记忆.

第十八章　训练数据工程基础

模型接触的不是抽象的“互联网知识”，而是经过采集、解析、筛选、变换和采样后形成的一组训练事件. 两个系统即使采用同一架构和相同的原始文件，也可能因为样本边界、去重规则、标签掩码和混合比例不同而优化不同的目标. 因此，**数据工程**（Data Engineering）既要保证处理链能够持续运行，也要解释每个训练词元为何进入数据集、代表什么分布，以及能否追溯到原始来源.

第四章《文本表示》已经完成数据主线的第一部分：从单条文本记录出发，定义样本、词元编号、特殊词元、批输入与嵌入表示. 本章进入第二部分，把视角从“一条样本怎样进入模型”扩大到“海量异构来源怎样成为可治理、可复现、可消费的训练版本”. 这些原则同时适用于预训练和后训练；第二十三章《预训练原理》进一步讨论预训练的数据配比与词元预算，第二十章、第二十五章、第二十六章分别展开指令、偏好和可验证推理数据. 因此，本章不重复分词算法本身，而以各训练阶段共用的原始资料到训练制品生产链为主线.

第八章《自回归语言建模》已经定义了条件概率与有效预测位置；这里进一步说明，数据系统如何决定这些位置. 数据质量并不等同于文件可读性，数据版本也不等同于目录名称. 只有将统计对象、语义结构、治理条件与物理存储分别建模，才能判断一次训练实际使用了什么.

本章的主要符号见表 18.1. 概率按记录还是按词元计算，将在使用处单独声明；容量统一用词元位置计量.

表 18.1: 主要符号与计量对象

符号	含义与范围
$P_\star, P_0, Q$	目标、原始采集及最终训练分布
A, B	用于比较的有限片段集合
$J(A,B), \tau$	Jaccard 相似度与确认阈值，取值 $[0,1]$
m, b, r	签名行数、带数、每带行数，$m = br$
n_i, p_i, q_i	来源规模、实际采样概率、目标风险权重
α	来源混合指数，本章取 $[0,1]$
L_i, M, T	样本长度、总长度、单块容量，单位为词元位置
K, R, P	块数、丢弃尾长、填充位置数

18.1　统计对象及数据分类

18.1.1　样本依赖性

设目标应用中的样本分布为 $P_\star$，原始采集分布为 P_0，处理链最终提供给训练器的分布为 Q. 训练风险 $\mathbb{E}_{z\sim Q}[\ell_\theta(z)]$ 与目标风险 $\mathbb{E}_{z\sim P_\star}[\ell_\theta(z)]$ 一般不同. 网站可访问性、抓取频率、语言识别器和长度筛选都会改变 Q. 数据量增加只能降低对当前采样过程的部分估计误差，不能自动消除分布偏移.

样本单位（Sampling Unit）需要随任务明确：文档、对话、问题家族、用户或时间窗口都是不同单位. 同一篇文档的多个切片、同一问题的不同改写、同一对话的多个轮次具有相关性. 把它们当作独立样本，会夸大有效信息量，也会使随机划分产生泄漏.

考虑 G 个相互独立的组，每组含 m 个观测，组内观测方差均为 σ^2，任意两个不同观测的相关系数均为 ρ，总数 $N = Gm$. 样本均值的方差为

$$\operatorname{Var}(\bar{Z}) = \frac{G[m\sigma^2 + m(m-1)\rho\sigma^2]}{G^2m^2} \tag{18.1}$$

$$= \frac{\sigma^2}{N}\,[1 + (m-1)\rho]. \tag{18.2}$$

相对于独立观测，**设计效应**（Design Effect）为 $1 + (m-1)\rho$，等效样本量为 $\dfrac{N}{[1+(m-1)\rho]}$. 例如，每个问题生成五个答案且 $\rho = 0.5$，一万个观测的均值方差相当于约 3333 个独立观测. 此式依赖等组大小、等相关及组间独立假设；它说明重复扩增可能增加存储和训练成本，却不同比例地增加信息.

18.1.2　六个可组合的分类维度

“这是 JSONL 数据”“这是推理数据”“这是监督微调数据”分别回答不同问题. 若把它们放入同一分类树，就无法表达一个 JSONL 文件同时包含数学领域、多轮消息和偏好样本. 应采用六个可以组合的维度.

1. **物理格式**描述字节如何编码和组织，例如纯文本、JSONL、CSV、Parquet、Arrow IPC 或 TAR. 格式影响压缩、随机访问和扫描成本，不定义训练目标.

2. **逻辑样本模式**描述记录的语义结构，例如文档、平行句对、指令与回答、消息序列、偏好对、强化学习提示或推理轨迹.

3. **消费目标**描述如何使用记录，例如分词器训练、预训练、监督微调、奖励学习、偏好优化、带可验证奖励的强化学习、蒸馏或评估.

4. **内容与能力**描述语言、领域以及代码、数学、工具使用、安全等能力属性. 一条记录可以具有多个标签；标签也可能带有置信度.

5. **模态**描述文本、图像、音频和视频等内容，以及跨模态对应关系．时间戳、图像区域和媒体引用属于语义的一部分．

6. **治理与血缘**描述来源快照、使用条件、允许用途、敏感信息、保留期限、质量结论和删除关系．

思维链（Chain of Thought，CoT）是推理过程的一种表达形式，不是物理格式，也不唯一决定损失函数．同一份推理轨迹可以产生仅学习最终答案的样本、学习完整轨迹的样本，或只保留问题供强化学习使用．消费目标改变时，应生成有明确版本的视图，不能直接修改原始记录来满足最后一个使用者．

18.1.3 类型化记录及适配

数据模式（Schema）定义字段类型、必要关系和约束．统一数据系统宜采用“公共信封加类型化载荷”：公共信封记录逻辑标识、来源、模式版本、语言、模态和治理信息；载荷由明确的类型标记选择文档、消息、偏好等结构．这样的**类型化记录**（Typed Record）可以同时检查公共约束和任务约束．

例如，一条偏好记录含公共提示、被偏好的回答和另一回答，二者必须在同一提示语境中可比较；它不能被当作两条无关联文本分别筛选．工具对话需要保存消息顺序、调用标识与返回结果之间的配对．推理记录应区分过程、最终答案和验证器结论；媒体记录应保留资源引用与对齐信息．把所有内容先展平为文本再尝试恢复结构，会丢失这些关系．

模式适配器（Schema Adapter）负责从指定来源版本转换到统一模式．它必须声明输入版本、输出版本、字段映射和失败条件．未知模式进入隔离队列，不能根据几个字段名猜测语义后继续训练．常见指令格式与消息格式即使名称相同，也可能在角色、工具消息或媒体字段上存在版本差异．[1]

记录标识 `record_id` 与分组标识 `group_id` 也应分离：前者标识逻辑记录，后者表达不能跨越训练与评估边界的家族．不同来源的局部编号可能相同，因此记录定位至少包含来源数据集标识和局部记录标识．重复来源和派生视图形成多对多关系，而非一条可覆盖的“来源字符串”．

18.2 数据版本构建

18.2.1 数据层及控制层

数据血缘（Data Lineage）记录对象由哪些父对象经过哪些变换产生．图 18.1 把字节处理与控制信息分开：数据层负责变换内容，控制层负责确定变换为何被允许、

[1] 面向模型接口的消息结构与数据仓库的归档结构职责不同．归档可以保留训练器不消费的治理字段，但不能因为某个接口不接受这些字段而删除原始语义．

采用哪个版本以及何时可发布.

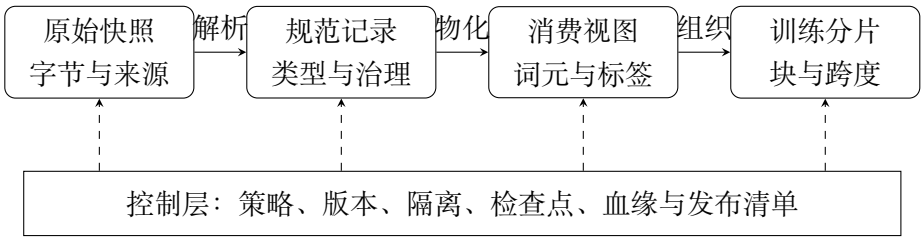


图 18.1: 原始数据、语义记录与训练制品分层. 虚线表示控制约束，实线表示内容变换.

原始层保存可追溯的快照，规范层保存经过解析并保持语义的记录，训练层保存针对特定分词器、目标和布局物化的制品. 三者不应互相替代：训练词元通常不能恢复解析前的字节，清洗文本也未必能够重建媒体、许可记录和原始边界. 保留层级不意味着无限保存；各层仍应遵循明确的保留与删除策略.

18.2.2　确定性及不同种类的哈希

确定性变换（Deterministic Transformation）要求在输入及显式配置相同的条件下产生相同结果. 配置包括代码、模式适配器、规范化规则、词表、特殊词元、随机种子以及所有影响内容的外部模型版本. 只保存种子而不固定输入顺序、随机数算法和并发归并规则，不能保证输出一致.

必须区分原始字节哈希、提取文本哈希、规范记录哈希、分词器输入哈希和去重规范化哈希. 它们回答不同的相等性问题. 去重时忽略空白，可能适合普通散文，却会破坏代码缩进或表格含义. 规范记录的哈希需要固定序列化规则，例如字段排序、编码、空值与数字表示；否则语义相同的对象可能得到不同摘要.

密码学哈希可以检测相对于可信摘要的内容变化，但不能证明来源允许使用，也不能单独证明清单的真实性. 如果攻击者同时替换内容与摘要，内部一致性仍然成立. 可信注册表或签名等机制属于控制层；不能用一个内容指纹替代来源治理.

18.3　质量、隐私及评估污染

18.3.1　过滤后的学习分布

设输入样本分布为 $P(z)$，样本以概率 $a(z) \in [0,1]$ 被接受，则接受后的分布为

$$Q(z) = \frac{a(z)P(z)}{\mathbb{E}_P[a(Z)]}, \qquad 0 \leqslant a(z) \leqslant 1, \quad \mathbb{E}_P[a(Z)] > 0. \tag{18.3}$$

若平均接受率为零，处理结果是空数据集，不能定义上述归一化分布；发布流程应报告无可用样本并停止该分支. 这表明质量筛选既减少不合格内容，也重新加权语言、

领域和表达方式. 若质量评分器偏好某一种文体，保留率提高并不必然代表目标能力提高. 阈值应按记录类型和目标用途解释：代码的有效性、对话的角色一致性、偏好对的可比较性与网页的正文密度不是同一个量.

解析、规范化、敏感信息处理和质量筛选应具有分开的阶段及原因码. 解析失败意味着无法可靠解释结构；质量拒绝意味着结构可解释但不满足某项条件. 将所有失败统称为“坏数据”，会使源格式升级与真实质量变化无法区分. 统计时还应声明分母：源记录保留率、规范记录保留率和有效词元保留率可能给出截然不同的结论.

个人可识别信息（Personally Identifiable Information，PII）检测可以结合规则、实体识别和上下文判定，但模式匹配只能覆盖已定义模式. 替换电话号码不等于匿名化，多个非敏感字段组合也可能重新识别个人. 审计日志应保留稳定原因和定位信息，避免把被移除的秘密再次写入普通日志. 对敏感记录采用隔离或删除后，还需要同步处理其派生视图.

18.3.2 污染及重复

评估污染（Evaluation Contamination）指评估信息通过训练、调参或数据选择进入被评估系统，使结果不能解释为预期的独立泛化. 问题全文、答案、解释、翻译和近似改写都可能构成信息通道. 仅检查整条记录的哈希相等，只排除了很窄的一类重叠.

设评估题的片段集合为 A，训练文档为 B. 当短题被嵌入长文档时，Jaccard 相似度可能很低，而**包含率**（Containment）$C(A,B)=\dfrac{|A\cap B|}{|A|}$ 仍为 1. 因此污染检查需要与任务匹配的粒度：整记录、片段、问题家族以及答案相关内容. 包含率与相似度的区分见 Broder 的原始研究 [109].

污染检查结论应表述为“在所用检测方法、阈值和参考集合下未发现匹配”，不能表述为没有污染. 保留评估参考集的访问边界也很重要：不能为了检测而将答案完整写入通用训练日志或生成提示. 数据去重研究显示训练重复与记忆、训练评估重叠存在值得控制的联系 [110]；这不意味着去重能够消除全部隐私风险或评估偏差.

18.4 精确去重及近似去重

18.4.1 精确相等的判定对象

精确去重（Exact Deduplication）先对指定对象进行确定性规范化，再比较其摘要；需要排除碰撞时进一步比较规范化字节. 对象可以是文档、对话或完整偏好对. 对结构化记录只比较回答文本，会把语境不同的合法样本误判为相同.

重复记录的删除应保留所有来源别名与治理关系. 一个内容在多个位置出现，并不使这些来源的使用条件自动相同. 代表记录的选择需明确优先级，例如允许用途、

质量、来源可信度、时间与稳定标识的顺序. 不能让并发完成顺序决定保留者，否则同一输入会产生不同训练版本.

18.4.2　MinHash 去重

把规范文本切成长度为 q 的连续片段，称为**连续片段**（Shingle），得到集合 A. 这里采用集合而非多重集，重复片段出现次数不增加权重. 两条记录的**Jaccard 相似度**（Jaccard Similarity）为

$$J(A,B)=\frac{|A\cap B|}{|A\cup B|}. \tag{18.4}$$

q 太小会使常用表达大量重合，太大则对局部改写敏感. 空集合需要另行约定，例如送入短文本专门处理；不能在公式中执行 $\frac{0}{0}$.

最小哈希的理想化模型

片段来自有限全集，$A\neq\varnothing$ 且 $B\neq\varnothing$；空集合已由前述短文本流程单独处理. 每次独立地对全集作均匀随机排列 π，并以排列次序最小的片段作为集合签名. 推导暂不考虑有限哈希值的碰撞及非理想哈希族偏差.

最小哈希（MinHash）用 $h_\pi(A)=\arg\min_{x\in A}\pi(x)$ 表示集合. 两个最小值相同，当且仅当 $A\cup B$ 中排列最靠前的元素落在 $A\cap B$ 中. 并集内每个元素成为最小者的概率相等，因此

$$\Pr[h_\pi(A)=h_\pi(B)]=\frac{|A\cap B|}{|A\cup B|}=J(A,B). \tag{18.5}$$

该关系把集合相似度转化为签名碰撞概率 [109]. 采用 m 次独立排列，令 I_j 为第 j 个签名相等的指示量，则

$$\widehat{J}=\frac{1}{m}\sum_{j=1}^{m}I_j,\qquad \mathbb{E}[\widehat{J}]=J,\qquad \operatorname{Var}(\widehat{J})=\frac{J(1-J)}{m}. \tag{18.6}$$

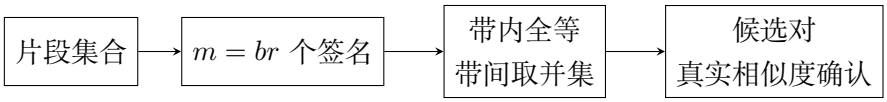
例如 $J=0.8, m=100$ 时标准差为 0.04. 它是随机估计的不确定性，不是一个样本对的确定误差上限. 工程上使用哈希函数近似随机排列，必须意识到函数相关性、摘要截断和碰撞可能改变上述理想性质.

18.4.3　局部敏感哈希候选集

对 N 条记录逐对计算相似度需要 $O(N^2)$ 次比较. **局部敏感哈希**（Locality-Sensitive Hashing，LSH）把签名分成 b 个带，每带 r 行，满足 $m=br$；同一带内所有行相同的记录进入同一候选桶. 若两记录相似度为 s，在独立行假设下，一个带完全相同的概率为 s^r，所有带均不同的概率为 $(1-s^r)^b$，所以成为候选的概率为

$$P_{\text{cand}}(s)=1-(1-s^r)^b. \tag{18.7}$$

取 $b=20, r=5$，$s=0.8$ 时候选概率约为 0.99964；$s=0.2$ 时约为 0.00638．概率曲线并不是硬阈值：相似度高的记录仍可能漏检，低相似度记录也可能进入候选．



候选召回控制计算量；确认阈值控制候选中的误判

图 18.2: 近似去重的候选生成与精确确认承担不同职责．

对候选计算真实 Jaccard 可以排除候选中的误报，但不能找回从未进入候选的漏报．候选桶中有 u 条记录时，完全展开仍产生 $\dfrac{u(u-1)}{2}$ 对．常见模板会形成热点桶，限制桶大小虽能保护资源，却改变召回率；这种限制必须作为策略版本的一部分记录，而不能隐藏成实现细节．

18.4.4　近重关系的非传递性

令 $A=\{1,2\}$, $B=\{1,2,3\}$, $C=\{2,3\}$, 阈值取 $\dfrac{2}{3}$. 此时 $J(A,B)=J(B,C)=\dfrac{2}{3}$, 但 $J(A,C)=\dfrac{1}{3}$．如果把近重边的连通分量全部只留一条，就可能删除与代表记录并不相似的内容．

分组和删除需要不同语义．为了防止泄漏，可以保守地把整个关联分量放入同一个划分；为了删除内容，则可按稳定优先级扫描，只有与已经保留的某个代表直接达到阈值的记录才删除．该策略并非求最小代表集合，也可能保留多个彼此通过中间节点相关的记录；它换取的是可解释的直接删除依据．

算法 18.1　稳定代表去重

输入：类型一致的记录、固定规范化规则、相似度阈值 τ、候选索引及稳定全序．**输出：**保留记录与删除依据．**状态：**保留集合 R，重复映射 D．

1. 按治理资格、质量和稳定标识确定全序；初始化 $R=D=\varnothing$.
2. 依次读取记录 x，从候选索引取得已在 R 中的候选，按同一全序排列．
3. 对每个候选 y 计算实际相似度；若有 $J(x,y)\geqslant\tau$，记录直接依据 (x,y,J)，并保留 x 的所有来源血缘．
4. 若没有确认匹配，将 x 加入 R．处理完所有记录后停止．

不变量：每条因近重被删除的记录都有一个直接达标且仍被保留的代表．候选漏检仍可能导致重复保留，必须单独评估召回．

18.5　划分及混合定义训练分布

18.5.1　分组哈希及时间边界

分组划分（Group Split）要求同一个泄漏家族只进入训练、验证或测试中的一个集合. 设稳定组标识为 g，固定盐值为 s_0，把哈希映射到 $u(g) \in [0,1)$. 例如 $u < 0.8$ 分配到训练，$0.8 \leqslant u < 0.9$ 分配到验证，其余进入测试. 同一组的新增记录仍进入同一集合；与按当前行号洗牌相比，它更适合增量数据.

这里稳定的是组到集合的映射，不是精确比例. 组大小不等时，样本比例和词元比例都可能偏离 $80:10:10$. 哈希还必须与目标变量及来源顺序没有系统性关联. 固定盐值是可复现配置，不是为得到更好测试成绩而不断调整的超参数.

如果新增近重边把两个已经落入不同集合的组连接起来，原划分的隔离条件已经失效. 不能既保持所有旧分配又宣称严格分组隔离；需要隔离新增冲突、重建划分或发布新的评估版本. 面向未来数据的应用还应采用时间边界，确保训练时不可见的信息没有进入特征构造. 分词器学习、统计归一化和基于语料的阈值拟合也属于需要限定训练范围的过程.

18.5.2　温度混合及目标校正

设清洗后有 D 个合格来源，第 i 个来源具有 $n_i > 0$ 个采样单位. **温度混合**（Temperature-Based Mixing）常写为

$$p_i = \frac{n_i^{\alpha}}{\sum_{j=1}^{D} n_j^{\alpha}}, \qquad 0 \leqslant \alpha \leqslant 1. \tag{18.8}$$

$\alpha = 1$ 保持自然数量比例，$\alpha = 0$ 等概率选择来源，较小的 α 提高小来源的相对权重. 式中的 n_i 必须声明是文档数、词元数还是其他单位，不能在同一配置里混用.

例如两个来源分别有 900 和 100 条记录，取 $\alpha = \dfrac{1}{2}$，则 $p = (0.75, 0.25)$. 若有放回抽取 1200 次，期望分别抽取 900 和 300 条，单条记录平均使用次数分别为 1 和 3. 若只对各来源各读一遍并拼接，实际记录比例仍为 $0.9:0.1$，并未实现指定目标.

假设两个来源平均长度分别为 100 和 400 个有效词元，按上述文档概率采样，则长期词元份额近似为

$$\widetilde{p}_i = \frac{p_i \mathbb{E}[L_i]}{\sum_j p_j \mathbb{E}[L_j]}, \tag{18.9}$$

得到 $\widetilde{p} = (\dfrac{3}{7}, \dfrac{4}{7})$. 因此目标配比、抽中文档配比、物化词元配比和最终有效标签配比必须分别统计. 截断、尾部丢弃和标签屏蔽会使它们进一步不同.

若真正希望优化来源权重 q_i，却按 p_i 抽样，可以采用**重要性采样**（Importance

Sampling）校正：对来自 i 的损失乘以 $\frac{q_i}{p_i}$. 因为

$$\sum_i p_i \,\mathbb{E}_i\left[\frac{q_i}{p_i}\ell_\theta\right] = \sum_i q_i \mathbb{E}_i[\ell_\theta], \tag{18.10}$$

期望目标恢复为指定加权风险. 前提是 $q_i > 0$ 时 $p_i > 0$，并且期望存在；极小的 p_i 会产生大权重和高方差. 若本来就希望改变学习目标，则不应再无意识地施加该校正.

课程学习（Curriculum Learning）让 p_i 随训练步变化. 它改变参数经历的样本顺序与梯度路径，不能仅用最终累计比例描述. **合成数据**（Synthetic Data）同样需要来源问题家族、生成模型、提示、解码配置和验证器版本. 生成器自评筛选可能放大其偏好，验证器通过也只证明所检验的条件；评估集应与生成、筛选和阈值调整隔离.

18.6 词元物化、打包及存储

18.6.1 训练样本消费视图

词元物化（Token Materialization）把规范记录转换为固定词元与监督信号. 它需要冻结分词器、特殊词元、对话模板、截断规则、标签策略和位置处理. 仅记录词表名称不足以确定结果. 对于偏好对，共同提示的处理必须一致；对于推理样本，应显式决定过程是否参与损失.

训练视图至少定义输入词元、有效预测位置和必要的边界信息. 自回归目标预测的是下一词元，输入与目标的错位必须与样本边界一致. 一个序列长度为 L，并不意味着必有 L 个有效标签：提示词元、填充、边界和被屏蔽的内容都可能不计入损失.

18.6.2 连续流及不可拆样本的容量公式

序列打包（Sequence Packing）把变长内容组织为长度上限为 T 的模型输入块. 首先区分允许跨块切分的连续流与不允许切分的原子样本. 令 L_i 已包含规则要求的分隔符或边界词元，总长度 $M = \sum_i L_i$.

对连续流，若尾部不足一块就丢弃，则

$$K_{\text{drop}} = \lfloor \frac{M}{T} \rfloor, \qquad R = M - K_{\text{drop}}T, \qquad 0 \leqslant R < T. \tag{18.11}$$

若保留尾部并填充，则对于 $M > 0$，

$$K_{\text{pad}} = \lceil \frac{M}{T} \rceil, \qquad P = K_{\text{pad}}T - M. \tag{18.12}$$

空输入约定零块. 这些公式先由输入长度确定块数，再推导余量，不把未知填充量重新放入块数定义形成循环.

如果每条样本不可拆分，$L_i \leqslant T$，则 $\lceil \frac{M}{T} \rceil$ 只是块数下界，实际值由装箱策略决定．例如三个长度均为 6 的样本，$T = 10$，总长 18 给出下界 2，但任意两条都无法同块，必须使用 3 块．超过 T 的样本还需要明确选择截断、切分或隔离，不能留给隐式溢出行为．

长度为 $(5, 4, 3)$、$T = 8$ 时，按顺序采用放不下便开启下一块的策略，得到第一块长度 5、第二块长度 7，共填充 4 个位置，内容利用率为 $\frac{12}{16} = 75\%$．连续流丢尾则生成一块、丢弃四个词元，已生成块利用率为 100%，但数据损失率为 $\frac{1}{3}$．只汇报利用率会掩盖这一差别．特殊词元占用率与有效标签占用率还应另外统计．

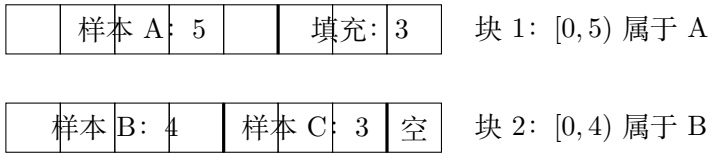


图 18.3: 不可拆样本的顺序打包示意．每格为一个位置，粗线是来源边界；是否允许跨边界注意力仍由掩码策略决定．

18.6.3　样本隔离约束

把多个文档放入一块，不代表它们必须互相可见．连续预训练流可以显式允许跨文档条件关系；若目标要求样本独立，则需要块内分段因果掩码．令 $g(t)$ 为位置 t 所属样本，允许位置 t 读取 j 的条件为 $j \leqslant t$ 且 $g(j) = g(t)$，填充位置另行处理．第五章《注意力机制》讨论了掩码归一化，包括全屏蔽行的边界．

仅插入结束词元并不会阻断注意力．标签也必须隔离：若前一样本的末位置被要求预测下一样本的首词元，即使注意力已隔离，损失仍引入跨样本预测．可以在每个样本加入开始标记，由其预测本样本首词元，或屏蔽没有合法本样本前缀的预测位置．位置编号是否重置应写入布局规范．[2]

算法 18.2　保持来源的原子样本打包

输入：按确定顺序排列的已序列化样本，最大长度 T，超长与尾部策略．**输出：**词元块、标签掩码、样本段及来源跨度．**状态：**当前块 B、已用长度 u、跨度表 S．

1. 初始化空块．读取下一样本，先执行已声明的超长策略，使进入打包器的原子样本长度不超过 T．
2. 若 $u + L_i > T$，按尾部策略封存当前块，记录填充或丢弃量；重新建立空块．
3. 追加样本，记录半开区间 $[u, u + L_i)$、来源记录与源词元偏移，并设置段标

[2]打包后的注意力掩码在概念上可以是分块矩阵；物理实现可以使用段边界等紧凑表示．本章规定语义，不要求显式存储一个稠密平方矩阵．

识和合法标签；更新 u.

4. 输入耗尽时按同一尾部策略处理最后一块，停止.

不变量： 块长不超过 T；每个非填充位置能定位到来源或明确标记为插入词元；样本边界与标签策略一致. 按顺序一次遍历的组织成本为 $O(\sum_i L_i)$，不含分词成本.

18.6.4 来源跨度及分片边界

来源跨度（Provenance Span）将块中的半开区间映射到来源数据集、记录、源偏移以及变换版本. 填充应有空来源，插入的特殊词元应注明生成规则，不能伪装为原文词元. 跨度可支持训练贡献统计、错误定位与删除影响分析；只保存“本块来自三个文档”不足以定位具体内容.

分片（Shard）是存储或调度单位，**训练块**（Training Block）是模型输入单位，两者不能等同. 一个分片可以容纳多个完整块，也可以在另一种明确格式中保存尚未打包的文档. 若规定分片只含完整 T 长块，容量上限为 S 个位置，则最多容纳 $\lfloor \frac{S}{T} \rfloor$ 块，且要求 $S \geqslant T$. 例如 $S = 18, T = 8$，只能放两块即 16 个位置；余下两个位置不能容纳第三块. 文件结束也不是文档结束或训练序列结束.

18.7 流式执行、发布及删除

18.7.1 读取状态及数据消费

流式读取（Streaming）降低预先装载需求，但不消除输入输出成本. 内存映射仍受页缓存、存储吞吐和访问局部性影响. 有限缓冲区洗牌只在局部窗口混合，不等价于全量均匀排列. 若某些来源更慢，而调度器总选择先完成的数据，目标混合也可能被吞吐差异改变.

恢复点必须覆盖分片位置、记录偏移、随机数状态、洗牌缓冲、未封存词元以及训练器已经提交的进度. 只恢复文件游标，可能重复或跳过缓冲区中的样本. 改变工作进程数或分片分配方式，也可能改变顺序与尾部结果. **恰好一次**（Exactly-Once）在这里应指逻辑训练事件的提交语义，而不是文件只被读取一次；是否重复更新参数，取决于数据消费状态与训练检查点如何一致提交.

遇到坏记录静默跳过会改变分布，重试直到成功又可能阻塞整个输入. 正确做法是固定失败分类和隔离策略，计量其来源分布，并把最终纳入或排除的决定记录到版本中. 批处理适合生成不可变制品，增量处理适合吸收新增快照；二者都需要相同的语义规则.

18.7.2　幂等发布及版本身份

幂等性（Idempotency）表示同一逻辑构建重复提交不会产生不同的可见结果. 构建标识可以由源快照集合、模式与适配器、代码和策略版本、分词与布局配置的规范表示计算得到. 它表示构建规格；最终内容摘要表示构建产物，两者应分开，以避免让清单把自己的摘要纳入计算而形成自引用.

发布清单（Manifest）列出制品、摘要、记录与词元计数、来源范围、策略版本和必要血缘. 中间文件先写入隔离的暂存空间，完整制品准备后才发布清单或更新可见指针. 单一文件系统中的原子重命名与对象存储中的条件指针更新具有不同条件；不能把多个对象依次上传称作整体原子发布.

算法 18.3　可重试的数据版本发布

输入：不可变源快照与完整构建规格. **输出：**可消费版本标识. **状态：**构建标识、暂存制品列表、阶段完成记录.

1. 计算构建标识；若已有已发布版本，比较规格与产物登记，匹配则复用，否则报告一致性冲突.
2. 按阶段生成不可变制品，记录父对象与完成状态. 重试只复用经过完整性确认的阶段结果，不把“文件存在”视为完成.
3. 生成独立内容摘要和清单，记录成功、拒绝、隔离、派生及有效标签数量；所有必需对象就绪后进入发布步骤.
4. 以平台支持的原子提交或条件更新暴露最终指针. 并发提交者若发现版本已经发布，核对后复用；同一规格产生不同内容时停止发布并保留诊断依据.

不变量：消费者只能发现完整版本；同一版本标识不覆盖为另一内容. 失败发生在发布前时可重试，发布后则返回已存在的版本.

内容摘要可以发现字节变化，但如果攻击者能同时替换制品与清单，自声明摘要不能证明来源真实. 发布目录应把认可的摘要保存在独立信任边界内，或者由受控签名及验证密钥建立真实性. 公开清单通常记录聚合数与受控审计资产的引用，不直接暴露隔离原文、个人标识或推理轨迹；细粒度血缘的可追溯性必须与访问权限同时设计.

对于一进一出的阶段，输入记录数应等于成功、拒绝与隔离数之和；对于一条文档切成多块或多条记录合并的阶段，简单行数守恒不再成立，必须使用父子映射和词元账目. 最大处理编号也不能证明之前所有编号都完成，特别是在并发执行时.

18.7.3 删除传播及模型影响

删除血缘（Deletion Lineage）用于从撤回的源记录定位规范记录、消费视图、词元跨度、分片以及使用过这些制品的训练任务. 精确去重后如果删除了来源别名，这条链就会断裂；保留一个代表不意味着其他来源不再需要治理.

删除请求首先阻止未来消费，并根据策略重建或撤销受影响的数据版本. 已经完成的训练还需要单独评估模型影响：删除存储对象不会自动从模型参数中消除其作用，重新发布数据也不等同于完成模型层面的遗忘. [3]

可追溯性闭环

一个训练数据版本至少应回答四个问题：哪些来源允许进入；哪些变换决定了内容与标签；训练器实际消费了什么；某个来源失效后哪些产物需要处理. 可读取的文件、正确的哈希和成功结束的任务分别提供局部证据，任何一个都不能独自证明闭环成立.

18.8 数据管线设计

考虑两个通过治理筛选的来源，共 900 条短记录与 100 条长记录. 采用式 (18.8) 中的 $\alpha = \dfrac{1}{2}$，按文档有放回抽样 1200 次，期望使用次数为 900 与 300. 若序列化后长度恒为 100 与 400，则期望物化长度为 210000 个词元；其中长来源贡献 120000，即 $\dfrac{4}{7}$.

现在取一个具体已经实现该次数的采样序列，按连续流以 $T = 1024$ 打包. 保留尾部时有 $\lceil \dfrac{210000}{1024} \rceil = 206$ 块，容量为 210944，填充 944 个位置. 丢尾时有 205 块，保留 209920 个词元，丢弃 80 个. 由于尾部来自具体排序中的记录，其来源份额不一定仍为 $\dfrac{4}{7}$. 若改成不可拆记录，这两个块数公式便不再直接适用.

进一步假设标签策略屏蔽所有提示，共屏蔽 42000 个实际词元；在保尾条件下有效标签最多为 168000，还需扣除策略要求屏蔽的边界位置. 按块容量统计的有效标签率最多为 $\dfrac{168000}{210944} \approx 79.64\%$，低于内容利用率约 99.55%. 这些是给定长度与计数的算术结果，不是训练性能实验.

这组计算揭示了数据审计应区分的四个分母：源记录、被抽中的记录、实际词元和有效标签. 除此之外，去重候选召回、隔离率、来源跨度覆盖率、重试产生的逻辑重复量和版本完整性也需要分别解释. 一个总“质量分”无法替代这些具有不同因果含义的指标.

[3] 用于防止再次导入的删除标记也可能含有可关联标识，其保留形式与权限应单独设计；不能为了证明已删除而无限保留原始敏感内容.

第十九章　训练稳定性

目标函数规定应当学习什么，优化过程决定参数如何接近这一目标．对于持续数日甚至更久的训练，优化过程还必须能够在有限精度下稳定运行，并在中断后保留其历史．相同的模型权重、不同的优化器状态，通常会产生不同的下一步更新．因此，训练的基本对象是一组随时间共同演化的状态，而不仅是参数矩阵．

第十七章《优化及泛化》讨论梯度方法与泛化的基础．本章进一步推导自适应更新，明确调度和裁剪的执行顺序，并建立可恢复训练的状态模型．并行状态如何划分以及如何通信，留给第二十九章《分布式训练》．

19.1　动量优化

19.1.1　符号及更新边界

本章的计量约定

将全部可训练参数展平为 $\theta \in \mathbb{R}^P$，但实际存储仍可保持各张量的形状．第 t 次成功更新前的参数为 θ_{t-1}，t 从 1 开始．g_t 是该次更新全部有效监督词元上的平均损失梯度；不等长微批次的加权聚合见第二十九章《分布式训练》．所有向量乘除、平方和开方均按元素进行，范数另行标出．推导先使用精确算术，随后再讨论有限精度偏差．

本章主要符号见表 19.1.

表 19.1: 训练稳定性的主要符号与计量对象.

符号	含义与范围
P, θ_{t-1}	可训练标量数，以及更新前的 P 维参数向量.
g_t, m_t, v_t	当前梯度、一阶矩估计与二阶原点矩估计，均为 P 维.
β_1, β_2	两种指数平均的衰减系数，取值在 $[0, 1)$.
$\eta_t, \lambda, \epsilon$	学习率、权重衰减系数和分母稳定项；前两者非负，$\epsilon > 0$.
c, s	正的梯度范数阈值与损失缩放因子.
t, a, n	成功更新次数、尝试更新次数、已消费有效目标词元数．三者不必同步增长.

动量（Momentum）利用历史梯度抑制方向上的快速摆动. 采用指数平均形式，令 $m_0 = 0$，则

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t = (1-\beta_1) \sum_{i=1}^{t} \beta_1^{t-i} g_i. \tag{19.1}$$

展开式表明，第 i 次梯度的权重随时间间隔指数衰减. 它不是最近若干步的等权平均. 权重总和为 $1-\beta_1^t$，因此从零初始化的平均在早期被向零压缩.

假设仅为分析初始化效应而有 $\mathbb{E}[g_i] = \mu$，由期望的线性性可得

$$\mathbb{E}[m_t] = (1-\beta_1^t)\mu, \qquad \widehat{m}_t = \frac{m_t}{1-\beta_1^t}. \tag{19.2}$$

于是 $\mathbb{E}[\widehat{m}_t] = \mu$. 这个结论不要求各步梯度独立，但要求它们的期望相同. 实际训练中参数不断变化，梯度分布也随之变化；此时修正消除了零初始化导致的权重总和不足，不能保证得到当前梯度期望的无偏估计.

19.1.2 Adam 的自适应尺度

自适应矩估计（Adaptive Moment Estimation，Adam）同时维护平方梯度的指数平均 [111]：

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(g_t \odot g_t), \qquad v_0 = 0, \tag{19.3}$$

$$\widehat{v}_t = \frac{v_t}{1-\beta_2^t}, \qquad \theta_t = \theta_{t-1} - \eta_t \frac{\widehat{m}_t}{\sqrt{\widehat{v}_t} + \epsilon}. \tag{19.4}$$

v_t 估计二阶原点矩，不是减去了均值平方的方差. 分母按照各坐标过去的梯度尺度调节步长. 它是对角形式的预条件更新，不包含坐标之间的曲率耦合，也不等同于逆 Hessian 方法.

为理解分母的作用，考虑某坐标每一步的梯度均为常数 $g \neq 0$.偏差修正后 $\widehat{m}_t = g$、$\widehat{v}_t = g^2$，该坐标的更新量为

$$\Delta\theta_t = -\eta_t \frac{g}{|g| + \epsilon}. \tag{19.5}$$

当 $|g| \gg \epsilon$ 时，其幅度接近学习率，方向与梯度相反. 这说明 Adam 的更新幅度不能由原始梯度范数单独推断；当历史梯度方向反转时，一阶矩还可能暂时保留旧方向.

ϵ 的位置属于算法定义. $\sqrt{v} + \epsilon$ 与 $\sqrt{v + \epsilon}$ 一般不相等. 若把式 (19.4) 写成未修正矩的形式，需要同时变换稳定项：

$$\frac{\widehat{m}_t}{\sqrt{\widehat{v}_t} + \epsilon} = \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} \frac{m_t}{\sqrt{v_t} + \epsilon\sqrt{1-\beta_2^t}}. \tag{19.6}$$

只移动偏差修正到学习率而保持原稳定项，会改变更新，尤其是在平方梯度很小时. 比较实现时应比较完整公式，不能只比较优化器名称.

19.2　解耦权重衰减

给定经验损失 $\mathcal{L}(\theta)$，添加 $\dfrac{\lambda}{2}\|\theta\|_2^2$ 后，梯度变为 $g+\lambda\theta$. 对无动量的普通梯度下降，

$$\theta_t=\theta_{t-1}-\eta_t(g_t+\lambda\theta_{t-1})=(1-\eta_t\lambda)\theta_{t-1}-\eta_t g_t. \tag{19.7}$$

此时二次惩罚与按学习率缩放的乘性衰减具有相同代数表达. 将 $g_t+\lambda\theta_{t-1}$ 送入 Adam 时，惩罚还会进入 m_t 和 v_t，并被各坐标不同的历史尺度调整，因而不再等价于上述乘性衰减.

解耦权重衰减（Decoupled Weight Decay）将这两个动作分别定义 [112]. 本章采用的 AdamW 为

$$\theta_t=(1-\eta_t\lambda)\theta_{t-1}-\eta_t\frac{\widehat{m}_t}{\sqrt{\widehat{v}_t}+\epsilon}, \tag{19.8}$$

其中矩估计仅接收数据损失的梯度. 多个参数组可有不同的 λ 或学习率. 偏置与归一化尺度常被排除在衰减组之外，但这是一项需记录的建模选择，不是所有模型都必须遵守的定理. 共享权重只应注册和更新一次.

当数据梯度恒为零、矩也为零时，连续 T 步后

$$\theta_T=\left[\prod_{t=1}^{T}(1-\eta_t\lambda)\right]\theta_0. \tag{19.9}$$

若每步 $\eta_t\lambda$ 很小且非负，利用 $\log(1-z)\approx -z$ 得到近似衰减因子 $\exp(-\lambda\sum_t\eta_t)$. 因此，即使衰减系数不变，改变日程总长度也可能改变累计收缩. 所谓“解耦”不表示实际衰减效果完全独立于学习率日程.

19.2.1　两步参数更新

例 17.1

取 $\theta_0=(1,-2)$，$\beta_1=0.9$、$\beta_2=0.99$、$\eta_1=\eta_2=0.1$、$\lambda=0.01$，并令梯度序列为 $g_1=(2,-1)$、$g_2=(0,3)$. 为清楚显示中间量，本例分母均非零，纸面计算取 $\epsilon=0$；实际算法仍保留正稳定项.

第一步有 $m_1=(0.2,-0.1)$、$v_1=(0.04,0.01)$，修正后为 $(2,-1)$ 和 $(4,1)$. 因此

$$\theta_1=0.999(1,-2)-0.1(1,-1)=(0.899,-1.898).$$

第二步有

$$m_2=(0.18,0.21),\qquad v_2=(0.0396,0.0999),$$

$$\widehat{m}_2=(0.947368,1.105263),\qquad \widehat{v}_2=(1.989950,5.020101).$$

自适应方向约为 $(0.67158, 0.49330)$，故

$$\theta_2 \approx 0.999(0.899, -1.898) - 0.1(0.67158, 0.49330) \approx (0.83094, -1.94543).$$

第一坐标的当前梯度为零，却仍因历史一阶矩而更新；第二坐标经历梯度符号反转，一阶矩从负转正．若只加载 θ_1 并把矩清零，这一步不可能得到相同结果．

19.3　学习率日程及训练时钟

学习率预热（Learning Rate Warmup）在训练初期逐步增大学习率；**余弦衰减**（Cosine Decay）在后续阶段平滑减小学习率．它们规定更新尺度随训练进度的变化，不能替代对错误标签、数值溢出或损失归一化错误的诊断．

为避免端点歧义，令计划成功更新次数为 S，预热次数满足 $1 \leqslant W < S$，第 t 次更新使用

$$\eta_t = \begin{cases} \dfrac{\eta_{\max} t}{W}, & 1 \leqslant t \leqslant W, \\ \eta_{\min} + \dfrac{\eta_{\max} - \eta_{\min}}{2}\left[1 + \cos\left(\pi \dfrac{t - W}{S - W}\right)\right], & W < t \leqslant S. \end{cases} \tag{19.10}$$

该定义在 $t = W$ 达到峰值，在 $t = S$ 达到 $\eta_{\min}$．例如 $S = 6, W = 2, \eta_{\max} = 0.1, \eta_{\min} = 0$ 时，六次学习率依次为 $0.05, 0.1, 0.085355, 0.05, 0.014645, 0$．最后一次梯度可以被计算而不改变参数，因此实际预算必须说明是否保留这一零学习率终点．其他日程可以在最后一次有效更新之后才达到零，两种约定不可混记．

调度器的时钟必须明确

读取一个微批次、尝试一次更新、成功更新参数和消费一批有效词元是不同事件．调度依据成功更新次数时，数值溢出导致的跳步不推进日程；依据消费词元数时，跳步是否消费数据以及日程如何推进必须另行定义．记录一个含义不明的“step”不足以恢复训练．

当批量或有效序列长度随时间变化时，按词元数调度可以直接对应数据预算，但不保证与按步调度产生相同参数轨迹．改变总预算 S 后重新构造余弦曲线，也会改变恢复点之后的学习率；这属于更改训练方案，应保存新的日程版本．

19.4　有限精度、损失缩放及梯度裁剪

19.4.1　数值异常定位

有限精度同时限制可表示范围和有效数字．前向矩阵乘法溢出、归一化统计不准确、概率对数的下溢以及梯度累加误差，是不同位置的问题．交叉熵宜通过稳定的

Log-Sum-Exp 计算：

$$\log \sum_j e^{z_j} = z_{\max} + \log \sum_j e^{z_j - z_{\max}}. \tag{19.11}$$

右式的指数输入不为正，从而避免对巨大正数直接取指数. 若输入本身已含非有限值，这个恒等改写不能恢复丢失的信息.

损失缩放（Loss Scaling）令 $\widetilde{\mathcal{L}} = s\mathcal{L}$. 在精确算术下，反向得到 $\widetilde{g} = sg$，更新前除以 s 即恢复原梯度. 在低精度中，放大可以使某些过小梯度免于下溢，但也可能使较大梯度溢出. 动态策略因而根据非有限梯度反馈调整 s，并在发生溢出时跳过更新. 缩放并不能修正前向激活已经发生的溢出，也不能弥补不适合该运算的精度范围.

19.4.2　范数裁剪的几何意义

梯度范数裁剪（Gradient Norm Clipping）以阈值 $c > 0$ 定义

$$g'_t = \min\left(1, \frac{c}{\|g_t\|_2}\right) g_t, \tag{19.12}$$

零梯度保持为零. 这个向量是 g_t 到闭球 $\{u : \|u\|_2 \leqslant c\}$ 的欧氏投影：球内点无需改变；球外最短距离点与 g_t 同方向且范数为 c. 因此它在触发时保持整向量方向，而逐坐标截断一般改变方向.

取 $g = (3, 4)$、$c = 2$，得到 $g' = (1.2, 1.6)$. 若损失缩放为 $s = 8$，先裁剪 $(24, 32)$ 到范数 2 再除以 8，将得到 $(0.15, 0.2)$，范数只有 0.25. 正确顺序是先恢复未缩放梯度，再聚合完整更新的梯度并裁剪. 对于参数分片，全局范数需要统计不重复的逻辑参数；数据并行副本不能被重复计数.

裁剪限制输入优化器的梯度，不直接保证 AdamW 参数更新的范数不超过 $\eta_t c$. 矩估计、分母和权重衰减仍会改变更新尺度. 非有限梯度应先判定并处理，不能期待将无穷乘以零得到有效方向.

19.5　训练状态闭合性

19.5.1　充分状态及恢复等级

将下一次计算表示为状态转移

$$\mathcal{S}_{t+1} = F(\mathcal{S}_t, \mathcal{A}), \tag{19.13}$$

其中 $\mathcal{A}$ 是版本固定的数据、模型配置和运行环境. **检查点**（Checkpoint）应保存足以决定后续计算的 $\mathcal{S}_t$. 常见组成见表 19.2.

保存随机种子不同于保存随机数生成器的当前位置. 种子只能重建起点；训练中 Dropout、采样和数据增强已经消耗了许多随机数. 恢复时还应避免模型初始化或数据预取在载入随机状态后意外消费它们.

表 19.2: 训练检查点的状态组成

状态	保存原因
参数与缓冲区	决定前向计算，包括不属于可训练参数的持久缓冲.
优化器状态	一阶矩、二阶矩、步数、参数组、主精度副本及稳定项配置.
日程与计数	成功更新、尝试更新、消费词元和学习率日程位置.
精度状态	动态缩放因子、增长或回退计数等.
数据位置	数据版本、分片、排列、采样器、已提交游标及尚未提交的预取状态.
随机状态	各进程、各设备及数据工作进程使用的随机数生成器状态.
资产与环境	分词器、模板、掩码规则、模型结构、代码和依赖版本的标识.

语义续训保留目标、资产和必要状态，允许数值误差带来的轨迹差异；**数值接近的续训**进一步要求相同运算组织并给定误差容限；**逐位一致的续训**要求算子、归约顺序、并发及随机行为均满足确定性条件. 完整状态是精确恢复的必要条件，却不单独保证不同设备和软件版本间逐位相同. [1]

19.5.2 一致性边界及原子发布

在一次完整更新之后、下一批数据消费之前保存，是较容易说明的一致性边界. 若在梯度累积途中保存，还必须保存部分梯度、累计有效词元数、微批次位置及相关随机状态. 否则恢复后会丢失或重复部分贡献.

后台写入检查点时，应先获得不可变快照. 直接把持续更新的参数张量交给异步写线程，可能将不同更新时刻的张量混在同一文件中. 分布式情况下还需确认全部分片属于同一代状态.

发布过程可采用“写分片、验证清单、提交代号”的顺序. 分片写入唯一代号的暂存位置，记录大小与摘要；全部必需分片持久化后，提交包含资产版本和状态边界的清单. 读取者只接受已经提交且清单完整的代. 文件系统重命名和对象存储的发布机制具有不同语义，应由存储层提供明确保证，不能把“文件存在”等同于“检查点完整”.

[1] 浮点加法不满足严格结合律. 即使输入相同，改变并行归约树也可能改变末位结果；训练迭代可能逐步放大这种差异.

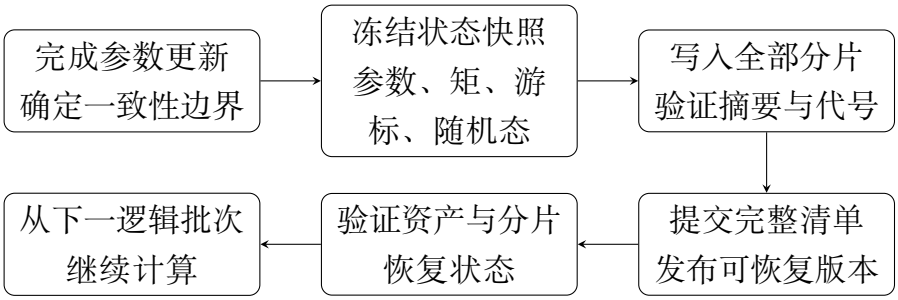


图 19.1: 检查点保存与恢复的数据流．发布发生在完整性确认之后；异步写入使用冻结快照．

19.6　训练算法及诊断

算法 19.1　具有明确跳步语义的训练更新

输入：固定版本数据与初始状态，成功更新预算 S，裁剪阈值 c．**输出：**最终训练状态与已提交检查点．本算法选择“溢出批次已消费但不更新”的策略，并分别记录三种计数．

1. 在一致性边界恢复状态；若为新训练，初始化参数、矩、随机生成器及数据游标，令 $t = a = n = 0$.
2. 当 $t < S$ 且尚有允许消费的数据时，构造完整更新所需的微批次，记录有效目标总数 N．若 $N = 0$，不执行除法或参数更新，记录原因并继续．
3. 清空旧梯度，以有效词元加权目标执行前向和缩放反向．恢复未缩放梯度，完成必要的梯度聚合．令 $a \leftarrow a + 1$、$n \leftarrow n + N$，提交本批数据消费位置．
4. 若梯度非有限，降低缩放因子，清空梯度，记录异常；保持参数、矩和成功更新计数不变．连续异常超过预设停止条件时保存诊断信息并停止．
5. 否则按式 (19.12) 裁剪，以 η_{t+1} 更新矩及 AdamW 参数，令 $t \leftarrow t + 1$，再更新缩放器的成功计数．
6. 按预定间隔记录损失、梯度范数、更新范数及学习率，执行固定验证；到达保存条件时，按图 19.1 提交完整检查点．

也可以选择在溢出后重放同一批次，但必须恢复相应随机状态、数据游标以及前向可能修改的缓冲区，并限制重试次数．改变这一策略会改变数据消费轨迹，不能在恢复时隐式切换．

诊断至少联合观察未裁剪梯度范数、裁剪比例、实际更新范数、学习率、缩放因

子和有效词元数. 对参数组 j，可记录相对更新量

$$r_{t,j} = \frac{\|\theta_{t,j} - \theta_{t-1,j}\|_2}{\|\theta_{t-1,j}\|_2 + \delta}, \qquad \delta > 0. \tag{19.14}$$

参数接近零时该比值敏感，应同时报告绝对量. 损失尖峰若伴随词元数锐减，先检查归一化和数据；若伴随非有限激活，定位首次异常算子；若梯度有限但更新异常，则检查矩状态、稳定项和学习率. 仅观察经过裁剪后的范数，可能掩盖持续发生的梯度异常.

验证指标须按有效目标词元聚合，而非简单平均批次均值. 用于选择检查点的验证集已经参与模型选择，不能再被当作独立的最终测试证据. 断点恢复的验证还应比较下一批标识、学习率、矩状态及后续若干次更新；只比较加载前后权重相等，无法证明后续训练一致.

第二十章　监督微调

能够延续文本的基座模型，不一定按照请求组织答案，也不一定遵循角色、格式与结束协议．后训练通过指令示范、偏好比较、环境反馈或可验证奖励调整这些条件下的生成分布．相应的数据不再只是文档集合，还包括指令与回答、偏好对、策略提示、采样轨迹和推理过程；它们共享数据治理原则，却具有不同的结构约束与质量标准．

本章先建立后训练数据的类型与生产接口，再研究指令示范如何成为精确定义的监督微调目标．第十八章《训练数据工程基础》给出版本、血缘、划分、去重和污染控制的共同原则，第二十三章《预训练原理》解释一般语言建模与训练预算；本章集中讨论回答区域、多轮对话、指令数据质量和能力保持．第二十一章《参数高效微调》将讨论减少可训练参数的方法，第二十四章《强化学习基础》建立策略优化基础，第二十五章《人类反馈学习》与第二十六章《可验证推理训练》分别展开偏好数据和可验证推理数据．更新哪些参数、生产哪类数据与采用哪一种监督信号，是三个相互关联但不能混为一谈的设计维度．

20.1　监督微调的学习对象

20.1.1　条件生成

监督微调（Supervised Fine-Tuning，SFT）以已有模型为起点，利用输入与目标输出的配对数据更新参数．语言模型中的**指令微调**（Instruction Tuning）是其中一类：输入说明任务要求，目标回答展示如何完成任务．分类、翻译等任务同样可以写成文本条件与文本回答，但条件字段的转换必须保留原任务含义．

设提示或历史为 x，示范回答为 $y=(y_1,\dots,y_n)$，策略模型为 π_θ．在起始条件和结束标记明确时，回答的条件概率为

$$\pi_\theta(y\mid x)=\prod_{t=1}^{n}\pi_\theta(y_t\mid x,y_{<t}). \tag{20.1}$$

这里 θ 包含本次可更新的参数；固定参数仍参与前向计算．最大化示范回答的条件似

然，等价于最小化

$$\ell(x, y; \theta) = -\sum_{t=1}^{n} \log \pi_\theta(y_t \mid x, y_{<t}). \tag{20.2}$$

训练中使用真实的先前回答词元作为条件，而生成时使用已经生成的前缀．这一差别使似然评估和自由生成评估各自不可替代．

InstructGPT 的研究将人工示范上的监督训练与后续偏好及奖励阶段区分开来[12]．SFT 本身不要求奖励模型，也不要求成对比较．给一个正确示范提高概率，并不等于直接学习“它为何优于某个错误回答”；若任务需要这种比较，必须另外定义偏好数据与目标．

20.1.2 条件目标及联合目标的区别

若将提示与回答整体作为语言建模目标，链式法则给出

$$-\log \pi_\theta(x, y) = -\log \pi_\theta(x) - \log \pi_\theta(y \mid x). \tag{20.3}$$

只监督回答保留第二项；全序列监督还拟合提示分布．后者不是数学错误，也不必然造成复述，但会把有限梯度预算分配给用户文本、系统描述和模板．对于长提示、短答案任务，这种额外目标可能占据多数词元．

更一般地，可以给提示目标与回答目标分别设置权重．选择回答专用损失，是为了直接拟合请求条件下的示范行为，而不是因为提示没有信息．目标选择应由任务确定，不应由数据列名称或训练库默认值决定．

20.1.3 符号及前提

符号	含义
x_i, y_i	第 i 个样本的提示条件与示范回答．
$c_{i,t}$	目标词元 t 实际可见的序列前缀．
$z_{i,t,v}$	第 i 条序列位置 t 对词表项 v 的预测分数．
T_i, V, M	序列长度、词表大小与样本数量．
$r_{i,j}$	序列位置 j 是否作为监督目标的标记．
$m_{i,t}$	预测位置 t 对后继目标的监督标记，$m_{i,t} = r_{i,t+1}$．
n_i, S	样本 i 的有效目标数与聚合范围内总有效目标数．
w_i	样本的非负权重；与回答长度的作用需分别定义．
θ_0, θ	初始模型参数与微调后参数．

条件监督的共同前提

本章采用因果语言模型和固定分词器．角色、消息顺序、目标轮次与停止标记由结构化数据及模板定义．监督目标对应的前缀不得包含该目标或其未来答案；填充不提供有效语义条件．所有损失比较固定目标区域与归一化方式，数值例为构

造算例.

20.2　回答掩码及移位的精确定义

20.2.1　监督位置映射

将结构化消息经过模板编码，得到序列 $u_0,\dots,u_{T-1}$. 位置 t 的输出预测 u_{t+1}. 先在目标词元所在位置建立 r_j，再定义

$$m_t = r_{t+1}, \qquad L = \frac{\sum_{t=0}^{T-2} m_t[-\log \pi_\theta(u_{t+1} \mid u_{\leqslant t})]}{\sum_{t=0}^{T-2} m_t}. \tag{20.4}$$

若标签数组与输入数组等长，提示和填充位置的标签为忽略值，回答位置保留对应词元编号，那么计算损失时应配对输出的前 $T-1$ 位与标签的后 $T-1$ 位. 不能先按输出位置建立回答掩码，再不加调整地复用到目标位置.

另一种接口显式准备输入与目标，其中输入相对于目标右移. 两种接口等价的条件是每个预测位置与后继目标恰好配对一次；“标签右移”这一含糊表述容易混淆数组操作与语言建模关系，应直接标注下标.

20.2.2　停止标记监督

回答正文之外，模型往往还需要学习轮次结束或序列结束标记. 两者可能不是同一词元：轮次结束允许对话继续，整个序列结束表示当前生成终止. 应根据制品模板决定哪些控制标记由模型预测，不能无条件把所有特殊词元屏蔽.

若始终删除结束标记的监督，模型就缺少在目标位置停止的直接示范. 若把人工截断的回答末尾强行补成结束标记，又会示范提前结束. 停止协议的有效性取决于语义完整性与实际生成接口共同一致.

例 18.1

考虑单轮样本，使用教学符号 U、A 表示用户与助手角色标记. 序列为

$$(u_0,\dots,u_7) = (\mathrm{BOS}, U, q, A, a, b, \mathrm{EOT}, \mathrm{PAD}). \tag{20.5}$$

q 是用户问题，回答为两个词元 a,b，EOT 为回答终止词元. 这里助手角色标记是预先提供的生成前缀，不作为本例目标，故

$$(r_0,\dots,r_7) = (0,0,0,0,1,1,1,0). \tag{20.6}$$

有效预测配对是位置 3 预测 a、位置 4 预测 b、位置 5 预测 EOT. 注意第一个回答词元的损失来自最后一个提示位置的输出.

输出位置 t	当前词元	后继目标	m_t	正确目标概率
0	BOS	U	0	不计
1	U	q	0	不计
2	q	A	0	不计
3	A	a	1	$\frac{1}{2}$
4	a	b	1	$\frac{1}{4}$
5	b	EOT	1	$\frac{1}{8}$
6	EOT	PAD	0	不计

于是

$$L = \frac{\log 2 + \log 4 + \log 8}{3} = \log 4, \qquad \mathrm{PPL} = 4. \tag{20.7}$$

若误把输出位置 4、5、6 作为回答损失范围，就会漏掉 a，并错误监督 PAD．即使张量形状完全正确，也已经优化另一个目标．

对有效输出位置，softmax 交叉熵的分数梯度为

$$\frac{\partial L}{\partial z_{t,v}} = \frac{m_t}{3}\left(\pi_{t,v} - \mathbf{1}[v = u_{t+1}]\right). \tag{20.8}$$

因此三个正确词元分数的导数分别为 $-\frac{1}{6}, -\frac{1}{4}, -\frac{7}{24}$．这说明损失对低概率正确目标施加较大的局部修正；整网参数梯度还需乘对应分数对参数的雅可比，不能仅凭这三个数判断参数更新范数．

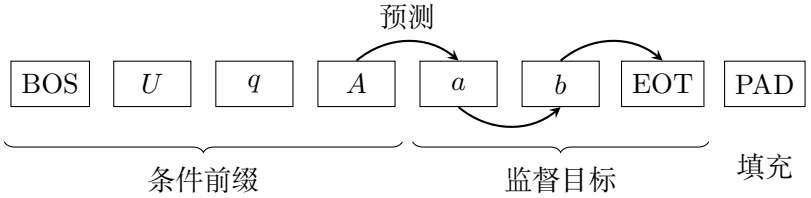


图 20.1: 目标位置与预测位置相差一位．回答目标从 a 开始，而预测它的输出位于助手前缀末端．

20.2.3　提示词元的梯度影响

回答位置的隐状态通过注意力读取提示的键和值．用单个注意力输出 $o_t = \sum_{j \leqslant t} a_{t,j} v_j$ 说明，在暂时固定权重 $a_{t,j}$ 的条件下，提示位置 j 的值向量具有梯度通路

$$\frac{\partial L}{\partial v_j} = \sum_{t \geqslant j} a_{t,j} \frac{\partial L}{\partial o_t}. \tag{20.9}$$

实际网络还存在经注意力权重、键和查询传播的梯度．故标签被忽略不等于提示嵌入或处理提示的参数被冻结．提示需要参与前向，也可能通过回答误差参与参数学习．

三个掩码的职责

填充掩码决定哪些输入位置有效；因果及样本掩码决定哪些位置之间可见；回答掩码决定哪些后继词元产生直接监督．回答掩码不能代替注意力隔离，也不能作为“哪些输入不需要计算”的依据．

20.3　样本权重及长度归一化

20.3.1　三种常见目标

设第 i 个样本有 $n_i > 0$ 个目标，目标损失和为 ℓ_i，平均损失为 $\bar{\ell}_i = \dfrac{\ell_i}{n_i}$．至少有三种不同目标：

$$L_{\text{token}} = \frac{\sum_i \ell_i}{\sum_i n_i}, \tag{20.10}$$

$$L_{\text{sample}} = \frac{1}{M} \sum_i \bar{\ell}_i, \tag{20.11}$$

$$L_{\text{sequence}} = \frac{1}{M} \sum_i \ell_i. \tag{20.12}$$

第一种词元等权，第二种样本内先平均再对样本等权，第三种平均序列负对数似然．固定数据集上，第一与第三仅相差整体常数；在长度变化的微批次中，逐批动态归一化则可能改变步间梯度尺度．第二种改变相对样本权重，对短回答中的每个词元赋予更大影响．

例如两条回答长度分别为 2 与 8，平均损失分别为 1 与 3，则 $L_{\text{token}} = 2.6$，$L_{\text{sample}} = 2$，$L_{\text{sequence}} = 13$．样本等权时，短回答中一个词元的权重为 $\dfrac{1}{4}$，长回答中一个词元为 $\dfrac{1}{16}$，前者是后者四倍．两种归一化都可以用于明确的目标，但不能在实验比较中静默切换．

20.3.2　带权目标及任务配比

给定非负样本权重 w_i，回答词元加权目标为

$$L_w = \frac{\sum_i w_i \ell_i}{\sum_i w_i n_i}, \tag{20.13}$$

而样本平均加权目标为 $\dfrac{\sum_i w_i \bar{\ell}_i}{\sum_i w_i}$．前者中长回答获得更大的样本总权重；后者中总权重由 w_i 决定．所谓“任务各占一半”必须说明按样本、回答词元还是梯度贡献计量．

长度本身不是质量．把所有长答案降权可能削弱完整论证；按词元等权也可能让重复冗长模板占据大量预算．合理做法是在数据阶段改进答案质量，再根据任务价值和覆盖选择统计单位，避免用权重修补错误答案．

20.3.3 梯度累积及评估归约

如果一次更新包含多个微批次，设第 k 批平均损失为 L_k，有效目标数为 S_k，词元等权更新应使用 $\dfrac{\sum_k S_k L_k}{\sum_k S_k}$. 动态填充、多轮掩码和截断使 S_k 不固定，简单除以累积批次数通常不足以保持该目标.

评估也应分别累积总负对数似然与目标数. 若报告样本等权指标，则保留各样本平均损失后再聚合. 对极大损失仅裁剪困惑度展示值时，必须保留未裁剪的负对数似然，并标明显示上限，不能把显示处理伪装成真实指标改善.

20.4 多轮对话及模板协议

20.4.1 结构化消息协议

会话模板（Chat Template）把消息角色、内容、调用关系和边界转换成模型识别的词元序列. 它不是可随意替换的装饰格式. 数据中的系统、用户、助手和工具角色，必须映射为与基座训练协议兼容的结构；普通文本中出现“assistant”一词，不应被解释为真实角色边界.

模板应提供可追溯的消息或生成跨度，使标签掩码来自结构而不是搜索回答字符串. 回答可能在问题中被引用，也可能多次出现；依赖第一个子串匹配会标错区域. [1]

编码整个会话后再根据可靠跨度映射到词元，或者使用能够返回生成区掩码的模板机制，都需要明确边界对齐规则. 跨文本跨度的单个词元如果无法按协议唯一归属，应修正模板或编码方法，不能依靠猜测赋予监督.

20.4.2 多轮监督范围

设一段对话为 (u_1, a_1, u_2, a_2). 全部助手监督对应

$$\ell_{\text{all}} = -\log \pi_\theta(a_1 \mid u_1) - \log \pi_\theta(a_2 \mid u_1, a_1, u_2), \tag{20.14}$$

最后一轮监督只保留第二项. 最后一轮训练仍读取前面的 a_1，但不直接把它当作本条样本的输出目标. 因此，历史中的错误答案虽然可以不受直接模仿，仍会改变第二轮的条件分布.

若把每个会话前缀都展开为一个样本，又在每个前缀中监督全部助手轮次，前面的回答会重复计权. 上述对话展开为 (u_1, a_1) 和完整四轮时，a_1 被监督两次，a_2 一次. 若希望每个助手轮次只计一次，可在前缀样本中仅监督最后一轮，或直接使用整段对话监督全部轮次，并避免重复展开.

[1] 分词通常不满足“先分别编码再拼接”等于“拼接后编码”. 边界空白和子词合并可能改变词元序列，因此提示编码长度不能无条件作为完整序列中的回答起点.

20.4.3　控制标记及训练—生成一致性

训练时可能包含完整助手回答及结束标记，生成时则应在助手内容开始前结束提示，并由模型继续输出．若生成提示多加一个结束标记，或者缺少助手起始前缀，条件已经改变．模板升级、额外特殊词元和词表扩展必须与权重绑定；新增词元的嵌入及输出参数需要适当训练，不能仅修改分词器文件．

对工具交互，助手产生调用，工具返回结果，两者属于不同来源．通常不能把工具原始返回文本当作助手应生成的答案；如果训练助手输出调用结构，则应监督调用参数与边界，并保留调用与结果关联．详细工具协议将在应用章节展开，本章只强调消息身份决定监督含义．

20.4.4　会话截断

长度超限时，删除最早消息可能丢失任务约束或问题指代；截断回答尾部可能留下不完整解答；从中间切断工具调用会破坏结构．截断不能仅凭“长度合法”判断成功，应确定保留的是哪个条件任务．

可按目标轮次组织窗口，优先保留系统约束、最近必要上下文与完整目标回答；对于无法在窗口内表达的任务，应隔离或转入长上下文训练．多轮切片仍以完整会话或问题家族进行训练—验证划分，避免不同前缀跨集合泄漏．

20.5　指令数据的来源及质量

20.5.1　人工示范、任务转换及合成

人工示范适合明确复杂行为标准，但需要一致的标注规范、复核流程和任务覆盖设计．一个答案可以语言流畅却漏掉限制条件，也可以形式正确却包含错误事实．标注应分别判断任务正确性、指令遵循、完整性、可支持性与表达质量，不能只给一个模糊总体印象．

已有任务数据可以转换为指令格式．例如分类标签映射为类别名称，需要固定标签语义；翻译句对需要保留语言方向；抽取任务需要说明输出边界及无答案规则．模板不能泄漏标签，训练中的固定表述也不能被当作指令多样性的充分证据．

合成指令数据（Synthetic Instruction Data）由模型或程序生成．Self-Instruct 展示了从种子任务生成指令、输入与输出，并过滤无效或相似样本的流程 [113]．这一方法提供扩大候选池的途径，却不保证每条生成答案正确．教师模型、提示、采样配置和过滤版本都应保留，以区分真实任务扩展与同一错误的重复传播．

20.5.2 数据过滤偏差

设候选回答正确概率为 p，过滤器对正确回答的接受概率为 r，对错误回答的误接受概率为 f. 由贝叶斯公式，保留集合的正确率为

$$p_{\text{keep}} = \frac{pr}{pr + (1-p)f}. \tag{20.15}$$

例如 $p = 0.8, r = 0.9, f = 0.2$，候选通过率为 0.76，通过后的正确率为 $\frac{0.72}{0.76} \approx 94.74\%$. 过滤提高了纯度，却同时删除部分正确样本；若错误集中在某个领域，其保留错误仍可能系统相关. 该算例假设判据与真值已定义，不能以模型自评分直接声称获得这些 r, f.

程序可执行、答案可精确核对的任务能够使用验证器，但验证覆盖必须说明. 代码通过有限测试不能证明所有输入正确；数学最终答案正确也不能证明每一步说明正确；引用存在不能证明引用支持结论. 教师生成与同一教师自评的误差可能相关，因此复核应尽可能引入独立证据或不同的检验机制.

20.5.3 多样性及难度

指令多样性需要覆盖任务目标、输入分布、约束组合、语言、长度和交互结构. 把同一道题改写十次，会增加文字形式而未必增加能力覆盖. 按问题家族去重后，仍可保留有价值的表达变体，但其训练权重应受到控制.

难度也不能只由回答长度或教师置信度决定. 短问题可能需要多步推理，长答案可能只是模板重复. 应结合可验证的前提数量、步骤依赖、约束组合和错误类型定义难度层，再观察模型在哪一层真正受益.

LIMA 研究在其特定基座与人工整理数据条件下展示了少量高质量指令示范的作用 [114]. 它不能推导出任意模型只需固定数量样本，也不能证明额外领域知识无须训练. 数据数量、覆盖和示范质量的收益必须结合基座已具备的能力判断.

20.5.4 多轮一致性及可信监督

多轮数据应检查实体、时间、单位和用户修正是否前后一致. 若用户已经纠正条件，助手仍沿旧条件作答，该样本会训练错误的对话更新行为. 包含澄清、修正与承认不确定性的示范，有助于表达正确交互模式；但不能用大量统一拒绝模板替代任务判断，否则可能导致过度拒答.

对于需要外部证据的任务，应保留证据来源与答案支持关系. 对于信息不足的问题，示范应体现缺失信息的处理方式，而不是编造具体事实. 错误样本经反复训练会被更有力地模仿，降低学习率不能把错误目标变成正确目标.

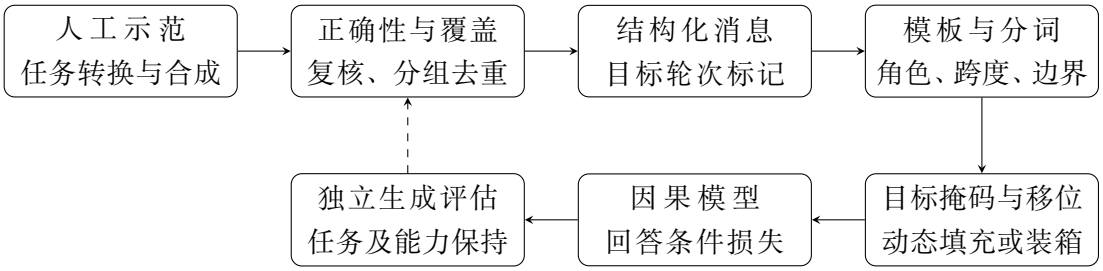


图 20.2: 指令数据到监督目标的形成过程. 内容正确性、模板编码与损失边界分别承担不同职责；评估反馈用于下一轮数据与方案选择.

20.6　训练批次及参数更新的接口

20.6.1　动态填充及独立会话装箱

动态填充（Dynamic Padding）按批内最长序列分配长度，减少相对于全局最大长度的空位. 若 PAD 与 EOS 共用编号，不能仅按词元编号屏蔽所有该值的位置，否则真实结束目标也被删除；应根据真实长度或明确位置掩码判断填充身份.

多个会话装箱时，独立会话默认不应相互读取. 令 $d(t)$ 为会话编号，可见关系为 $j \leqslant t$ 且 $d(j) = d(t)$，回答目标还须属于相同会话. 高装箱率不证明标签正确，块对角注意力也不自动修正跨会话目标配对. 对于标准批次，输入与标签形状为 $[B, T]$，预测分数形状为 $[B, T, V]$；掩码的语义比这些维度更重要.

20.6.2　全参数及受限参数更新

SFT 可以更新全部参数，也可以只更新部分模块或附加参数. **参数高效微调**（Parameter-Efficient Fine-Tuning，PEFT）改变可训练子空间，通常保留相同回答损失；其秩、模块选择与制品合并在第二十一章《参数高效微调》展开. 不能把“使用适配器”当成一种新的监督目标，也不能仅凭可训练参数少推断不会遗忘.

训练过程应记录有效回答词元量、样本与任务配比、优化器更新数和实际消费的数据版本. 训练损失窗口均值反映历史参数，不等于最终模型质量. 完整恢复还需要优化器、调度器、随机状态和数据位置；只保存适配器或权重可用于加载模型，但不足以描述原训练轨迹.

算法 20.1　由会话构造条件监督批次

输入为结构化会话、目标轮次规则、固定模板与分词器；输出为输入、可见性掩码、目标标签及有效目标计数. 状态包含会话身份和截断记录.

1. 校验角色顺序、必要字段及调用关联，确定允许监督的助手轮次与终止标记.

2. 按完整会话语义选择窗口；无法保留必要条件或完整回答时隔离样本，不把

截断答案伪装成完整示范.

3. 用模板编码并保留消息和目标跨度，映射到词元级 r_j；若边界不能可靠对齐则停止该样本的构造.

4. 动态填充或按独立会话装箱，生成真实位置与会话可见性掩码；填充位置的目标标记为零.

5. 将位置 t 的输出与位置 $t+1$ 的目标配对，令 $m_t = r_{t+1}$，同时排除跨会话配对.

6. 汇总有效目标数；零目标样本不进入更新. 输出计数、来源身份与掩码，供训练归一化使用.

不变量是每个被监督词元只由合法前缀预测，提示与填充不被误计为回答. 长度为 T 的跨度处理通常可线性完成；显式构造稠密可见性矩阵需要 $O(T^2)$ 空间，实际系统可采用等价的边界表示.

20.7 评估及能力保持

20.7.1 条件似然及自由生成

条件验证损失使用示范前缀，衡量模型对参考回答的概率分配. 自由生成评估只提供请求与历史，衡量模型在自身生成轨迹上的任务结果. 一个问题可能有多个同样正确的答案，单条参考回答似然不能覆盖全部正确表达；相反，模板化答案高概率也不保证满足任务.

评估应至少区分内容正确、格式有效、约束遵循和停止行为. 对代码、抽取、分类等任务可采用明确规则；对开放回答可采用预先定义的评分标准和盲评. 模型评审器可能存在长度、风格与顺序偏差，应校准并保留人工或客观证据. 采样温度、输出上限、模板和工具条件必须固定，否则比较的是多个变量的混合影响.

20.7.2 泄漏及重复选择

会话前缀、题目改写、同一来源答案应按问题家族分组. 训练与验证文本哈希不同，只能排除精确重复；不能排除同一任务实例的轻微重写. 合成数据的种子、教师提示和筛选过程也应避开封存测试题，不把测试失败题直接改写成训练样本后仍声称测试独立.

验证集用于选择训练步数与配方，测试集用于方案冻结后的估计. 比较新旧模型时应在同一批题目上进行配对分析，并按任务、语言、长度和风险切片. 总体均值可能掩盖某个关键领域退化；置信区间应按独立问题或会话聚合，而不是把高度相关的

词元当作独立试验.

20.7.3　遗忘的局部机制

灾难性遗忘（Catastrophic Forgetting）指学习新分布时原有任务表现显著退化. 设旧任务损失为 $L_0(\theta)$，新任务梯度为 g_1，一步更新 $\theta' = \theta - \eta g_1$. 泰勒展开给出

$$L_0(\theta') - L_0(\theta) \approx -\eta \nabla L_0(\theta)^\top g_1 + \frac{\eta^2}{2} g_1^\top H_0 g_1, \tag{20.16}$$

H_0 为旧损失的局部 Hessian. 若两任务梯度内积为负，一阶项就可能使旧损失上升；即使接近旧任务驻点，二阶曲率也会影响保持程度. 这个局部近似解释更新冲突，不证明所有退化均由单一步骤造成.

例如 $\nabla L_0 = (1, 0)$，$g_1 = (-1, 1)$，$\eta = 0.1$，忽略二阶项，旧损失增加约 0.1. 新目标下降并不排除旧目标上升. 模板变更、解码配置和评估失配也可能表现为能力下降，诊断时需要先区分参数变化与接口变化.

20.7.4　保持策略及其限制

混入代表旧能力的**回放数据**（Replay Data）可采用

$$L_{\text{mix}} = (1 - \lambda) L_{\text{SFT}} + \lambda L_{\text{replay}}, \qquad 0 \leqslant \lambda \leqslant 1. \tag{20.17}$$

它显式重新分配训练资源，但回放数据覆盖不足时仍可能遗忘. 较小更新量、较短训练、正则化或受限参数更新可以减少变化范围，却没有跨任务的普遍保持保证. 参数距离小也不意味着所有输出变化小，因为输入分布和网络敏感度不同.

应在微调前冻结基座基线，分别观察新任务收益和旧任务保持，并选择符合需求的折中. 若领域知识主要缺失于基座，少量格式示范可能不足；若主要问题是输出协议，增加大量同质事实文本也未必有效. 区分知识补充、任务迁移和行为格式，有助于选择继续预训练、SFT 或其他后训练阶段.

SFT 的证据边界

回答损失下降证明模型更贴近被选中的示范目标. 能否遵循新指令、处理自己的生成错误、保持旧能力以及可靠停止，必须分别通过生成与回归评估判断. 示范质量、目标掩码与评估分布共同决定结论范围.

算法 20.2　监督微调与模型选择

输入为冻结基座、指令训练集、验证集、任务指标与预算；输出为所选模型及绑定的数据和模板版本.

1. 在统一模板和生成条件下记录基座的任务与保持基线，冻结归一化规则和有效回答词元预算.

2. 按指定配比构造合法批次，累计有效目标并形成相应的词元或样本加权梯度；执行参数更新并推进实际训练计数.

3. 在预定检查点同时计算条件验证损失与独立自由生成指标，记录格式、长度、停止和任务切片.

4. 若出现持续退化，判断来源是错误示范、模板边界、目标权重还是更新幅度，再据验证集选择下一轮方案.

5. 达到预算或预定停止条件后，冻结候选并执行封存测试；保存完整权重或适配器、基座身份、分词器、模板、训练状态与评测证据.

全过程保持训练与测试问题家族隔离，禁止以训练损失最小替代多维质量选择.

20.8　训练契约及后续方法

回答掩码、动态填充、移位交叉熵、有效回答词元聚合与训练循环共同实现监督目标. 小型数据可用于逐位置检查监督关系；少量验证样本只能检验评估链路，不能支持通用指令能力结论. 标准训练库承载同一目标时，仍需绑定模板、目标轮次、截断和保存语义，不能仅凭训练器名称推定其默认配置与本章一致.

偏好比较与组相对优化由后续章节展开：本章不把成对偏好公式混入条件示范目标，也不把在线奖励循环当成 SFT 的必要步骤. 参数高效方法则与三类目标都可结合，其技术机制由第二十一章《参数高效微调》展开.

第二十一章　参数高效微调

预训练模型已经形成了可复用的表示与生成能力，下游任务未必需要重新调整全部权重. 但“只训练少量参数”并不意味着模型只执行少量计算，也不意味着任何任务都能用同样的小型增量解决. **参数高效微调**（Parameter-Efficient Fine-Tuning，PEFT）研究的是：在冻结大部分基座参数的条件下，如何选择可训练变量及其进入网络的位置，使有限的更新预算产生有效的函数变化.

PEFT 与训练目标是两个不同维度. 监督微调、偏好优化和强化学习规定如何评价输出，PEFT 规定损失能够更新哪些变量. 相同损失可以用于全参数微调或低秩适配；相同适配结构也可以服务于不同目标. 本章从低秩更新的代数结构出发，推导前向与反向传播，再讨论量化、提示类方法与适配器的部署生命周期.

21.1　受约束的模型适配

21.1.1　参数预算及函数空间

令冻结基座为 θ_0，任务参数为 ϕ，适配模型为 $f(x;\theta_0,\phi)$. 优化问题写为

$$\min_{\phi}\ \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\ell(f(x;\theta_0,\phi),y)\right]. \tag{21.1}$$

与直接优化 θ 相比，可训练变量的数量、进入计算图的位置以及参数化方式均受到限制. 限制有时降低小数据条件下的过拟合，也可能导致表达不足；它不是普遍的质量保证.

全参数微调（Full Fine-Tuning，FFT）允许所有指定基座权重变化. 选择性微调只开放其中的偏置、归一化参数或词嵌入行. 附加模块方法增加可训练的旁路或连续提示；重参数化方法则把权重增量分解为更小的变量. 不同方法可能具有相同的可训练参数量，却允许不同的函数变化.

21.1.2　冻结参数的梯度传播

考虑一个冻结线性层 $y=W_0x$. 尽管无需形成用于更新的 $\nabla_{W_0}\mathcal{L}$，链式法则仍要求

$$\nabla_x\mathcal{L}=W_0^{\mathsf{T}}\nabla_y\mathcal{L}. \tag{21.2}$$

表 21.1: 本章主要符号. 线性层采用列向量约定.

符号	含义与形状
W_0	冻结权重，$d_o \times d_i$
x, y, g	输入 d_i、输出 d_o、上游梯度 d_o 维列向量
A, B	低秩因子，分别为 $r \times d_i$、$d_o \times r$
r, α, s	秩上限、缩放超参数、实际尺度 $s = \dfrac{\alpha}{r}$
X, Y, G	n 个样本按列堆叠的输入、输出、上游梯度
p, d, L	连续提示长度、隐藏维度、网络层数
$W_q, \widehat{W}$	量化存储对象与其恢复的计算权重

如果 x 依赖较早层的适配器，缺少此梯度便无法训练那些适配器. 冻结参数与切断计算图是完全不同的操作. 只有当某段计算的输入也不依赖任何需要训练的变量时，才可能把它作为不需反传的固定特征提取阶段.

训练内存可以概念性地分为基座权重、可训练权重及其梯度与优化器状态、激活、临时工作区和运行时开销. PEFT 主要压缩其中与可训练变量有关的部分；长序列激活和冻结矩阵乘法依然存在. 因此，可训练参数比例不能直接用来预测峰值显存或每步时延.

21.2 LoRA 的低秩更新

21.2.1 前向计算及参数量

低秩适配（Low-Rank Adaptation，LoRA）将权重增量表示为两个窄矩阵的乘积 [115]：

$$y = W_0 x + sBAx, \qquad s = \frac{\alpha}{r}, \quad A \in \mathbb{R}^{r \times d_i}, \quad B \in \mathbb{R}^{d_o \times r}. \tag{21.3}$$

原有偏置若保持冻结，可单独加到右侧而不改变以下推导. 先计算 $z = Ax \in \mathbb{R}^r$，再计算 $sBz \in \mathbb{R}^{d_o}$，使增量与基座输出具有相同形状. 把 B 的列记为 b_k，A 的行记为 a_k^{T}，则

$$\Delta W = sBA = s \sum_{k=1}^{r} b_k a_k^{\mathsf{T}}, \qquad \Delta y = s \sum_{k=1}^{r} b_k (a_k^{\mathsf{T}} x). \tag{21.4}$$

每个通道先把输入投影为一个标量，再沿某个输出方向生成贡献. 这些通道是分解中的代数方向，不天然对应独立概念或专家.

由于乘积的列空间包含于 B 的列空间，$\operatorname{rank}(BA) \leqslant r$. 这一约束针对增量，不针对 W_0 或 $W_0 + sBA$. 可训练参数量由 $d_o d_i$ 变为 $r(d_i + d_o)$，比例为

$$\gamma = \frac{r(d_i + d_o)}{d_i d_o}. \tag{21.5}$$

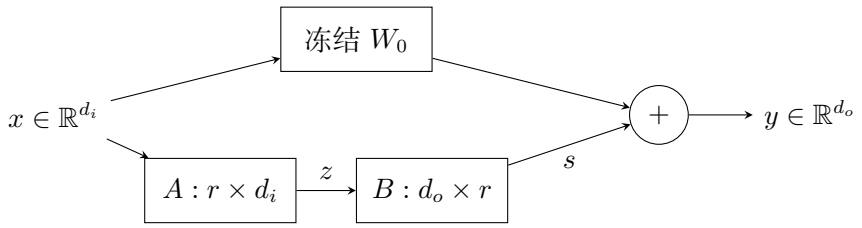


图 21.1: LoRA 只训练低秩旁路，基座仍参与前向计算和输入梯度传播.

仅当 $r < \dfrac{d_i d_o}{d_i + d_o}$ 时，新增因子数少于原矩阵元素数. 对于方阵 $d_i = d_o = d$，比例为 $\dfrac{2r}{d}$. 例如 $d = 4096, r = 8$，基座矩阵有 16777216 个元素，适配器有 65536 个参数，占 0.390625%. 这是该矩阵的比例，不能直接当成整模型比例.

21.2.2　低秩假设的适用边界

如果已知理想增量 $\Delta W_\star$，其奇异值按降序为 $\sigma_1, \sigma_2, \ldots$，在无权 Frobenius 范数下最优秩 r 近似的平方误差为 $\sum_{k>r} \sigma_k^2$. 这个线性代数结论有助于理解容量，但真实微调优化的是任务损失，并非已知矩阵的无权逼近.

即使只有一层，输出误差也取决于输入协方差. 令 $E = \Delta W - \Delta W_\star$，输入二阶矩为 $C = \mathbb{E}[xx^\mathsf{T}]$，则

$$\mathbb{E}\|Ex\|_2^2 = \operatorname{tr}(ECE^\mathsf{T}). \tag{21.6}$$

当 $C \neq I$ 时，不同输入方向的误差权重不同；深层非线性网络中的任务关系更复杂. 由人工低秩目标产生的合成回归可以验证实现能够学习该结构，却不能证明所有任务更新都低秩.

因子本身也不唯一：对任意可逆 $R \in \mathbb{R}^{r \times r}$，有 $(BR)(R^{-1}A) = BA$. 因此不同训练得到的通道不能按编号直接比较，其范数还受分解尺度影响. 增大 r 扩大可表示集合，但不保证固定步数、固定学习率下更容易优化.

21.3　反向传播、初始化及缩放

LoRA 梯度推导

基座 W_0 固定，s 为非零常数，损失对输出可微；此处先不使用随机丢弃. 批量样本按列组成 $X \in \mathbb{R}^{d_i \times n}$. 若损失包含求均值，其归一化已经体现在上游梯度 $G = \dfrac{\partial \mathcal{L}}{\partial Y}$ 中.

21.3.1 由微分得到三个梯度

批量前向为 $Y = W_0X + sBAX$. 保持 X 不变时，输出微分为

$$\mathrm{d}Y = s(\mathrm{d}B)AX + sB(\mathrm{d}A)X. \tag{21.7}$$

使用 $\mathrm{d}\mathcal{L} = \operatorname{tr}(G^{\mathsf{T}}\,\mathrm{d}Y)$ 及迹的循环性质，把参数微分移到右侧，可得

$$\nabla_B\mathcal{L} = sG(AX)^{\mathsf{T}} \in \mathbb{R}^{d_o\times r}, \tag{21.8}$$

$$\nabla_A\mathcal{L} = sB^{\mathsf{T}}GX^{\mathsf{T}} \in \mathbb{R}^{r\times d_i}. \tag{21.9}$$

若 X 也依赖可训练参数，则还必须计算

$$\nabla_X\mathcal{L} = (W_0 + sBA)^{\mathsf{T}}G = W_0^{\mathsf{T}}G + sA^{\mathsf{T}}B^{\mathsf{T}}G. \tag{21.10}$$

第一项来自冻结基座，不能删除. 这也解释了为何冻结主路的参数梯度后，训练计算量不会按 γ 同比例下降.

21.3.2 低秩因子初始化

常见初始化采用随机 A 和零矩阵 B，使 $\Delta W = 0$，初始函数保持为基座函数. 由式 (21.9)，第一步 $\nabla_A\mathcal{L} = 0$，而 $\nabla_B\mathcal{L} = sG(AX)^{\mathsf{T}}$ 通常非零. B 更新以后，A 才开始获得数据梯度. 若 $A = B = 0$，两个梯度同时为零，普通一阶梯度更新无法启动该分支.

“通常非零”仍需条件：若输入投影 $AX = 0$ 或上游梯度与投影相消，B 的批量梯度也可能为零. 初始化为零增量不代表每次训练的第一次更新必然有效. 反过来，也可以把 A 置零而令 B 非零，此时首先更新的是 A；两种参数化的优化路径不必相同. [1]

对于单样本、$B_0 = 0$ 的一次梯度下降，记学习率为 η，则

$$B_1 = -\eta s\, gx^{\mathsf{T}}A_0^{\mathsf{T}}, \qquad \Delta W_1 = -\eta s^2 gx^{\mathsf{T}}A_0^{\mathsf{T}}A_0. \tag{21.11}$$

更新依赖 $A_0^{\mathsf{T}}A_0$，而非简单地把完整矩阵梯度乘以常数. 若 A_0 的元素独立、零均值、方差为 σ_A^2，则 $\mathbb{E}[A_0^{\mathsf{T}}A_0] = r\sigma_A^2 I$；在固定初始基座梯度的条件下，第一步期望增量为 $-\eta s^2 r\sigma_A^2 gx^{\mathsf{T}}$. 该式直接说明秩、初始化尺度、学习率和 s 共同影响更新幅度.

21.3.3 尺度及旁路正则化

原始 LoRA 使用 $s = \dfrac{\alpha}{r}$[115]. 这是一种参数化约定，不是所有秩下最优更新相同的定理. 比较秩时，应明确是固定 α、固定 s，还是采用其他缩放规则；否则同时改变了容量与初始更新尺度.

[1] 这里讨论损失的数据梯度. 某些优化器的独立权重衰减可能在数据梯度为零时改变非零参数；应将这种变化与反向传播产生的学习信号区分.

若采用**随机失活**（Dropout），可令其只作用于低秩旁路输入：$y = W_0x + sBA\widetilde{x}$. 保留概率为 q，使用倒置失活 $\widetilde{x} = \dfrac{m \odot x}{q}$，则条件于 x 有 $\mathbb{E}[\widetilde{x}] = x$. 因此线性旁路输出的期望与无失活时相同，但非线性网络的损失期望通常不等于对期望输出求损失. 推理时关闭失活，才能使用后文的固定权重合并等价式.

21.3.4　LoRA 单步更新

例 19.1

取

$$W_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad A_0 = \begin{bmatrix} 1 & -1 \end{bmatrix}, \quad B_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad s = 1. \tag{21.12}$$

目标为 $t = (0, 1)^{\mathsf{T}}$，损失 $\mathcal{L} = \frac{1}{2}\|y - t\|_2^2$. 初始投影 $A_0x = -1$，输出 $y_0 = (1, 2)^{\mathsf{T}}$，损失为 1，上游梯度 $g = (1, 1)^{\mathsf{T}}$. 于是

$$\nabla_B \mathcal{L} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \qquad \nabla_A \mathcal{L} = \begin{bmatrix} 0 & 0 \end{bmatrix}, \qquad \nabla_x \mathcal{L} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{21.13}$$

取 $\eta = 0.1$，同时使用旧参数计算梯度后更新，得到 $B_1 = (0.1, 0.1)^{\mathsf{T}}$、$A_1 = A_0$. 新输出为 $(0.9, 1.9)^{\mathsf{T}}$，损失为 0.81. 更新后的矩阵为

$$W_0 + B_1A_1 = \begin{bmatrix} 1.1 & -0.1 \\ 0.1 & 0.9 \end{bmatrix}, \tag{21.14}$$

乘以 x 得到同一输出. 下一次反传时，$\nabla_A \mathcal{L} = (0.18, 0.36)$ 已不为零. 这个小矩阵例子用于展示梯度启动和等价路径；其四个可训练参数与四个基座元素数量相同，并不体现大模型中的显著压缩.

21.4　目标模块及更新预算

Transformer 中的查询、键和值投影改变注意力匹配和被聚合的信息，输出投影改变聚合结果进入残差流的方式，前馈层投影改变位置上的非线性特征变换. 只适配其中一类模块与适配全部线性映射，允许的函数变化不同. 第五章《注意力机制》给出了这些投影的形状；具有分组查询的结构中，键值投影往往并非 $d \times d$ 方阵，不能统一套用方阵参数公式.

设目标集合为 $\mathcal{M}$，每个模块有秩 r_j、输入输出维度 $d_{i,j}, d_{o,j}$，则总可训练因子数为

$$N_{\text{LoRA}} = \sum_{j \in \mathcal{M}} r_j(d_{i,j} + d_{o,j}). \tag{21.15}$$

若同时训练偏置、词嵌入或输出头，必须另行加上其参数量，并将其写入制品．模块名属于具体架构的实现，匹配规则必须依据实际模块树解析，不能仅凭名称子串推定已正确覆盖．合并查询键值的线性层还需要说明是适配整个融合矩阵还是其中指定切片．

固定总预算时，秩分配也是一个优化选择．**自适应低秩适配**（Adaptive Low-Rank Adaptation，AdaLoRA）通过重要性相关的预算分配调整不同增量方向的保留 [116]．**权重分解低秩适配**（Weight-Decomposed Low-Rank Adaptation，DoRA）将权重幅值与方向变化分别参数化，方向部分使用低秩更新 [117]．两者改变的是参数化与预算使用方式，不能只替换方法名称而沿用所有 LoRA 缩放、初始化和保存假设．

一个合理比较应固定基座、数据划分和有效训练词元预算，再分别观察目标模块、秩、缩放和学习率．参数量相同但更新位置不同的两种配置，应同时比较任务能力、原有能力保留和部署代价．窄域训练损失下降不代表全局能力得到保留．

21.5　合并、组合及数值语义

21.5.1　单适配器合并

当推理旁路是固定线性映射且失活关闭时，可定义

$$W_{\text{merged}} = W_0 + sBA. \tag{21.16}$$

在实数算术中，双分支和单矩阵形式完全相等．浮点中，先形成 BA 再与 W_0 相加，与逐层矩阵乘法的舍入次序不同，因此应按合理容差比较．若旁路含输入相关门控或非线性，便不能直接采用式 (21.16)．

合并消除了显式低秩旁路的运行开销，却使该模型成为某一任务配置的实体副本．多任务动态切换时，保留共享基座与独立适配器可能更有价值．应保存未合并原件，而不是依赖对合并权重反复相减来恢复；有限精度下的加减不保证无损往返．

21.5.2　适配器组合

若多个适配器均相对于同一基座、同一目标模块训练，固定线性组合可写为

$$\Delta W = \sum_{j=1}^{k} c_j s_j B_j A_j. \tag{21.17}$$

它可以通过拼接精确表示：$B_{\text{cat}} = [c_1 s_1 B_1 \ \cdots \ c_k s_k B_k]$，$A_{\text{cat}} = [A_1^{\mathsf{T}} \ \cdots \ A_k^{\mathsf{T}}]^{\mathsf{T}}$，总秩上界为 $\sum_j r_j$．这只是代数等价，不证明任务能力可以无冲突地相加．

直接计算 $(B_1 + B_2)(A_1 + A_2)$ 会产生 $B_1A_2 + B_2A_1$ 交叉项，通常不等于增量相加．由于因子分解不唯一，逐元素平均因子还依赖任意选取的内部坐标．若为了维持小秩而重新压缩合成增量，则引入新的近似误差，需要作为独立制品评估．

如果第二个适配器是在“基座加第一个适配器”上继续训练，其依赖关系也不同于两个独立任务增量．请求相关的动态组合系数更不能一次性固化为唯一矩阵．注册与部署时必须区分并行组合、串行训练依赖和单任务切换．

21.6　QLoRA 的量化及梯度边界

21.6.1　量化基座的梯度路径

量化低秩适配（Quantized Low-Rank Adaptation，QLoRA）结合低比特基座存储与可训练低秩参数 [118]．令 $Q(W_0)$ 为量化对象 W_q，包含码值和量化元数据；令 $D(W_q)=\widehat{W}$ 表示恢复到计算精度的权重，则训练的线性映射为

$$Y=\widehat{W}X+sBAX. \tag{21.18}$$

基座码值与其尺度保持冻结，梯度更新 A,B，但输入梯度仍为

$$\nabla_X\mathcal{L}=\widehat{W}^{\mathsf{T}}G+sA^{\mathsf{T}}B^{\mathsf{T}}G. \tag{21.19}$$

这里无需求量化离散码值的梯度；需要的是在固定恢复权重下对输入求导．因此“量化不可微”并不阻止该输入梯度的计算．是否临时物化全部恢复权重，还是按块在内核内恢复，属于运行时实现，不能从此公式推定必然分配一个完整高精度副本．

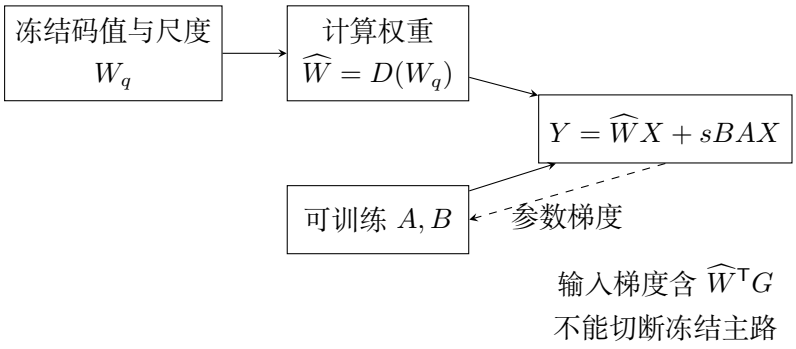


图 21.2: QLoRA 的存储、计算和更新是三个不同边界．虚线仅表示适配器参数梯度．

21.6.2　NF4、双重量化及内存账目

QLoRA 原始方案使用**四比特正态浮点**（4-bit NormalFloat，NF4）表示权重，量化码本面向近似正态分布的权重设计；它不是普通四比特整数的另一名称．**双重量化**（Double Quantization）进一步压缩块级量化常数，**分页优化器**（Paged Optimizer）则管理优化器状态的内存峰值 [118]．这些措施分别作用于不同内存项，并不意味着激活、梯度和键值缓存都成为四比特．

考虑自行给定的简化账目：P 个基座元素每个用四比特码值，每 g 个元素另存一个 32 位尺度，忽略对齐与其他元数据，则基座存储为 $\frac{P}{2}+\frac{4P}{g}$ 字节．$g=64$ 时为每元素 0.5625 字节，而不是 0.5 字节．双重量化改变第二项的表示，但不会消除元数据．此算例用于理解成本来源，不对应某个后端的实测布局．

训练还要加入适配器状态．例如给定 N_a 个参数使用两字节权重、两字节梯度、两个四字节优化器矩及四字节主副本，则该部分为 $16N_a$ 字节；并非所有运行时都采用这一布局．激活、工作区与分页迁移成本仍需另计，不能由权重位宽推出训练峰值或加速比例．

21.6.3 量化误差及部署转换

令 $\widehat{W}=W_0+E_q$，则模型实际学习的是 W_0+E_q+sBA．适配器可能同时响应任务需求与量化扰动，因此将同一 A,B 换到未量化 W_0 上不一定保持原行为．由于 E_q 可能包含许多方向，低秩增量也不保证完全补偿它．

纯推理量化是在训练结束后改变部署表示；QLoRA 则在训练前向与反向路径中使用固定量化基座并训练适配器．两者既可能组合，也不可混称．若部署时重新量化合并权重，得到 $D(Q'(\widehat{W}+sBA))$，这与训练时的 $\widehat{W}+sBA$ 又有差别．合并到原始高精度基座得到 W_0+sBA 则改变了另一个对象．发布记录必须说明实际选择哪条路径，而非笼统标注“已合并”．[2]

21.7 其他参数高效方法

21.7.1 瓶颈适配器及激活缩放

瓶颈适配器（Bottleneck Adapter）在网络中插入小型非线性模块 [119]．一种典型残差形式为

$$h'=h+U\,\sigma(Dh+b_d)+b_u,\quad D\in\mathbb{R}^{b\times d},\quad U\in\mathbb{R}^{d\times b},\quad b\ll d. \tag{21.20}$$

每个模块含 $2db+b+d$ 个参数．下降投影压缩到瓶颈，非线性转换后再映射回残差空间．插入在注意力之后、前馈层之后或并行旁路，会形成不同结构；不能把某个插入位置视为所有 Adapter 的定义．因为包含非线性，通常不能合并为原层的单一固定矩阵．

内激活抑制与放大适配（Infused Adapter by Inhibiting and Amplifying Inner Activations，IA^3）使用可学习向量缩放键、值和前馈中间激活 [120]．若一头注意力的键值按行排列，可写为 $K'=K\,\mathrm{diag}(\ell_k)$、$V'=V\,\mathrm{diag}(\ell_v)$；前馈部分可写为

$$h'=W_{\text{down}}\,[\ell_f\odot\sigma(W_{\text{up}}h)]. \tag{21.21}$$

[2] 量化格式、分组粒度、计算精度和后端均属于制品兼容性．某个后端能够读取四比特文件，不等于它实现了训练时相同的量化与恢复规则．

向量通常从全一初始化，初始映射不变．它不能任意混合通道，只能在所选坐标上调整幅度．固定缩放可以吸收到相邻线性映射的行或列中，但必须遵守实际放置位置；把缩放移动到非线性或归一化另一侧一般不等价．

21.7.2　输入软提示

提示微调（Prompt Tuning）学习一组输入连续向量，冻结语言模型权重 [121]．令词嵌入序列为 $E(x) \in \mathbb{R}^{T\times d}$，**软提示**（Soft Prompt）为 $P \in \mathbb{R}^{p\times d}$，输入为 $[P; E(x)]$，新增参数为 pd．这些向量不必等于词表中某个词的嵌入，也不必能够翻译成自然语言．

损失仍由真实目标位置计算，梯度通过整个冻结网络传回 P．相较于自然语言提示工程，它需要梯度训练；相较于 LoRA，它不直接修改线性映射．它增加输入位置和相应计算预算，具体对最大文本长度与位置编码的影响由模型协议决定．小型制品不意味着零推理开销．

21.7.3　逐层前缀及 P-Tuning

前缀微调（Prefix Tuning）在注意力层提供可学习的前缀状态 [122]．对单头，令 $P_K \in \mathbb{R}^{p\times d_k}$、$P_V \in \mathbb{R}^{p\times d_v}$，真实查询为 Q，则

$$O = \operatorname{softmax}\!\left(\frac{Q[P_K; K]^{\mathsf{T}}}{\sqrt{d_k}} + M\right)[P_V; V]. \tag{21.22}$$

M 允许真实词元读取前缀，并对真实词元间关系实施任务所需掩码．前缀不必通过输入词嵌入在每层间接产生，因而与仅学习输入 P 的约束不同．若每层键与值各自的总维度为 d_K, d_V，层数为 L，直接保存前缀键值需 $Lp(d_K + d_V)$ 个参数．当键和值各有 d 维时，才化为 $2Lpd$；分组键值结构应按实际维度计算．

前缀还增加注意力可见的键值位置．真实长度为 T 时，单头分数矩阵由 $T \times T$ 变为 $T \times (T + p)$．前缀可以预先计算和缓存，并不意味着其存储和读取免费．训练时使用提示编码器重参数化前缀、部署时保存生成的静态状态，是另一项需要记录的转换．

连续提示调优（P-Tuning）的早期方法使用提示编码器产生连续嵌入，并与离散提示组合 [123]；其提示可依任务模板放置，而非必然只是一个字符串前缀．原始方法也研究过同时调整语言模型的设置，因此不能从名称推定基座始终冻结．本章在 PEFT 比较中只考虑冻结基座的配置．

深层提示调优（P-Tuning v2）将连续提示扩展到多层，并针对不同任务调整训练方式 [124]．它与逐层前缀属于紧密相关的方法路线，但输入布局、参数生成和任务头可能不同．比较时应写出真实计算位置与保存状态，不能把 Prompt、Prefix、P-Tuning 三个名字当作完全同义词．

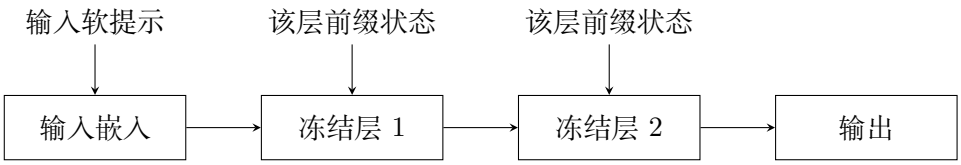


图 21.3: 输入软提示只在入口增加可训练向量；逐层前缀在多个注意力层直接提供状态. 图示比较插入位置，不表示必须同时启用两种方法.

21.7.4 选择性微调及方法比较

偏置微调（BitFit）只训练选定偏置 [125]；可训练词元方法只开放指定词嵌入行. 它们仍属于改变已有参数的策略. 无偏置架构中，可用偏置集合可能很小或为空；新增词元还涉及词表、输入输出嵌入绑定和保存范围，不能只交付几行矩阵而忽略词元编号协议.

表 21.2: 不同适配方法的主要资源与表达边界.

方法	可训练对象	主要约束
全参数微调	基座权重	状态多，通常每任务保留完整权重
LoRA	低秩因子	更新秩与目标位置受限；线性旁路可合并
QLoRA	量化基座上的因子	基座存储较小；仍需激活和输入梯度
瓶颈 Adapter	非线性瓶颈模块	增加算子，通常不能线性合并
IA^3	激活缩放向量	通道缩放表达受限，吸收位置须正确
Prompt	输入连续向量	增加输入位置，梯度穿过冻结网络
Prefix/深层提示	多层前缀状态	增加注意力状态与读取成本
选择性微调	指定原参数	可用集合依架构而异，需保存实际变化

表 21.2 中的资源特征用于提出候选方案，不能替代测量. 对于同一任务，应比较质量、有效词元吞吐、峰值内存、制品大小、加载时延、合并能力和目标后端支持. 尤其不能在改变训练数据或上下文长度的同时，把全部收益归因于 PEFT 方法.

21.8　训练及制品生命周期

21.8.1　可训练集合及数据目标同时受控

算法 21.1　冻结基座的适配训练

输入：固定基座与分词协议、目标模块集合、适配方法配置、训练数据和停止预算．**输出：**任务参数与依赖清单．**状态：**适配参数、优化器状态、数据位置、有效标签累计量．

1. 加载基座，解析实际模块和形状；冻结基座参数．按方法注入并初始化适配变量，建立精确可训练名单．
2. 优化器只接收该名单．对每批样本形成输入、边界与标签掩码；有效标签数为零的批次按预定策略处理．
3. 在保留必要输入梯度的计算图上前向，计算目标损失，再反向传播到可训练变量；按实际有效标签归一化梯度累积．
4. 完成规定的梯度处理与参数更新，记录数据和优化器状态；到达词元预算、验证集停止条件或明确失败条件时停止．
5. 保存适配变量、所有额外训练模块和不可变依赖．记录合并状态及后续重载、任务与性能验收所需条件．

不变量：参数更新范围等于声明范围；冻结基座不被更新，但必要的梯度链保持连通．训练状态保存与部署适配器保存具有不同职责．

例如面向对话的监督训练只对指定助手回答计算损失，系统消息、用户消息和填充不应意外进入目标．掩码的数学含义由有效标签集合定义，某个库使用的忽略索引只是实现约定．训练模板与部署模板可以为记录监督区间采用不同标记机制，但实际模型词元序列必须遵守同一协议．第十八章《训练数据工程基础》进一步讨论了这些消费视图与版本关系．

21.8.2　角色一致性的能力边界

身份与角色一致性适配可以作为一个具体任务：模型在身份询问、错误前提和能力边界问题中稳定遵循产品定义，同时在普通任务中避免反复自我介绍．这样的训练改变行为分布，不提供模型具有意识的证据，也不形成租户隔离或工具权限边界．

应按信息变更周期分层：在适配器生命周期内稳定的公开角色与行为约束可进入训练；当前日期、部署版本、租户和可用工具来自运行时配置；账户状态、实时事实和检索结果来自真实观察．把频繁变化的信息固化进权重，会把一次配置更新变成训

练与发布问题.

为识别适配器的真实增量，角色测试中不能只使用已经直接给出全部答案的提示. 还应保留部署上下文下的组合测试，以及翻译、数学等非触发控制任务. 训练与测试按提示家族隔离，分别报告角色稳定性、错误前提处理、动态事实边界和过度触发，而非只统计几个名字是否正确. 干净基座、启用适配器和从干净基座重载适配器构成三种必要的比较对象.

21.8.3 绑定、组合及回滚

适配器不是独立模型. 其版本至少绑定基座权重身份、架构、词表与模板、目标模块及形状、方法配置、额外可训练对象、量化条件、数据与代码版本. 基座名称相同而权重不同，或矩阵形状相同而语义位置变化，都不能保证兼容.

动态加载服务需要把请求绑定到具体适配器与组合配置. **适配器注册表**（Adapter Registry）记录依赖和发布状态，**适配器路由**（Adapter Routing）按服务端授权选择版本，而非让模型自行判断当前租户. 不同适配器通常会改变隐藏状态和键值缓存，因此缓存身份也必须包含影响其生成的模型与适配配置；切换后不能无条件沿用旧缓存.

算法 21.2 适配器发布与恢复

输入：适配器原件、依赖清单、批准的组合与部署模式. **输出：**可回滚的发布版本. **状态：**基座版本、适配器版本、路由版本及必要缓存命名空间.

1. 在干净基座上恢复完整训练变化，检查依赖和张量语义；拒绝不匹配版本，不静默跳过缺失模块.
2. 在固定输入和明确精度条件下比较恢复前后的数值输出，再评估目标能力与原有能力保留.
3. 若合并或重新量化，生成新的不可变制品并独立评估；若动态组合，按实际组合与系数评估，不复用单适配器结论代替.
4. 发布新的路由与缓存身份，保留前一版本. 出现回滚条件时恢复路由并隔离不兼容缓存，停止向新请求分配失效版本.

不变量：请求能追溯到实际基座、适配器及组合；未合并原件保留，恢复不依赖对低精度权重反复加减.

参数效率的准确含义

PEFT 压缩的是每个任务需要学习与维护的参数变化. 其收益只有结合梯度路径、激活成本、任务质量和部署依赖才完整. 冻结不等于不计算，低秩不等于模型低

秩，四比特基座不等于四比特训练，成功重载也不等于任务验收通过.

21.9　微调框架的职责及组合

框架选型首先要区分模型实现、可训练参数管理、训练目标和运行组织. 它们处于不同层次，同一训练过程常同时使用多个项目；比较框架时，应先确定是否在比较同一种职责.

组件	负责的对象	需要核对的边界
Transformers	模型结构、分词器、权重及生成接口	架构支持、聊天模板、特殊词元和模型修订.
PEFT	LoRA 等适配方法、参数注入与适配器制品	实际目标模块、冻结范围、额外保存层及基座绑定.
TRL	SFT、偏好优化和策略训练等目标的训练器	数据语义、监督掩码、参考模型与奖励定义.
LLaMA-Factory、Axolotl	配置驱动的数据处理、训练与导出流程	最终生效配置、模板映射和底层依赖组合.

Transformers 与 PEFT 分别提供模型和适配器层接口 [126, 127]；TRL 的 SFT 训练器可以与 PEFT 组合 [128]. LLaMA-Factory 与 Axolotl 则把多项训练步骤组织成配置化工作流 [129, 130]. 集成程度较高可以减少重复编排，但研究者仍须知道配置如何改变底层目标.

以问答微调为例，分词器把角色与文本转成词元，训练器确定哪些位置受到监督，PEFT 决定哪些参数获得梯度，分布式运行时再决定这些参数与状态怎样放到设备上. 若切换框架时同时改变聊天模板、回答掩码和有效批量，观察到的收益便无法只归因于框架实现. 应先对同一条样本核对词元、标签和损失，再比较训练速度.

使用集成框架时，应保存解析后的完整配置及依赖版本，而不只保存手写的几行覆盖项. 默认模板、填充、打包、优化器和检查点形式都可能随版本变化. 适配器导出后还应在干净基座上重载，验证张量绑定和固定输入输出；只有训练过程成功结束，尚不足以证明制品可以交付.

需要直接研究训练目标和数据处理时，可先采用 Transformers、PEFT 与 TRL 的组合，使监督链条便于检查；需要重复管理多种数据格式与训练任务时，可采用集成框架，并保留同样的数据与数值验收. 框架不会替代本章推导的参数预算、可训练范围和合并条件.

第二十二章　训练工程优化

高效微调研究如何在达到任务质量要求的条件下降低显存、计算和时间成本，适用于单设备、单机多卡和多机执行. 监督微调规定数据与损失，参数高效微调规定可训练参数集合，本章讨论参数集合、数值精度、批量组织与并行策略如何共同影响资源和质量. LoRA 等方法的参数化与梯度机制见第二十一章《参数高效微调》；本章侧重方案组合与选择，跨设备通信和分片机制见第二十九章《分布式训练》.

比较方案时，应固定监督词元、验证集和质量要求，同时记录各设备峰值显存、全局有效监督词元吞吐、达到目标质量的时间与设备时成本. 设备数量与学习阶段分别描述执行条件和学习路线，不能据此限定高效微调的适用范围.

22.1　显存预算及可训练集合

设冻结参数数量为 P_f，可训练参数数量为 P_t，每个冻结参数的实际存储开销为 b_f，每个可训练参数及其梯度、优化器状态的总字节数为 b_t. 单设备峰值可表示为

$$M_{\text{peak}} = P_f b_f + P_t b_t + M_{\text{act}} + M_{\text{temp}} + M_{\text{runtime}}. \tag{22.1}$$

量化尺度、分组元数据和对齐开销应计入 b_f；主参数副本与优化器矩状态应计入 b_t. 激活项取决于序列长度、微批量、可训练模块的位置和重计算策略. 冻结参数仍然参与前向计算；当梯度需要经过冻结层传向更早的适配器时，该层的反向计算也必须保留.

例如冻结基座有 10^9 个参数，每参数实际开销为 0.6 字节，可训练参数有 10^7 个，每参数训练状态为 16 字节，两项合计 7.6×10^8 字节. 若激活、临时空间和运行时另占 3×10^9 字节，则峰值为 3.76×10^9 字节. 只报告可训练参数比例会遗漏其中占比最大的执行开销. 该算例是预算计算，实际容量需由完整训练步的峰值测量确认.

22.2　有效监督词元及梯度累积

设一次更新包含 G 个微批次，第 j 个微批次的有效监督词元数为 n_j、损失和为 S_j．在固定标签掩码下，令 $N=\sum_j n_j>0$，则

$$\mathcal{L}=\frac{\sum_j S_j}{N},\qquad \nabla\mathcal{L}=\sum_{j:n_j>0}\frac{n_j}{N}\nabla\overline{\mathcal{L}}_j,\quad \overline{\mathcal{L}}_j=\frac{S_j}{n_j}. \tag{22.2}$$

因此，逐微批次反向传播前需要按有效监督词元数加权．提示词元和填充词元可能参与前向计算，却不属于损失分母．若两个微批次分别有 2 和 6 个监督词元，平均梯度分别为 1 和 3，则目标梯度为 $\frac{2\times1+6\times3}{8}=2.5$，简单平均得到 2，改变了优化目标．

所有微批次应使用同一参数版本；一次累积窗口结束后再归一化、裁剪、更新参数和优化器状态．学习率日程按实际更新计时．累积降低单次前向的容量需求，但不减少每个样本所需的计算．涉及批内统计的层、随机掩码以及浮点归约顺序会影响累积与完整批次的数值一致性．

22.3　序列长度及数据装箱

对批内长度 $\ell_1,\dots,\ell_B$，统一填充到 $\ell_{\max}$ 时，填充位置比例为

$$\rho_{\mathrm{pad}}=1-\frac{\sum_j \ell_j}{B\ell_{\max}}. \tag{22.3}$$

长度为 $(2,2,6,6)$ 的样本合为一个批次需要 24 个位置，分为两个等长批次需要 16 个位置．长度分桶还需保留随机化和任务混合，避免形成固定的长度课程．注意力的序列长度代价与前馈层不同，位置数减少的比例不能直接解释为端到端加速比．

将多条指令装入同一序列时，应维护样本边界、位置语义和回答区域掩码．独立样本通常还要求阻止跨样本注意力；插入终止词元本身不能保证隔离．截断则需要明确保留哪些监督区域，并检查是否系统性地排除了长回答或特定任务．

22.4　混合精度及激活重计算

混合精度分别规定权重存储、乘法输入、累加、梯度和优化器状态的格式．损失缩放后的梯度须在裁剪前还原；出现非有限值时，参数更新和优化器时间应遵循一致的跳步规则．数值稳定性的机制与恢复条件见第十九章《训练稳定性》．

激活重计算保存部分前向边界，在反向时重新求值区间内部．对 L 层均匀链式网络，每层一个大小为 A 的激活单位，每隔 k 层保存边界时，近似存储为

$$M_{\mathrm{act}}(k)\approx A\left(\frac{L}{k}+k\right). \tag{22.4}$$

将 k 暂视为正实数，导数在 $k=\sqrt{L}$ 时为零，得到近似最小值 $2A\sqrt{L}$. 若 $L=64$、$A=16$ MiB，选择 $k=8$ 得到 256 MiB，全部保存则为 1024 MiB. 边界、临时张量和非均匀层会改变实际峰值.

重计算应复用相同的随机函数状态，并避免重复修改持久缓冲. 若每层前向代价为 F，反向为 $2F$，额外重算一次前向会把近似工作量从 $3LF$ 增加到 $4LF$. 节省的显存可用于更长序列或更大微批量，但是否降低总时间需测量完整训练步. 相关存储构造见 [131].

22.5　分布式微调方案的选用

先区分容量与吞吐瓶颈. 完整微调状态能够在每个设备容纳时，可使用分布式数据并行（DDP）让不同副本处理不同样本，以提高全局吞吐；它保留各副本的模型与训练状态，不能直接解决单副本容量不足. 参数高效微调减少需要同步的可训练梯度，但冻结基座的存储、前向计算以及必要的反向路径仍然存在.

若瓶颈是优化器状态、梯度或参数的重复存储，应根据占比选择 ZeRO 或全分片数据并行（FSDP）等状态分片方案. 可训练参数很少时，优化器状态分片的绝对收益可能较小；基座参数分片能够降低常驻容量，却增加参数收集及临时缓冲. 若瓶颈来自长序列激活，应同时评估重计算、微批量和上下文切分，而非仅扩大参数分片组. 张量或流水并行可进一步切分模型执行，但需要评估通信、流水气泡和实际设备拓扑. 这些机制的计算与通信推导集中于第二十九章《分布式训练》.

全局有效批量取决于数据并行副本数、微批量、累积次数和有效监督词元数. 增加设备后，应明确保持全局批量还是增加每步监督量；两者改变更新次数和学习率日程的方式不同. 张量并行成员处理同一批数据时，不能重复计入全局监督词元. 各副本监督长度不等时，还需按全局词元数归一化，保证执行策略保持同一训练目标.

例如，固定监督预算和质量要求下，单设备耗时 100 分钟，两设备耗时 60 分钟，则时间加速比为 $\frac{5}{3}$，设备时从 100 设备分钟增至 120 设备分钟. 该构造算例说明缩短完成时间与降低计算成本是不同目标；若增加设备同时改变监督预算，就不能据此计算同任务加速比. 多机选择还应计入数据读取、通信尾部、验证与检查点保存时间.

量化基座、适配器训练与状态分片的组合，需要核对存储格式、可训练与冻结参数的组织、反向传播、优化器和检查点恢复是否相容. 支持某一种方法不等于支持所有组合；选择后应对完整更新和恢复过程做数值与资源验证.

22.6　微调执行评估

先固定模型、分词器、模板、回答掩码和可训练模块，再用短序列检查前向损失与梯度. 随后按目标长度测量一个完整更新窗口，计入优化器首次分配的状态；依据

峰值调整微批量、累积步数和重计算范围. 每次调整后同时检查损失分母、有效批量和随机状态，保证资源设置与训练目标一致.

比较全参数微调、LoRA 和 QLoRA 及其并行组合时，记录验证质量、全局有效监督词元吞吐、各设备峰值显存、总更新次数、墙钟时间和设备时. 分布式执行还应核对数据分片、梯度同步、全局归一化、非有限值跳步与恢复一致性. 检查点应包含适配器或完整参数、基座身份、优化器状态、学习率进度、随机状态与数据游标；分片检查点还需明确恢复所需的并行配置或重分片能力. 发布时核对适配器合并、量化转换与服务模板造成的质量变化.

第二十三章　预训练原理

预训练决定模型首先学习怎样的文本分布，以及有限资源应投入多少参数和多少训练词元. 两者不能分开讨论：同样的词元数量可能对应不同语言、领域与重复程度；同样的参数数量也可能因训练不足而不能充分发挥作用. 规模定律提供的是受条件约束的预算模型，训练目标则规定该模型究竟在衡量什么.

本章从有效监督词元上的最大似然出发，讨论文档边界、数据混合和预训练任务，再推导给定计算预算下的模型与数据配置. 第十八章《训练数据工程基础》的数据流水线负责构造合格语料，第二十七章《模型规模及资源预算》的算力规划负责把工作量映射到设备、网络和存储，第十九章《训练稳定性》的训练优化负责参数如何更新. 本章连接这三者，回答“用什么分布、优化什么目标、分配多少训练量”.

23.1　预训练所估计的分布

23.1.1　由联合概率到条件预测

预训练（Pretraining）通常从随机初始化开始，利用大规模数据学习可迁移的表示与生成分布. 已有基座上的**继续预训练**（Continued Pretraining，CPT）沿用相近目标，但初始参数已经包含先前训练分布的信息，不能把它当作同等数据预算下的从零训练.

设文档 $x = (x_1, \dots, x_T)$，在给定起始标记与终止规则下，链式法则给出

$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t}), \qquad -\log p_\theta(x) = -\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}). \tag{23.1}$$

θ 为可学习参数，$x_{<t}$ 是当前位置之前的词元. 该概率分解并不要求文档内部各词元独立；相反，相关性通过条件分布表达. 训练时使用真实前缀，称为**教师强制**（Teacher Forcing）. 推理时前缀包含模型自身生成的内容，因此训练似然改善与生成质量改善有关，却不是完全相同的判据.

若训练样本来自分布 P，在固定上下文定义下，期望负对数似然可写为

$$\mathbb{E}_{c\sim P}\mathbb{E}_{y\sim P(\cdot|c)}[-\log p_\theta(y \mid c)] = \mathbb{E}_c H(P(\cdot \mid c)) + \mathbb{E}_c D_{\mathrm{KL}}(P(\cdot \mid c) \| p_\theta(\cdot \mid c)). \tag{23.2}$$

这里 c 表示可见上下文，y 为预测目标，H 为条件熵，D_{KL} 为相对熵. 第一项在分布固定时与参数无关，最小化损失因而推动模型逼近该训练条件分布. 改变文档筛选、采样权重或可见前缀，就改变了这里的 P；数据处理不是目标之外的中性准备步骤.

23.1.2　符号及假设

符号	含义
P_i, π_i	来源 i 的有效目标词元分布及其训练混合权重.
B, T, V	批次序列数、物理块长度与词表大小.
$m_{b,t}, S$	预测位置的监督掩码；当前聚合范围内有效目标总数.
N, D, C	模型参数规模、训练消费词元数与训练浮点运算量.
D_u, D_s, D_p	不同内容的可用词元量、有效监督量、实际处理位置量.
E, A, B_s, α, β	经验损失底座、两个正系数及两个正幂律指数.
κ, C_0	计算近似中的比例系数与给定计算预算.

B_s 使用下标，以避免与批次大小 B 混淆；N 在本章专指参数规模，不用于表示有效词元数.

预算模型的适用条件

似然比较固定分词器、验证分布、边界策略与监督区域. 规模拟合针对相近的密集 Transformer 架构、数据质量、训练长度与优化完成程度；参数计数和词元计数采用同一口径. 幂律是有限观测范围内的经验假设，不是对任意任务、模型或重复语料都成立的定理.

23.2　有效词元上的目标及归一化

23.2.1　标签配对及稳定交叉熵

考虑长度为 T 的输入块 $(x_0, \dots, x_{T-1})$. 模型输出 $Z \in \mathbb{R}^{B\times T\times V}$，位置 t 的向量 $z_{b,t,:}$ 预测 $x_{b,t+1}$，因此块内候选目标只有 $T-1$ 个. 若库中标签最初复制输入，损失内部须将输出的前 $T-1$ 位与标签的后 $T-1$ 位配对. 另一种等价接口显式构造目标序列，并让输入相对于目标右移. 两种实现不能同时移位.

令 $m_{b,t}=1$ 表示这一预测配对参与监督，$m_{b,t}=0$ 表示忽略，则

$$\ell_{b,t} = \log \sum_{v=1}^{V} \exp z_{b,t,v} - z_{b,t,x_{b,t+1}}, \tag{23.3}$$

$$S = \sum_{b=1}^{B}\sum_{t=0}^{T-2} m_{b,t}, \qquad \mathcal{L} = \frac{1}{S}\sum_{b,t} m_{b,t}\ell_{b,t}. \tag{23.4}$$

实现对数求和指数时可先减去最大分数再加回，以避免指数溢出．$S=0$ 的批次没有定义平均训练目标，必须跳过更新或在构造阶段排除，不能把分母替换成一之后声称取得零损失．

由 softmax 的导数可得

$$\frac{\partial \mathcal{L}}{\partial z_{b,t,v}} = \frac{m_{b,t}}{S}\left(p_{b,t,v} - \mathbf{1}[v = x_{b,t+1}]\right). \tag{23.5}$$

因此，分母不仅改变日志数值，也改变梯度权重．填充掩码控制哪些位置是实际输入，因果掩码控制可见性，监督掩码控制哪些配对贡献梯度；三个对象必须分别定义．

23.2.2　批次均值的聚合偏差

设两个微批次分别含 $S_1=2$、$S_2=6$ 个有效目标，对应平均损失 $L_1=1$、$L_2=3$．整体目标为

$$L = \frac{2\times 1 + 6\times 3}{2+6} = 2.5, \tag{23.6}$$

而简单平均得到 2．后者给短批次中的每个词元三倍权重．对梯度累积，正确的整体梯度为

$$g = \sum_{k} \frac{S_k}{\sum_j S_j} \nabla_\theta L_k. \tag{23.7}$$

混合精度和裁剪见第十九章《训练稳定性》，分布式梯度平均见第二十九章《分布式训练》；目标层面的原则是先确定总监督量，再分配各微批次的权重．

按文档等权也是合法但不同的目标．若文档 i 的有效长度为 T_i，词元等权与文档等权分别为

$$L_{\text{token}} = \frac{\sum_i T_i L_i}{\sum_i T_i}, \qquad L_{\text{doc}} = \frac{1}{M}\sum_{i=1}^{M} L_i. \tag{23.8}$$

文档等权提高短文档中单个词元的影响．选择哪一种取决于希望拟合的统计单位，不应由数据加载器的默认归约偶然决定．

23.2.3　困惑度的比较条件

困惑度（Perplexity，PPL）定义为 $\exp(\mathcal{L})$，其中对数以自然底计算．它等于正确目标概率几何平均数的倒数．上例整体困惑度为 $\exp(2.5)\approx 12.18$，既不等于两个批次困惑度的算术平均，也不等于 $\exp(2)$．

困惑度不是模型可能输出的词数，更不是事实错误率．它的比较依赖相同分词器、验证文本、可见上下文和目标掩码．使用相同文本而改变分词器，会改变目标单位及条件分解；此时可以补充每字节负对数似然，但仍须统一规范化和文本覆盖．[1]

[1] 以自然对数计量的损失常称为每词元 nat；换算为 bit 应除以 $\log 2$．改变对数底不改变同一实验中的排序，却会改变所报告的系数与数值．

23.3　序列装箱的条件分布

23.3.1　终止标记及注意力隔离

序列装箱（Sequence Packing）把多个片段安排进物理块以减少空位. 若独立文档 a、b 被连成 $a, \mathrm{EOS}, b, \mathrm{EOS}$，采用普通因果注意力时，$b$ 的预测条件包含 a. 加入终止标记只是提供边界信息，并不把联合概率自动变为 $p(a)p(b)$.

若训练目标要求文档独立，设位置 t 的文档编号为 $d(t)$，可见性需要满足

$$A_{t,j} = \mathbf{1}[j \leqslant t]\mathbf{1}[d(j) = d(t)]. \tag{23.9}$$

同时必须处理跨段目标配对. 屏蔽 a 对 b 的注意力，并不自动阻止 a 的末尾输出预测 b 的第一个词元. 通常每段含起始标记，由起始标记预测该段首词元；上一段终止位置到下一段起始位置的配对不计入损失. 是否重置位置编号也需固定，因为位置分布会因此改变.

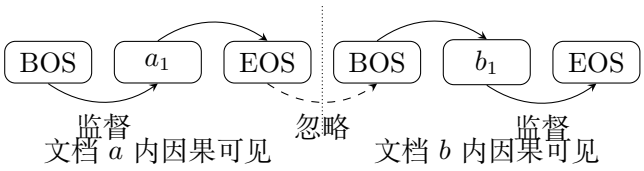


图 23.1: 独立文档装箱的两种边界：注意力禁止跨文档读取，损失忽略跨文档预测配对. 实线表示有效相邻监督.

对于连续章节、代码文件集合或保留上下文的对话，跨片段可见也可能是刻意设计. 关键在于物理切片是否对应统计独立单位. 把同一文档拆段后完全隔离，会损失长距离条件；把不相关文档连接而不隔离，则引入人为前缀分布.

23.3.2　块边界及尾部损失

连续词元流切成多个独立输入块时，每个块最后位置缺少块内后继目标. 若采用重叠一个词元的窗口，可保留该边界预测，但更大的重叠会重复监督内容. 若只保留重叠区域作为上下文，应将已经计入过的目标屏蔽，以免改变权重.

尾部丢弃、填充、跨分片续接分别影响覆盖率、计算利用率和读取状态. 假设 100 个文件各有 129 个词元，每个文件单独按 128 切块会丢弃 100 个词元；跨文件保留缓冲尾部可减少这一损失，却必须另外保留文档边界，不能以减少浪费为理由改变条件结构. 验证集也必须使用同一明确策略，否则损失变化可能只是可见上下文或被评分目标不同.

装箱率与监督率是不同指标

非填充位置占物理位置的比例描述装箱利用率；有效目标占物理位置的比例描述监督密度．边界标记、仅供上下文的前缀和掩码区域均可能消耗计算而不贡献直接损失．规模预算必须同时保留这两类计数．

23.4　数据混合及长尾覆盖

23.4.1　重要性权重

设来源 i 的目标词元分布为 P_i，希望以权重 π_i 训练，则

$$P_{\text{train}} = \sum_i \pi_i P_i, \qquad L(\theta) = \sum_i \pi_i L_i(\theta), \qquad \sum_i \pi_i = 1. \tag{23.10}$$

若加载器先以概率 q_i 选择来源，再抽取平均有 $\bar{s}_i$ 个有效目标的文档，聚合后的词元比例近似为

$$\pi_i = \frac{q_i \bar{s}_i}{\sum_j q_j \bar{s}_j}. \tag{23.11}$$

因此，两个来源按文档各取一半，若平均有效长度分别为 100 与 900，实际词元比例为一比九．要得到词元各半，需要文档采样概率九比一．对具有不同监督密度的任务，应使用有效目标长度而不只使用原始文档长度．

常用温度平滑以合格可用词元量 n_i 定义 $\pi_i \propto n_i^{\tau}$，$0 \leqslant \tau \leqslant 1$．这里 τ 是混合指数，并非 softmax 温度或规模幂律指数．$\tau = 1$ 保留规模比例，$\tau = 0$ 对非空来源等权．降低指数会提高小来源相对份额，但不能增加该来源实际拥有的不同信息．

23.4.2　数据混合配比

例 16.1

设三个来源可用词元量为 90、9、1 十亿．自然比例为 $0.90, 0.09, 0.01$；取 $\tau = \dfrac{1}{2}$ 后，权重与 $\sqrt{90}, 3, 1$ 成正比，约为 $0.7034, 0.2224, 0.0741$，最后一项因四舍五入可作相应校正．计划消费 200 十亿词元时，各来源的名义遍历倍数为

$$e_i = \frac{D\pi_i}{n_i}, \tag{23.12}$$

约为 $1.56, 4.94, 14.83$．小来源的词元权重提高约七倍，其重复程度也显著增加．该指标是消费量与库存量之比，不证明每篇文档恰好被访问同样次数．

带替换采样 m 次、均匀抽取 n 个不同对象时，期望见过的不同对象数为

$$U = n\left[1 - \left(1 - \frac{1}{n}\right)^{m}\right]. \tag{23.13}$$

推导只需对每个对象定义是否至少出现一次的指示变量，再求期望之和．$m = n$ 时，大样本近似只有 $(1 - e^{-1})n$ 个不同对象被访问．真实文档长度不等且存在近重复，不能直接把此对象公式视为不同信息词元的精确计数．

23.4.3　训练分布及评测分布

训练混合比例表达资源投入，评测混合比例表达质量偏好，两者不必相同．若目标评测分布权重为 r_i，训练采样权重为 $\pi_i > 0$，可用 $\dfrac{r_i}{\pi_i}$ 对来源损失作重要性加权，使期望匹配目标，但极端权重会放大方差．若目标来源完全没有训练支持，加权无法产生缺失样本．

质量筛选也会改变语言和文体覆盖．低困惑度文本可能更贴近评分模型熟悉的分布，却未必覆盖罕见术语或形式化推导．预训练配方应分别记录准入、质量层、语言、领域、长度和重复程度；不宜把所有因素压成一个无法解释的“优质分数”．数据流水线的实现见第十八章《训练数据工程基础》，本章关心这些决策怎样改变 P_{train}．

23.4.4　课程及继续预训练

课程学习（Curriculum Learning）使混合权重或难度随训练进程变化，记为 $\pi_i(u)$，其中 u 可以是累计词元数．即便总消费量相同，先通用后领域与交错混合也不一定得到相同参数，因为连续梯度更新并不交换顺序．课程应同时说明阶段长度、过渡方式和通用能力保持条件．

继续预训练通常需要评估领域收益与旧分布退化．保留部分通用数据可以约束漂移，却没有一个跨模型通用的保证比例．小规模配方比较应固定有效词元预算、候选模型与评测切片，再决定是否扩展；测试集不得反复用于选择混合权重．

23.5　不同预训练任务的监督结构

自回归、掩码恢复与去噪的区别首先在于可见信息和被预测变量，而不只是模型名称．统一表示为：从原始文档 x 通过变换 $q(c, y \mid x)$ 产生条件 c 与目标 y，优化

$$L_q(\theta) = \mathbb{E}_x \mathbb{E}_{(c,y)\sim q(\cdot\mid x)} \left[-\sum_t \log p_\theta(y_t \mid c, y_{<t}) \right], \tag{23.14}$$

并明确按目标词元还是样本归一化．对于并行预测多个掩码位置的模型，求和项使用各位置的条件概率，不意味着这些概率构成任意长度文本的自回归联合概率．

任务	可见信息	预测内容	主要边界
自回归语言建模	当前词元之前的前缀	下一词元	直接匹配逐步续写；训练前缀是真实数据.
掩码语言建模	含遮蔽的双向上下文	被选中的词元	擅长双向表示；训练损失不能直接当作原文自回归困惑度.
跨度去噪	删除跨度后的文档与哨兵标记	被删跨度的序列	需要固定破坏分布、跨度顺序和停止协议.
前缀语言建模	完整前缀与已生成后缀	后续部分	前缀通常不计目标损失，监督密度随切分点变化.
中间填充	原文前段与后段	中间缺失内容	需要重排及边界标记；序列顺序不同于原文.

T5 的研究比较了文本到文本框架中的去噪目标与迁移设置 [132]. 以原文“甲乙丙丁戊”为例，删除“乙丙”并以哨兵 s_0 替代，条件可写为“甲 s_0 丁戊”，目标为“s_0 乙丙 s_1”；哨兵用于定位缺失跨度，不能临时解释成通用起始标记. 这里的符号例用于说明条件结构，具体制品的终止标记仍以其协议为准.

中间填充（Fill-in-the-Middle，FIM）可以将文档拆为前段 a、中段 b、后段 c，重排为带边界标记的 a, c, b，从而让因果模型在生成 b 时同时读取原文两侧 [133]. 它并未让因果注意力读取重排序列的未来，而是通过改变顺序使两侧内容成为已知前缀.

混合任务可写为 $L = \sum_k \lambda_k L_k$，但 λ_k 的含义依赖 L_k 的归一化. 如果一个任务只监督 15% 位置，另一个监督近乎全部位置，相同样本抽样率并不等于相同目标词元贡献. 比较任务还应报告处理位置量和实际计算量，否则较低损失可能来自较容易的条件或更高的监督资源，而不是更优的表示学习机制.

23.6 参数、数据及计算的统一预算

23.6.1 词元计量口径

D_u 表示去重后可用内容的规模，D_s 表示累计有效目标数，D_p 表示模型实际处理的位置数. 重复多个遍历增加 D_s 和 D_p，不会同比增加 D_u；填充和仅供上下文的位置增加 D_p，不一定增加 D_s.

设每次参数更新有 R 个数据并行进程，每进程累积 G 个微批次，每批次 B 个长度 T 的块，训练 K 步，在规则相同且无额外掩码时

$$D_p = KRGBT, \qquad D_s = KRGB(T-1). \tag{23.15}$$

例如 $K = 1000, R = 4, G = 2, B = 8, T = 1024$，得到 $D_p = 65536000$，$D_s = 65472000$. 跨段屏蔽和填充会进一步降低 D_s，实际应累加监督掩码，而不是永远套用 $T - 1$.

23.6.2　训练计算近似的来源

对主要由密集矩阵乘法构成的网络，前向每个词元约需与参数规模成比例的运算；反向包含输入与权重梯度两部分．因此常用

$$C \approx \kappa N D, \qquad \kappa \approx 6 \tag{23.16}$$

作为数量级近似．这里乘加计为两次浮点运算，D 通常接近处理词元量；只有在监督密度高且边界损失很小时，才近似等于有效目标量．[2]

式 (23.16) 没有显式表达随上下文增长的注意力开销、激活重计算和特殊算子成本．长上下文时应重新估计 κ 或改用更详细模型．设备峰值乘运行时间是硬件上限；通信、存储等待与内核效率决定实际完成的模型计算，不应把峰值预算直接代入损失定律．

23.7　经验规模定律的拟合

23.7.1　幂律描述及数据条件

规模定律（Scaling Law）以经验函数描述模型规模、数据和计算与损失之间的关系．Kaplan 等人的研究系统分析了语言模型交叉熵随这些变量变化的规律 [10]；后续 Chinchilla 工作重新研究固定训练计算下的参数与训练词元分配 [11]．这些研究的配置与拟合程序不同，不能把其中的指数拼接成一条统一物理定律．

本章使用一个便于分析的加性模型

$$L(N, D) = E + AN^{-\alpha} + B_s D^{-\beta}, \qquad A, B_s, \alpha, \beta > 0. \tag{23.17}$$

E 表示该经验模型中的渐近底座，$AN^{-\alpha}$ 与 $B_s D^{-\beta}$ 分别描述参数和数据受限时的损失余量．有限观测不能证明拟合出的 E 就等于真实语言熵；数据、分词器和条件窗口的变化也会改变它．

单独固定数据研究参数增长时，若已知底座 L_∞，有

$$\log(L - L_\infty) = \log A - \alpha \log N. \tag{23.18}$$

例如参数每扩大四倍、损失余量减半，则 $\alpha = \dfrac{\log 2}{\log 4} = \dfrac{1}{2}$．若错误地对 $\log L$ 作线性拟合，底座占比较高时斜率会明显偏小．把未知底座与指数同时从狭窄区间估计，也容易产生多组近似等价解．

[2] 参数口径可能包含或排除嵌入，稀疏专家模型还须区分总参数与激活参数．引用某篇规模拟合的系数时，不能更换参数定义后沿用原数值．

23.7.2 拟合应覆盖两个维度

仅在一条 ND 固定曲线上取样，会使参数与数据强相关，难以分别识别两者贡献. 应在多组参数规模和训练长度上设计实验，并保留独立规模点检验预测. 每个实验须具有可比的优化完成程度：若大模型使用明显不合适的学习率，拟合可能把优化不足误归因为规模效率差.

一种拟合形式为

$$\min_{E,A,B_s,\alpha,\beta} \sum_j w_j \, \rho\Big(\log L_j - \log[E + AN_j^{-\alpha} + B_s D_j^{-\beta}]\Big), \tag{23.19}$$

其中 j 为实验点，w_j 为可靠性权重，ρ 可取平方损失或稳健损失. 使用对数残差表示偏重相对误差；它不是唯一正确选择，应与预算决策关注的误差相匹配. 约束正系数可通过指数参数化实现.

同一训练运行中的多个检查点共享数据与随机轨迹，不能当成相互独立实验无限增加置信度. 应按训练运行或随机种子组织重采样，并报告外推点的误差区间. 残差若随规模、长度或来源系统变化，说明模型遗漏了因素，而不是增加小数位就能提高可信度.

算法 23.1 规模拟合与预算候选生成

1. 冻结分词器、数据混合、评测集合、上下文和参数计数口径；保留测试集不参与配方选择.
2. 设计覆盖多个 N 与 D 的训练网格，为每个规模分配合理优化配置，记录真实消费量与计算估计.
3. 以有效目标总和聚合固定验证集损失，保留运行身份、重复程度与异常说明.
4. 拟合式 (23.17)，在未参与拟合的规模点比较预测与实际损失，分析残差及参数不确定性.
5. 在给定预算下求连续候选，再映射到可部署的离散架构与可用数据范围；对相邻候选开展预算内比较.
6. 保存数据配方、拟合区间、候选选择依据与中止条件，使后续扩大规模有明确依据.

23.8　计算最优分配的推导

23.8.1　拉格朗日条件

将式 (23.17) 与 $C = \kappa ND \leqslant C_0$ 联立. 因为在假设范围内损失随 N, D 单调下降, 内部最优点会用尽预算. 定义拉格朗日函数

$$\mathcal{J} = E + AN^{-\alpha} + B_s D^{-\beta} + \lambda(\kappa ND - C_0). \tag{23.20}$$

分别对 N, D 求导并令其为零:

$$-\alpha AN^{-\alpha-1} + \lambda\kappa D = 0, \tag{23.21}$$

$$-\beta B_s D^{-\beta-1} + \lambda\kappa N = 0. \tag{23.22}$$

第一式乘 N，第二式乘 D，消去相同的 $\lambda\kappa ND$，得到

$$\alpha AN^{-\alpha} = \beta B_s D^{-\beta}. \tag{23.23}$$

该条件平衡的是按对数规模计算的边际收益；只有 $\alpha = \beta$ 时，两项损失余量才相等.

令 $K_c = \dfrac{C_0}{\kappa}$，代入 $D = \dfrac{K_c}{N}$，可得

$$N_* = \left(\frac{\alpha A}{\beta B_s}\right)^{\frac{1}{\alpha+\beta}} K_c^{\frac{\beta}{\alpha+\beta}}, \tag{23.24}$$

$$D_* = \left(\frac{\beta B_s}{\alpha A}\right)^{\frac{1}{\alpha+\beta}} K_c^{\frac{\alpha}{\alpha+\beta}}. \tag{23.25}$$

这是所选经验函数及计算约束的数学结论. 它本身并不能证明真实模型满足该函数.

23.8.2　最优点的唯一性及预算指数

令 $u = \log N$，固定预算后的非底座损失为

$$f(u) = Ae^{-\alpha u} + B_s K_c^{-\beta} e^{\beta u}. \tag{23.26}$$

其二阶导数

$$f''(u) = \alpha^2 Ae^{-\alpha u} + \beta^2 B_s K_c^{-\beta} e^{\beta u} > 0. \tag{23.27}$$

因此在没有额外边界的连续正数域中，驻点是唯一全局最优点. 存在显存上限、最小模型规模或数据库存约束时，最优解可能落在可行区边界.

最优规模随预算的指数分别为 $\dfrac{\beta}{\alpha+\beta}$ 与 $\dfrac{\alpha}{\alpha+\beta}$. 当两指数相等时，模型和数据都按预算平方根增加；预算扩大四倍时，两者各扩大两倍. 最优余量则满足

$$L_*(C) - E \propto C^{-\frac{\alpha\beta}{\alpha+\beta}}. \tag{23.28}$$

收益随计算增长而递减. 若 $\alpha \neq \beta$，最优 $\dfrac{D_*}{N_*}$ 也随预算变化；固定 “每参数多少词元” 的口诀无法代替式 (23.25).

23.8.3　计算最优分配

例 16.2

以下数值为便于手算而构造的模型，不是任何公开模型的拟合结果.定义 $n=\dfrac{N}{10^9}$、$d=\dfrac{D}{10^9}$，假设

$$L(n,d)=2+n^{-\frac{1}{2}}+4d^{-\frac{1}{2}},\qquad C_0=9.6\times10^{19},\quad \kappa=6. \tag{23.29}$$

计算预算给出 $nd=16$. 由边际平衡 $n^{-\frac{1}{2}}=4d^{-\frac{1}{2}}$，得 $d=16n$，联立后 $n_*=1,d_*=16$，预测损失为 4.

参数 n	词元 d	乘积 nd	预测损失
0.25	64	16	4.5
1	16	16	4.0
4	4	16	4.5

较小模型获得更多数据但容量项较大，较大模型压缩了训练长度而数据项较大. 两端计算量相同，不能通过比较参数规模直接判断结果. 预算扩大四倍后，最优为 $n=2,d=32$，损失 $2+\sqrt{2}\approx3.414$.

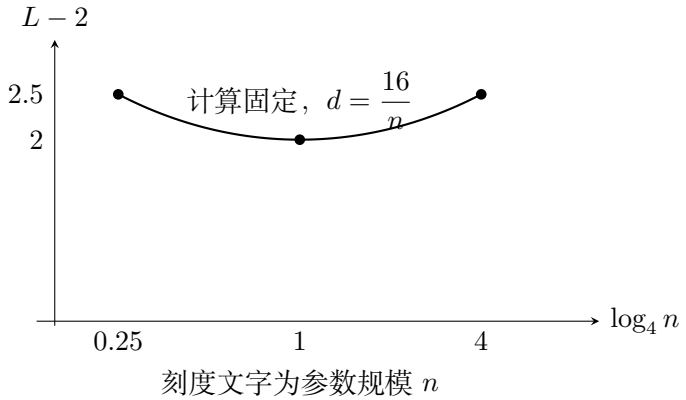


图 23.2: 数值例在固定计算下的损失余量. 曲线由本节解析函数计算；纵轴从零起算，曲线最低点对应参数与数据的边际收益平衡.

若可用不同数据只有四十亿词元且不允许重复，则 $d\leqslant4$. 在相同预算下，连续可行最优落在 $d=4,n=4$，损失为 4.5. 允许重复并不意味着可以继续沿原函数把 d 增至 16：原拟合如果主要使用新数据，就必须为重复训练重新建模.

23.9　规模结论的外推边界

23.9.1　数据受限及优化受限

经验式中的数据量通常建立在特定重复范围和质量分布上. 重复曝光有时仍能改善优化，却不能无限提供新的统计信息. 合成数据也必须考察新信息、错误相关性与

验证覆盖，不能因为文件数增加就把它等价计入原始独立数据预算.

同样，计算量相同但学习率、批次、衰减与停止时机不同，可能导致不同损失. 规模最优讨论要求各候选达到可比的预算内训练效率，而不是所有模型机械使用同一更新步数. 第十九章《训练稳定性》将展开优化器状态与调度；在本章的预算实验中，它们属于必须记录的控制条件.

23.9.2　训练最优及全生命周期最优

Chinchilla 式问题关注给定训练计算的模型选择. 部署时还可能关注模型容量、延迟和累计推理量. 若未来生成 Q 个词元，简化生命周期成本可写为

$$C_{\text{life}} \approx \kappa N D + \kappa_{\text{inf}} N Q, \tag{23.30}$$

其中 κ_{inf} 是给定推理场景下的近似系数. 较小模型经过更多训练，可能在大量调用时降低总成本；这改变了约束与优化目标，不能直接套用仅训练计算最优点. 真实推理成本还随上下文、缓存与硬件变化，上式仅说明新增目标项的方向.

23.9.3　损失及能力的不同尺度

验证交叉熵平滑下降，并不保证每个下游任务同时改善. 离散正确率可能受评分阈值、提示方式和样本量影响；某个能力切片也可能因混合配比下降. 模型选择应同时保留可比的预训练损失与领域、语言、长上下文等任务指标，不能只把损失变小解释为所有能力提升.

规模定律的正确用法

规模定律用于缩小预算候选范围，并提出可被实验否定的预测. 拟合区间、分布、架构或目标发生变化后，应重新评估系数与误差. 连续最优解是架构选型的起点，最终方案还须满足数据库存、设备容量和部署要求.

23.10　预训练的数据流及状态流

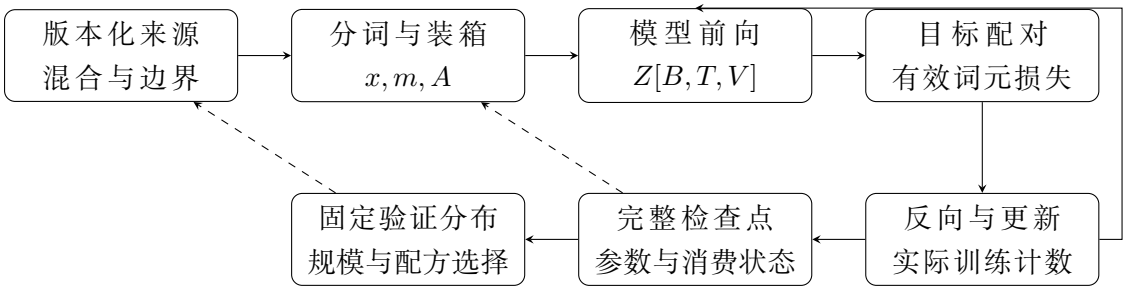


图 23.3: 预训练同时积累模型知识与可恢复状态. 实线表示计算和保存路径，虚线表示恢复或下一轮实验设计；验证结果不直接生成当前训练目标.

一次训练状态可记为 $s_k = (\theta_k, o_k, h_k, r_k, d_k)$，分别表示参数、优化器状态、调度状态、随机状态与数据消费位置. 一次更新为 $s_{k+1} = F(s_k, \mathcal{B}_k)$. 只恢复参数会改变后续状态转移，不能称为精确续训. 数据配比随进度变化时，还需要恢复其阶段位置；流式洗牌则需要能够重建缓冲与分片消费关系.

算法 23.2　按有效词元推进预训练

1. 固定初始模型、分词器、来源版本、配比日程、边界策略和总预算，分别初始化处理量与有效监督量计数.
2. 从可恢复采样状态读取下一组文档，构造输入、段编号、注意力与监督掩码；保留来源跨度以归属消费量.
3. 对一个更新范围汇总有效目标数，按式 (23.4) 与式 (23.7) 形成正确归一化梯度；无有效目标时不推进参数更新.
4. 完成优化器更新后推进调度和数据计数；记录各来源实际有效词元份额、重复程度与计算消耗.
5. 在预定评估点累计固定验证集的负对数似然总和与有效目标总数，分别报告总体及来源切片.
6. 在明确更新边界保存完整状态及不可变数据引用；达到预算后交付所选检查点、计量口径与独立评测结果.

训练过程平均损失反映历史参数在各批次上的表现，不等于最终参数的验证损失. 用于选择最佳检查点的验证集已经参与模型选择，最终测试集应继续隔离. 配套小型实践能够说明目标、更新与恢复机制；它的语料范围和预算并不构成通用基座能力的证据.

第二十四章　强化学习基础

监督微调给出应当模仿的回答，强化学习则给出行为产生的反馈，并据此调整未来行为的概率. 语言模型的输出是一个逐词元决策过程，工具使用还会改变外部环境. 要理解策略优化，必须先区分状态、观测、动作、奖励和回报，再解释为什么不可微的采样与环境仍能产生可估计的参数梯度.

本章从决策过程与价值函数出发，推导策略梯度、基线与优势估计，最后连接近端策略优化. 第二十章《监督微调》讨论示范模仿，本章讨论采样行为的期望回报. 奖励建模与直接偏好优化由第二十五章《人类反馈学习》展开，可验证推理及组相对方法由第二十六章《可验证推理训练》展开；本章为这些方法建立共同的数学基础.

24.1　决策过程及语言模型

24.1.1　马尔可夫决策过程

强化学习（Reinforcement Learning，RL）研究策略如何通过与环境交互最大化期望累积反馈. **马尔可夫决策过程**（Markov Decision Process，MDP）可表示为 $(\mathcal{S}, \mathcal{A}, P, r, d_0, \gamma)$：$\mathcal{S}$ 是状态集合，$\mathcal{A}$ 是动作集合，$P(s' \mid s, a)$ 是状态转移分布，r 描述奖励机制，d_0 是初始分布，γ 是折扣因子.

马尔可夫性要求给定当前状态与动作后，下一状态和奖励的条件分布不再依赖更早历史. 它是状态定义的充分性要求，而不是说任务没有记忆. 时间有限的问题可将剩余步数纳入状态，使最优行为能够随期限改变.

策略 $\pi_\theta(a \mid s)$ 给出动作概率. 一次**轨迹**（Trajectory）写为

$$\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_H), \tag{24.1}$$

r_t 在执行 a_t 后产生，H 为终止步数. 这里的奖励下标对应当前动作，部分文献将同一量记为 R_{t+1}，两种约定不可混用. [1]

[1] 本章区分即时奖励 reward 与累计回报 return. 中文资料偶尔都称“奖励”，推导时应以时间下标和求和定义识别具体对象.

24.1.2 文本生成的状态及动作

对给定提示 x，可令 $s_t = (x, y_{<t})$，动作 $a_t = y_t$ 为词表中的下一词元，转移为把该词元追加到前缀. 采样策略为模型条件分布. 若评分规则只依赖提示和最终文本，完整前缀足以描述纯文本生成的决策状态.

结束标记是合法动作，会进入终止状态. 提示词元由任务分布提供，不是当前策略采样动作；工具返回文本由环境产生，也不能像助手生成词元一样乘入策略动作概率. 若动作是一整次工具调用，则动作空间、时间尺度与成本单位均发生变化，不能把一次调用与一个词元的折扣无说明地混合.

24.1.3 部分可观测交互

工具或应用环境通常包含不可见状态，如数据库真实内容、服务内部进度和用户未表达的目标. 此时更适合**部分可观测马尔可夫决策过程**（Partially Observable Markov Decision Process，POMDP）：环境有状态 s_t，模型仅获得观测 o_t，策略依赖历史 $h_t = (o_0, a_0, \ldots, o_t)$.

理论上，已知环境模型时可用信念分布 $b_t(s) = P(s_t = s \mid h_t)$ 作为充分状态，并按

$$b_{t+1}(s') \propto O(o_{t+1} \mid s') \sum_{s} P(s' \mid s, a_t) b_t(s) \tag{24.2}$$

更新，O 为观测概率. 本式采用观测只依赖新状态的简化形式. 实际语言智能体通常没有完整环境模型，而是使用历史或记忆近似信息状态. 截断历史可能丢失必要信息，因此有限上下文不自动满足马尔可夫性.

24.1.4 符号及共同假设

符号	含义
s_t, a_t, r_t	决策前状态、所选动作及随后获得的即时奖励.
G_t, γ, H	从 t 开始的折扣回报、折扣因子与终止长度.
V^π, Q^π, A^π	状态价值、动作价值与优势函数.
$\pi_\theta, \mu, \pi_{\text{ref}}$	待优化策略、采样行为策略与参考策略.
$V_\phi, \delta_t, \widehat{A}_t$	参数化价值估计、时序差分残差与优势估计.
λ, ϵ	广义优势估计的混合参数与 PPO 裁剪宽度.
$\rho_t, W(\tau)$	单动作概率比与整轨迹重要性权重.

策略梯度的前提

初始分布和环境转移不直接依赖策略参数，奖励在本次策略更新中固定. 策略可微且在所讨论动作支持集上为正；奖励有界，轨迹有限，或 $\gamma < 1$ 使求和收敛. 交换导数与期望需要相应可积条件. 采样概率必须对应实际生成分布.

24.2　回报及 Bellman 关系

24.2.1　价值函数递推

定义

$$G_t=\sum_{k=t}^{H-1}\gamma^{k-t}r_k,\qquad G_t=r_t+\gamma G_{t+1},\qquad G_H=0. \tag{24.3}$$

有限时域可取 $\gamma=1$. $\gamma<1$ 表示相对降低远期奖励权重，也可能有助于估计稳定，但它改变任务目标；对相同最终奖励，不同长度的回答会获得不同折扣.

状态价值（State Value）与**动作价值**（Action Value）分别为

$$V^\pi(s)=\mathbb{E}_\pi[G_t\mid s_t=s],\qquad Q^\pi(s,a)=\mathbb{E}_\pi[G_t\mid s_t=s,a_t=a]. \tag{24.4}$$

由全期望公式，先对动作条件化，得到 $V^\pi(s)=\sum_a\pi(a\mid s)Q^\pi(s,a)$；再代入式 (24.3)，得到**贝尔曼方程**（Bellman Equation）

$$Q^\pi(s,a)=\bar r(s,a)+\gamma\sum_{s'}P(s'\mid s,a)V^\pi(s'), \tag{24.5}$$

$$V^\pi(s)=\sum_a\pi(a\mid s)\left[\bar r(s,a)+\gamma\sum_{s'}P(s'\mid s,a)V^\pi(s')\right], \tag{24.6}$$

$\bar r(s,a)$ 是即时奖励的条件期望. 终止状态价值为零；有限时域若未把时间并入状态，应显式写作 V_t^π.

24.2.2　策略评估及最优性

固定策略时，Bellman 方程是策略评估；最优价值把动作平均替换为最大化，得到

$$V^*(s)=\max_a\left[\bar r(s,a)+\gamma\sum_{s'}P(s'\mid s,a)V^*(s')\right]. \tag{24.7}$$

在有限状态、折扣情形下，固定策略算子满足

$$\|\mathcal{T}^\pi V-\mathcal{T}^\pi U\|_\infty\leqslant\gamma\|V-U\|_\infty, \tag{24.8}$$

因为转移概率非负且求和为一. $\gamma<1$ 时这说明其为压缩映射，存在唯一不动点. 有限终止任务取 $\gamma=1$ 时可后向递推，但不能照搬严格压缩证明；任意神经网络近似与离策略更新也不因此获得收敛保证.

24.2.3 优势表示相对收益

优势函数（Advantage Function）定义为

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s). \tag{24.9}$$

它衡量在同一状态下选择该动作相对于当前策略平均水平的增益，满足 $\sum_{a} \pi(a \mid s) A^{\pi}(s,a) = 0$. 优势为正不一定表示任务绝对成功；在整体很差的策略中，它也可能仅表示“较不差”.

24.3 两步决策过程

例 21.1

初始状态 s_0 选择左动作 L 或右动作 R，概率分别为 p 和 $1-p$，第一步奖励均为零. 选择 L 到达 s_L，第二步选择成功动作 S 的概率为 q，奖励 4；选择失败动作 F 的概率为 $1-q$，奖励 0. 选择 R 到达 s_R，第二步只有一个动作，奖励 2. 随后均终止，取 $\gamma = 1$.

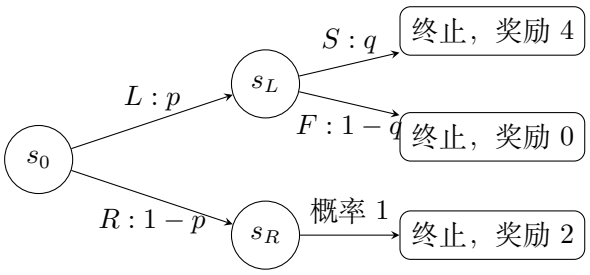


图 24.1: 两步 MDP. 所有第一步奖励为零，第二步奖励给出完整回报. 该构造用于精确计算价值与策略梯度.

由 Bellman 递推得到

$$V(s_L) = 4q, \quad V(s_R) = 2, \quad J = V(s_0) = 4pq + 2(1-p). \tag{24.10}$$

取 $p = \dfrac{1}{2}, q = \dfrac{3}{4}$，则 $V(s_L) = 3, V(s_R) = 2, V(s_0) = 2.5$. 初始状态的优势为 $A(s_0, L) = 0.5$、$A(s_0, R) = -0.5$；在 s_L，成功优势为 1，失败优势为 -3. 概率加权后分别为零.

三条可能轨迹概率与回报为 $P(LS) = \dfrac{3}{8}, G = 4$；$P(LF) = \dfrac{1}{8}, G = 0$；$P(R) = \dfrac{1}{2}, G = 2$. 平均回报为 2.5，二阶矩为 $\frac{3}{8} \times 16 + \frac{1}{2} \times 4 = 8$，所以方差为 $8 - 2.5^2 = 1.75$. 一次成功样本回报 4 并不等于策略价值 4；价值来自所有可能轨迹的期望.

令 $p = \sigma(\theta)$、$q = \sigma(\psi)$，则

$$\frac{\partial J}{\partial \theta} = p(1-p)(4q-2) = 0.25, \tag{24.11}$$

$$\frac{\partial J}{\partial \psi} = 4pq(1-q) = 0.375. \tag{24.12}$$

后文将用采样轨迹的对数概率重新得到同一梯度，说明策略梯度并不要求对动作选择本身求导.

24.4　策略梯度推导

24.4.1　对数导数技巧

目标为 $J(\theta)=\mathbb{E}_{\tau\sim P_\theta}[R(\tau)]$，其中 $R(\tau)=\sum_t \gamma^t r_t$. 由轨迹概率分解

$$P_\theta(\tau)=d_0(s_0)\prod_{t=0}^{H-1}\pi_\theta(a_t\mid s_t)P(s_{t+1},r_t\mid s_t,a_t). \tag{24.13}$$

环境项不含待优化参数，因此

$$\nabla_\theta \log P_\theta(\tau)=\sum_t \nabla_\theta \log \pi_\theta(a_t\mid s_t). \tag{24.14}$$

利用 $\nabla P = P\nabla \log P$，得到**得分函数估计**（Score-Function Estimation）

$$\nabla J=\sum_\tau R(\tau)\nabla P_\theta(\tau) \tag{24.15}$$

$$=\mathbb{E}_\pi\left[R(\tau)\sum_t \nabla \log \pi_\theta(a_t\mid s_t)\right]. \tag{24.16}$$

这一形式属于 REINFORCE 类方法的基础 [134]. 离散动作和外部奖励无需可微；梯度来自调整产生该轨迹的概率. 若奖励函数直接包含可训练参数并且目标要求对其求导，还会出现额外导数，不能继续使用固定奖励推导而省略它.

24.4.2　因果性消除过去奖励

给定动作前的历史，过去奖励已经确定，而

$$\mathbb{E}_{a_t\sim\pi(\cdot|s_t)}[\nabla \log \pi(a_t\mid s_t)]=\sum_a \nabla\pi(a\mid s_t)=\nabla 1=0. \tag{24.17}$$

所以当前位置得分函数与过去奖励乘积的期望为零，可将完整回报换为从当前动作开始的后续回报，得到

$$\nabla J=\mathbb{E}_\pi\left[\sum_{t=0}^{H-1}\gamma^t G_t\nabla \log \pi_\theta(a_t\mid s_t)\right]. \tag{24.18}$$

外面的 γ^t 来自目标对起点的折扣，G_t 内部则从当前时刻重新计量. 若取 $\gamma=1$，两者自然消失；若省去外部折扣，需要说明采用了怎样的状态采样或目标约定.

以 $Q^{\pi}(s_t,a_t)$ 替代随机 G_t 的条件期望，可写成策略梯度定理形式

$$\nabla J=\sum_{s} d_{\gamma}^{\pi}(s)\sum_{a}\pi(a\mid s)Q^{\pi}(s,a)\nabla\log\pi(a\mid s), \tag{24.19}$$

其中 $d_{\gamma}^{\pi}(s)=\sum_{t}\gamma^{t}P_{\pi}(s_t=s)$ 是未归一化的折扣访问量. 状态分布虽然依赖策略，却已经通过轨迹推导处理，不需要在该表达式外再对访问量重复求导.

24.4.3 策略梯度分析

例 21.2

初始动作得分为：选 L 时 $\dfrac{\partial\log p}{\partial\theta}=1-p=\dfrac{1}{2}$，选 R 时为 $-p=-\dfrac{1}{2}$. 故式 (24.16) 给出

$$\frac{3}{8}\times 4\times\frac{1}{2}+\frac{1}{8}\times 0\times\frac{1}{2}+\frac{1}{2}\times 2\times\left(-\frac{1}{2}\right)=0.25. \tag{24.20}$$

对于 ψ，仅访问 s_L 的轨迹贡献梯度：成功得分 $1-q=\dfrac{1}{4}$，失败得分 $-q=-\dfrac{3}{4}$，所以 $\frac{3}{8}\times 4\times\frac{1}{4}=0.375$. 结果与直接求导式 (24.12) 一致.

终点奖励会乘到整条轨迹各决策的得分函数上，因而能够给早期动作提供信号. 但一次成功不能证明每个动作都必要或最优；这种**信用分配**（Credit Assignment）依靠跨轨迹统计相关性完成，奖励稀疏时方差通常较大.

24.5 基线、优势及方差

24.5.1 状态基线的无偏性

令 $b(s)$ 与当前采样动作无关，则

$$\mathbb{E}_{a}[b(s)\nabla\log\pi(a\mid s)]=b(s)\nabla\sum_{a}\pi(a\mid s)=0. \tag{24.21}$$

因此式 (24.18) 中可把 G_t 替换为 $G_t-b(s_t)$，而不改变梯度期望. 选择 $b=V^{\pi}$ 时得到优势估计. 即使基线近似不精确，只要它在动作采样条件下满足独立性并按固定量使用，完整回报减基线的估计仍无偏；不准确基线可能降低方差效果.

若基线与策略共享参数，策略损失中应阻断通过基线的求导；否则会加入不属于上述估计的项. 用包含当前动作回报的样本均值作为基线，也不自动满足动作独立条件. 例如同一状态 K 个独立动作，以包括自身的平均回报为基线，平均得分估计的期望为原梯度的 $(1-\dfrac{1}{K})$ 倍；使用其余样本均值可在对应独立条件下避免这项偏差. 标准差归一化还会产生额外随机比例，不能仅凭“均值为零”证明无偏.

24.5.2　最小方差基线

设 $g=\nabla\log\pi(a\mid s)$，在固定状态下最小化 $\mathbb{E}[\|(G-b)g\|^2]$，对 b 求导可得

$$b^*(s)=\frac{\mathbb{E}[G\|g\|^2\mid s]}{\mathbb{E}[\|g\|^2\mid s]}. \tag{24.22}$$

当得分范数与回报无关或近似固定时，$V(s)$ 接近该选择. 这个公式说明“状态价值是最小方差基线”需要附加条件；工程上使用价值网络主要因为它可学习且能在不同状态间泛化.

在两步算例中，初始状态两动作得分范数相同，取基线 2.5.三条轨迹对 θ 的样本梯度为 $0.75,-1.25,0.25$，期望仍为 0.25. 二阶矩为 $\frac{3}{8}(0.75)^2+\frac{1}{8}(1.25)^2+\frac{1}{2}(0.25)^2=0.4375$，方差为 0.375；无基线时样本梯度为 2、0、-1，方差为 1.9375. 基线改变方差而保持期望，本例可以逐项验证.

24.6　蒙特卡洛、时序差分及广义优势估计

24.6.1　回报估计及自举

蒙特卡洛估计（Monte Carlo Estimation，MC）使用完整实际回报 G_t 拟合 $V_\phi(s_t)$. 它不依赖下一状态的价值预测，但需要等待后续奖励，且受到整段轨迹随机性的影响.

时序差分（Temporal Difference，TD）使用一步目标 $r_t+\gamma V_\phi(s_{t+1})$，残差为

$$\delta_t=r_t+\gamma V_\phi(s_{t+1})-V_\phi(s_t). \tag{24.23}$$

以现有预测构造目标称为**自举**（Bootstrapping）. 若 $V_\phi=V^\pi$，给定状态和动作后 $\mathbb{E}[\delta_t\mid s_t,a_t]=A^\pi(s_t,a_t)$；若价值近似有误，残差会同时包含奖励信息和价值误差.

n 步优势估计为

$$\widehat{A}_t^{(n)}=\sum_{k=0}^{n-1}\gamma^k r_{t+k}+\gamma^n V_\phi(s_{t+n})-V_\phi(s_t)=\sum_{k=0}^{n-1}\gamma^k\delta_{t+k}. \tag{24.24}$$

等号来自中间价值项逐项消去. 越长的回报使用更多实际奖励，越短的回报更多依赖价值估计.

24.6.2　广义优势估计的展开

广义优势估计（Generalized Advantage Estimation，GAE）用指数衰减组合 TD 残差 [135]：

$$\widehat{A}_t^{\mathrm{GAE}}=\sum_{l=0}^{H-1-t}(\gamma\lambda)^l\delta_{t+l},\qquad 0\leqslant\lambda\leqslant 1. \tag{24.25}$$

它可以由 n 步估计的几何加权得到：在无限展开中，残差 δ_{t+l} 出现在所有 $n > l$ 的项中，其权重之和为 $(1-\lambda)\sum_{n=l+1}^{\infty}\lambda^{n-1}\gamma^{l} = (\gamma\lambda)^{l}$. 有限轨迹需要把剩余权重放到最后可用回报上，并正确处理边界价值.

当 $\lambda = 0$ 时只保留一步 TD；当 $\lambda = 1$ 且轨迹真实终止、末状态价值为零时，得到 $G_t - V_\phi(s_t)$. 后一形式对策略梯度仍具有状态基线的无偏性质；较小 λ 在价值近似不准确时可能引入偏差，但通常减少远期随机性. 若使用精确 V^{π}，相应在策略条件下各 n 步估计的条件期望一致.

例 21.3

在成功轨迹 LS 上取估计值 $V_\phi(s_0) = 2.4, V_\phi(s_L) = 2.8$, 终止价值 $0, \gamma = 1, \lambda = 0.8$. 于是

$$\delta_0 = 0 + 2.8 - 2.4 = 0.4, \qquad \delta_1 = 4 - 2.8 = 1.2, \tag{24.26}$$

$$\widehat{A}_1 = 1.2, \qquad \widehat{A}_0 = 0.4 + 0.8 \times 1.2 = 1.36. \tag{24.27}$$

完整回报减基线在初始状态为 $4 - 2.4 = 1.6$. 差异来自 λ 对后续残差的衰减，而不是算术错误. 一次成功轨迹的估计也不应与精确平均优势 0.5 直接等同，因为后者对第二步随机动作取了期望.

24.6.3 终止及截断

真正终止后没有未来回报，价值设为零. 若只是采样批次或计算窗口结束，而任务还会继续，应在最后状态自举. 若达到输出预算被任务定义为失败终止，则属于另一种环境终止规则，不能沿用继续任务的自举解释.

实践中需要分别标识“允许从下一状态自举”和“允许沿当前缓冲继续传播残差”. 数据切片末尾可以保留下一状态价值，却不能跨到另一个样本的残差. 把这两类标记合并为一个含糊的 done，容易造成边界回报错误.

24.7 Actor–Critic 及策略分布

24.7.1 策略及价值的分工

行动者—评论家（Actor–Critic）用策略网络选择动作，用价值网络估计未来回报并降低更新方差. 策略更新最大化近似目标 $\mathbb{E}[\log \pi_\theta(a_t \mid s_t)\widehat{A}_t]$，其中采样动作与优势在本次求导中固定；价值网络可最小化

$$L_V(\phi) = \frac{1}{2}\mathbb{E}[(V_\phi(s_t) - \widehat{G}_t)^2], \tag{24.28}$$

并把回报目标视为固定量. 共享骨干可以节约计算，却也会引入策略与价值梯度的相互影响，需要明确权重与更新范围.

评论家并不判断一段文字是否“写得好”，而是估计当前策略下的未来回报．奖励器定义反馈，价值模型估计反馈期望，参考策略约束变化，三者不能混称为一个评分模型.

24.7.2　在策略及离策略

在策略学习（On-Policy Learning）主要使用当前待评估策略产生的数据；**离策略学习**（Off-Policy Learning）使用其他行为策略产生的数据．一次策略更新后，旧轨迹已经不完全来自新策略，多次复用必须处理这种偏移.

语言模型训练常同时存在当前策略 π_θ、采样旧策略 μ 与参考策略 π_{ref}．μ 决定数据从哪里来，π_{ref} 通常作为行为保持的比较基准．它们即使最初参数相同，也承担不同职责；不能用参考概率代替实际采样概率计算重要性比.

24.8　重要性采样及 PPO

24.8.1　整轨迹分布校正

假设行为策略对目标策略的轨迹支持集均为正，环境相同，则

$$W(\tau)=\frac{P_\theta(\tau)}{P_\mu(\tau)}=\prod_t \frac{\pi_\theta(a_t\mid s_t)}{\mu(a_t\mid s_t)}=\prod_t \rho_t, \tag{24.29}$$

$$\mathbb{E}_{\pi_\theta}[f(\tau)]=\mathbb{E}_\mu[W(\tau)f(\tau)]. \tag{24.30}$$

这是**重要性采样**（Importance Sampling，IS）的精确换测度关系．长序列中概率比连乘可能产生极端方差；某些每步仅略大的比率，累计后仍可能很大.

如果采样使用温度、词表截断或其他概率变换，行为概率应对应变换后的实际分布．尤其硬截断可能令目标策略仍有概率的动作在行为分布中为零，破坏精确校正所需支持条件．记录原始模型 softmax 并不能补回未被采样的支持集.

24.8.2　局部替代目标

PPO 采用旧策略状态与动作样本，构造单步概率比的**替代目标**（Surrogate Objective）[136]：

$$L_{\text{sur}}(\theta)=\mathbb{E}_{(s_t,a_t)\sim\mu}[\rho_t(\theta)\widehat{A}_t],\qquad \rho_t=\exp[\log\pi_\theta(a_t\mid s_t)-\log\mu(a_t\mid s_t)]. \tag{24.31}$$

本节为突出更新机制取 $\gamma=1$；折扣目标需要相应状态或时间权重．单步比率只校正旧状态下的动作分布，不能精确校正更新后整条状态访问分布．因此它是局部近似，不能把该式称为任意新策略回报的无偏估计.

24.8.3 裁剪目标的方向性

近端策略优化（Proximal Policy Optimization，PPO）的常见裁剪目标为

$$L_{\text{clip}} = \mathbb{E}_{\mu}\left[\min\{\rho_t \widehat{A}_t, \text{clip}(\rho_t, 1-\epsilon, 1+\epsilon)\widehat{A}_t\}\right]. \tag{24.32}$$

目标被最大化. 优势为正时，继续把概率比推到 $1+\epsilon$ 以上不再获得额外收益；优势为负时，把概率比降到 $1-\epsilon$ 以下不再获得额外收益. 朝错误方向变化则仍可能受到惩罚.

取 $\epsilon = 0.2$：当 $\widehat{A} = 2, \rho = 1.4$ 时两项为 2.8 和 2.4，选 2.4；当 $\widehat{A} = -2, \rho = 0.6$ 时两项为 $-1.2, -1.6$，选 -1.6. 但当 $\widehat{A} = -2, \rho = 1.4$，选的是 -2.8，不会把错误增大负优势动作概率的惩罚裁为 -2.4.

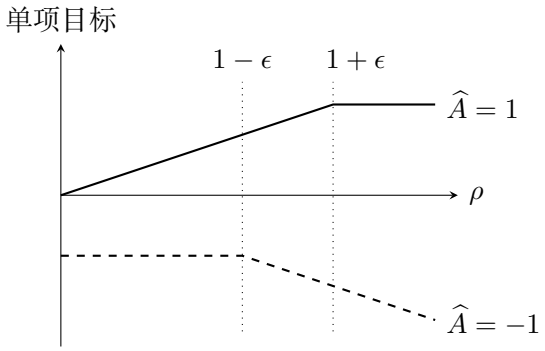


图 24.2: PPO 单样本裁剪目标的解析分段线. 裁剪限制有利方向上的继续增益，并不是把所有概率比强制限制在区间内.

裁剪不是对参数或全分布散度的硬约束，也不保证每个动作概率比都在区间内. 共享参数、其他样本梯度与多次更新仍会改变被裁剪样本. 因此应监控新旧策略差异、裁剪比例、熵与实际回报，并在偏移过大时停止复用旧批次. 较高裁剪比例说明许多样本进入局部饱和区域，不等于质量已经提升.

24.9 语言模型的奖励及长度目标

24.9.1 序列奖励及词元信用分配

若只有终点评分 $R(x, y)$，取 $\gamma = 1$，则一条回答中每个动作的后续回报均为同一终点奖励. 策略梯度为

$$\nabla J = \mathbb{E}_{x,y}\left[R(x,y)\sum_{t=1}^{T(y)} \nabla \log \pi_\theta(y_t \mid x, y_{<t})\right]. \tag{24.33}$$

这与把示范所有词元一律提高概率不同：回报或优势的符号可以使采样回答概率降低. 奖励稀疏时需要足够不同轨迹才有区分信号；同一问题所有回答同分，往往无法提供有效的相对改进证据.

结果评分、步骤评分和成本惩罚表达不同目标. 把程序状态验证结果作为反馈，不意味着自然语言解释就是模型内部计算的完整记录. 奖励定义与验证器正确性将在后续章节展开；本章推导只假定给定了固定反馈机制.

24.9.2　长度归一化效应

式 (24.33) 对一条序列的所有动作得分求和. 若改成每条序列除以自身随机长度 $T(y)$，并使用终点奖励而无基线，则得到

$$\mathbb{E}\left[\frac{R(x,y)}{T(y)}\sum_t \nabla \log \pi_\theta(y_t \mid x, y_{<t})\right] = \nabla \mathbb{E}\left[\frac{R(x,y)}{T(y)}\right], \tag{24.34}$$

即优化单位长度奖励，而不是原始期望奖励. 这里长度是采样轨迹的函数，不能当作所有样本共用常数. 再加入依赖状态的基线时，若将基线也除以未来随机长度，原来的动作独立无偏证明不一定成立.

例如两个可选完整回答长度为 1 和 4、奖励均为 1，原始目标对它们无偏好；$\frac{R}{T}$ 目标明显偏向短回答. 按整批总词元归约与按每条回答长度归约又是不同权重：前者在一个冻结批次中对全部项施加共同尺度，后者改变序列之间的相对贡献. 比较语言模型策略算法时，必须同时写出提示、序列和词元三个层级的归约.

24.9.3　停止、预算及惩罚

若 $\gamma < 1$ 且只在终点给奖励，则起点回报为 $\gamma^{T-1}R$，隐含偏向更早取得相同奖励. 显式每步成本 c 则给出 $R - cT$ 一类目标，语义与折扣不同. 结束动作是否正确、达到最大长度如何评分、无答案或工具失败怎样终止，都属于环境目标而非日志选项.

超长输出可能来自奖励偏好、停止协议或探索分布. 仅缩短最大长度可以限制成本，却不必然训练出更有效的策略；反过来，无限制提高生成预算也可能增加无效搜索和验证负担. 应同时记录成功率、输出长度、终止原因和单位成功成本.

24.10　采样、估计及更新的数据流

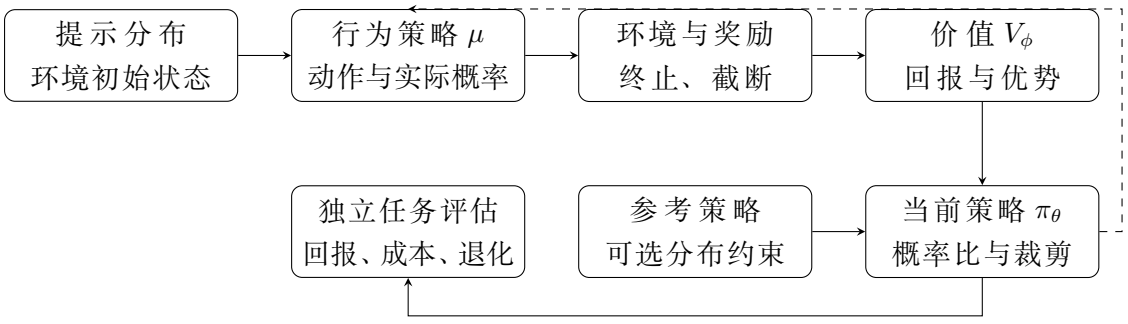


图 24.3: 策略训练的数据流．行为策略产生数据，价值模型估计回报，当前策略执行更新，参考策略提供可选约束．更新后须明确何时刷新采样策略．

算法 24.1　有限轨迹上的 GAE 与 PPO 更新

输入为提示分布、初始策略与价值网络、奖励规则、采样预算及裁剪参数；输出为新策略和可追溯训练状态．本算法取有限终止任务、$\gamma = 1$，一般折扣情形须补相应时间权重．

1. 冻结行为策略快照 μ，按明确采样分布生成轨迹，记录动作、行为对数概率、奖励及真实终止与截断标记．
2. 固定本批次价值预测；在真实终止处置尾值为零，在未终止切片末端保留自举值．
3. 从后向前计算 δ_t 与 $\widehat{A}_t = \delta_t + \lambda \widehat{A}_{t+1}$；在缓冲或轨迹边界停止传播．以采样时冻结的旧价值预测构造 $\widehat{G}_t = \mathrm{stopgrad}(\widehat{A}_t + V_{\mathrm{old}}(s_t))$，作为价值回归的有限长度 λ 回报目标；本轮优化期间不随当前价值网络重算此目标．
4. 在有限轮次内计算新策略对同一动作的概率比，最大化式 (24.32)，并按独立价值目标更新评论家．行为概率、优势与目标在策略求导中固定．
5. 监控新旧策略差异、裁剪、价值误差、终止与奖励分量；偏移超出预定范围时停止复用当前批次．
6. 刷新行为策略并重新采样，直到达到交互或计算预算；用独立任务指标评估，保存模型、优化状态、采样和奖励版本．

长度为 H 的 GAE 递推时间与额外存储均可为 $O(H)$；主要计算由策略、价值及环境调用决定．动作概率必须来自对应的实际行为策略，不得用更新后的概率覆盖历史记录．

估计正确不等于奖励正确

策略梯度无偏性讨论的是对给定期望目标的估计．若奖励遗漏任务要求、验证器被利用或提示分布偏离部署，即使梯度计算完全正确，也可能强化错误行为．算法、反馈和独立任务评估必须分别说明．

24.11　奖励及验证边界

采样策略、参考概率、裁剪和优势应统一到决策过程与概率比中解释；组相对目标由第二十六章《可验证推理训练》展开．终点奖励、步骤证据和轨迹预算承担不同职责，验证器设计不等同于价值估计．固定少量奖励只能展示目标计算关系，不能证

明策略经过真实交互已经改善.

第二十五章　人类反馈学习

监督微调要求示范者给出应当生成的回答，偏好数据则允许示范者在多个候选之间作比较．这种差异使反馈更容易针对语气、完整性或任务适配程度，但也引入了新的统计问题：一次比较究竟代表谁的偏好，候选从哪里产生，以及局部排序能否外推到训练后出现的新回答．

本章先把比较反馈写成概率模型，再推导奖励正则化目标的最优策略，并由此得到直接偏好优化．在线强化学习的策略梯度基础见第二十四章《强化学习基础》；可验证奖励和推理训练在第二十六章《可验证推理训练》进一步展开．这里的“对齐”指相对于所声明反馈分布调整行为，不意味着已经证明模型满足所有使用者的全部价值要求．

25.1　人类反馈及偏好数据

25.1.1　比较单位及数据分布

一条偏好记录可表示为 (x, y_a, y_b, z)，其中 x 为共同上下文，y_a, y_b 为两个候选，$z \in \{0, 1\}$ 指示是否偏好 y_a．上下文包括角色、约束和必要历史；若两条候选实际上回答了不同问题，就不能用同一提示下的偏好模型解释它们．

偏好对（Preference Pair）的生成机制至少包含提示分布、候选策略、解码配置和标注规则．设提示来自 μ，候选来自条件分布 $q(y_a, y_b \mid x)$，标注者来自群体分布 h，则训练风险的期望同时依赖这三者．只扩大比较数量，不扩大提示和候选覆盖，可能只是反复学习同一狭窄区域．

候选显示次序、长度、格式、模型身份泄漏和标注者知识都会影响比较．应保存可重建的原始候选、匿名化呈现条件和标注版本．多个候选产生的成对组合共享同一提示和回答，并非独立观测；划分训练与评估集合时应以提示或更高层来源分组．

25.1.2　符号及假设

本章主要符号见表 25.1.

表 25.1: 偏好学习的主要符号.

符号	含义
x, y, T_y	提示、回答序列和回答所计词元数，终止标记是否计入须固定.
y_w, y_l	标注中相对优选与相对拒选的回答.
$r_\phi(x, y)$	参数为 ϕ 的标量奖励函数.
$\pi_\theta, \pi_{\text{ref}}, \pi_{\text{old}}$	当前策略、固定参考策略及产生一批在线样本的旧策略.
β	KL 正则化系数；DPO 推导沿用同一尺度.
$\sigma(u)$	Logistic 函数 $\dfrac{1}{1+e^{-u}}$.
$Z(x)$	固定提示下的归一化配分函数.
Δ_θ	优选与拒选回答的相对参考对数概率差.

偏好与策略推导的范围

先假设固定提示下回答集合有限，参考策略在该集合上为正，奖励有限且 $\beta > 0$. 偏好用一个标量效用差的 Logistic 概率描述. 对可数无限回答集合，还需保证配分函数有限及相关期望存在. 真实群体可能有冲突、平局和上下文依赖，单标量模型只是一种近似.

25.2　奖励建模

25.2.1　Bradley–Terry 形式

将每个候选的正强度写为 $u_a = e^{r_a}$ 和 $u_b = e^{r_b}$，以相对强度定义胜出概率，得到 Bradley–Terry 形式 [137]

$$p_\phi(y_a \succ y_b \mid x) = \frac{e^{r_a}}{e^{r_a} + e^{r_b}} = \sigma(r_a - r_b), \qquad r_a = r_\phi(x, y_a). \tag{25.1}$$

对二元标签采用负对数似然，单对损失为

$$\ell_{\text{RM}} = -z \log \sigma(d) - (1 - z) \log \sigma(-d), \qquad d = r_a - r_b. \tag{25.2}$$

其间隔导数为 $\dfrac{\partial \ell}{\partial d} = \sigma(d) - z$；二阶导数为 $\sigma(d)(1 - \sigma(d)) \geqslant 0$. 损失对间隔凸，不代表对深度奖励网络参数凸. 标注偏好 a 却预测 $\sigma(d)$ 很小时，梯度会推动 $r_a - r_b$ 增大.

如果同一对有多个独立判断，z 可替换为偏好 a 的比例，表示二项似然按比较数归一化后的目标. 把一个平局直接写为 $z = \dfrac{1}{2}$，相当于要求模型对两者给出相等偏好概率；这未必等同于有显式平局概率的三分类机制，应根据标注含义选择.

25.2.2 奖励的可辨识性

对同一提示给所有奖励加上任意 $c(x)$，式 (25.1) 不变．因此成对比较无法识别提示相关的绝对零点．若数据只连接一部分回答子集，不同连通分量之间的相对偏移也无法由这些比较确定．

尺度则不同：将差值乘常数会改变偏好概率．若同时引入未知噪声温度 τ，模型只由 $\dfrac{r_a - r_b}{\tau}$ 决定，此时奖励尺度和温度不能分别识别．奖励归一化或温度修改会改变后续策略优化的有效强度，不能视为无关的显示调整．

标量效用还不能表示确定性的循环偏好：若 $a \succ b, b \succ c, c \succ a$ 均被要求以大于 $\dfrac{1}{2}$ 的概率成立，则分别需要 $r_a > r_b > r_c > r_a$，产生矛盾．出现系统性冲突时，应检查群体、任务和呈现条件，不能只增加训练轮数．

25.2.3 奖励网络及校准

奖励模型可以用编码器或因果模型提取 (x, y) 的表示，再经标量读出头输出 r_ϕ．若使用回答末端表示，须取实际有效末端，而非填充位置；若截断删除了结论或约束，原来的偏好标签可能不再适用．模型结构并不要求与策略完全相同，但输入语义必须兼容．

排序准确率关注差值符号，概率校准关注预测胜率与观察频率是否一致．二者不能替代．即使在离线比较集上校准良好，策略优化后产生的新回答也可能偏离原候选分布．对奖励模型进行优化会主动寻找其高分区域，因此评估需要覆盖被优化后的候选，而不只是原始数据．

构造一个奖励算例：$r_w = 1.2, r_l = 0.2$，间隔为 1，预测胜率 $\sigma(1) \approx 0.7311$，优选标签损失约 0.3133．若在相同条件下只有 55% 的人选择 w，高胜率预测并不由单次排序正确所支持．换成均为 0.7 的奖励，胜率为 0.5，无法表达排序但不虚构差距．

25.3 基于人类反馈的强化学习

25.3.1 目标及 KL 方向

基于人类反馈的强化学习（Reinforcement Learning from Human Feedback，RLHF）可通过比较数据拟合奖励，再优化策略；InstructGPT 是这种训练链路的代表性工作 [12]．固定提示下的一种目标为

$$J_x(\pi) = \sum_y \pi(y \mid x) r(x, y) - \beta \sum_y \pi(y \mid x) \log \frac{\pi(y \mid x)}{\pi_{\text{ref}}(y \mid x)}. \tag{25.3}$$

KL 的方向为当前策略到参考策略．它使参考零概率事件具有无穷偏离代价，所以支持集条件是推导的一部分．参考约束限制分布漂移，却不自动阻止奖励漏洞、事实错误或某一能力退化．

25.3.2　最优策略的推导

对归一化约束 $\sum_y \pi_y = 1$ 引入乘子 λ，记 $q_y = \pi_{\text{ref}}(y \mid x)$，拉格朗日函数为

$$\mathcal{F} = \sum_y \pi_y r_y - \beta \sum_y \pi_y \log(\frac{\pi_y}{q_y}) + \lambda(\sum_y \pi_y - 1). \tag{25.4}$$

对正概率内部点求导得

$$r_y - \beta[\log(\frac{\pi_y}{q_y}) + 1] + \lambda = 0. \tag{25.5}$$

整理可得 $\pi_y = q_y \exp(\frac{r_y}{\beta}) \exp(\frac{\lambda}{\beta} - 1)$. 最后一项不依赖回答，通过归一化确定为 $\frac{1}{Z(x)}$，因此

$$\pi^*(y \mid x) = \frac{\pi_{\text{ref}}(y \mid x) e^{\frac{r(x,y)}{\beta}}}{Z(x)}, \qquad Z(x) = \sum_y \pi_{\text{ref}}(y \mid x) e^{\frac{r(x,y)}{\beta}}. \tag{25.6}$$

不仅可以检查驻点，还可以直接证明全局最优. 把最优策略的对数表达代回式 (25.3)，得到

$$J_x(\pi) = \beta \log Z(x) - \beta D_{\text{KL}}(\pi \| \pi^*). \tag{25.7}$$

KL 非负，且仅在两个分布相等时为零，所以 π^* 是支持集内的唯一最优分布. 这个证明同时给出任意策略距离最优目标值的差额.

当固定奖励且 β 增大时，指数倾斜减弱；当 $\beta \to \infty$ 时趋近参考策略. $\beta \to 0^+$ 时质量集中到最高奖励集合，并在并列最优集合中保留参考权重比例. 奖励与 β 同时乘同一个正数不改变式 (25.6)，再次说明奖励尺度与约束强度必须一起解释.

例 22.1

设回答集合仅有 a, b，参考概率为 $(0.8, 0.2)$，奖励为 $(0, \log 4)$，$\beta = 1$. 未归一化质量为 $(0.8, 0.8)$，$Z = 1.6$，故最优策略为 $(0.5, 0.5)$.

最优期望奖励为 $\frac{1}{2}\log 4 = \log 2 \approx 0.6931$. KL 为

$$\frac{1}{2}\log\frac{0.5}{0.8} + \frac{1}{2}\log\frac{0.5}{0.2} = \frac{1}{2}\log 1.5625 \approx 0.2231.$$

目标值为 0.4700，等于 $\log 1.6$. 虽然 b 的奖励较高，最优策略并未以概率 1 生成它；参考约束保留了概率分配的代价. 若将两奖励同时增加 10，最优策略不变，而目标值增加 10.

25.4　直接偏好优化

25.4.1　由最优策略表示奖励

式 (25.6) 等价于

$$r(x, y) = \beta \log \frac{\pi^*(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x). \tag{25.8}$$

同一提示下，两回答奖励相减时，配分函数项消失. 将策略参数化为 π_θ，用相对参考的对数概率差表示偏好间隔，就得到**直接偏好优化**（Direct Preference Optimization，DPO）[138]：

$$\Delta_\theta = \log \pi_\theta(y_w \mid x) - \log \pi_\theta(y_l \mid x) - \log \pi_{\text{ref}}(y_w \mid x) + \log \pi_{\text{ref}}(y_l \mid x), \tag{25.9}$$

$$\ell_{\text{DPO}} = -\log \sigma(\beta \Delta_\theta) = \text{softplus}(-\beta \Delta_\theta). \tag{25.10}$$

这将显式奖励模型的拟合转成策略上的偏好似然. 有限数据、有限模型容量和有限优化并不保证实际策略达到式 (25.6) 中的理想解，不能把代数表示等同于训练后的全局最优性.

25.4.2 偏好梯度及概率变化

设 $u = \beta \Delta_\theta$，则

$$\nabla_\theta \ell_{\text{DPO}} = -\beta \sigma(-u)[\nabla_\theta \log \pi_\theta(y_w \mid x) - \nabla_\theta \log \pi_\theta(y_l \mid x)]. \tag{25.11}$$

参考概率固定，不参与求导. 排序错误且间隔很负时，系数趋近 $-\beta$；间隔很正时，梯度趋近零. β 同时改变间隔缩放和饱和程度，所以不能简单称“越大更新越强”，更不能称“越大偏好越可靠”.

DPO 优化相对比值，不保证优选回答的绝对概率每一步都增加. 例如参考在三回答 (w, l, o) 上为 $(0.4, 0.4, 0.2)$，某策略变为 $(0.3, 0.1, 0.6)$. 优选概率从 0.4 降到 0.3，但 $\dfrac{w}{l}$ 比从 1 升到 3，因而 $\Delta = \log 3 > 0$. 这说明需要同时监控两条回答的概率和自由生成行为，而不能只报告间隔.

取 $\beta = 0.5$，上述间隔给出 $u = 0.549306$，偏好概率为 $\dfrac{\sqrt{3}}{1+\sqrt{3}} \approx 0.6340$，损失约 0.4557，低于初始相等策略时的 $\log 2 \approx 0.6931$. 损失下降并不消除第三个回答质量未知的问题.

25.4.3 序列概率及长度

回答序列的条件概率必须遵循相同的停止协议：

$$\log \pi_\theta(y \mid x) = \sum_{t=1}^{T_y} \log \pi_\theta(y_t \mid x, y_{<t}). \tag{25.12}$$

若把求和改成除以 T_y 的平均，所得量一般不是原回答分布的对数概率，不能继续声称与式 (25.3) 推导相同. 长度归一化可以定义新的目标，但须重新说明其数学含义.

长度偏差可能同时来自标注偏好、候选生成和似然累积. 参考比值抵消部分基线倾向，却不保证消除全部偏差. 若截断只保留两回答共同前缀，原本可辨别的比较可能变成无信号；若仅一条被截断，标签可能转而反映截断质量. 共同提示应保持一致，优选与拒选的完整目标跨度应分别校验.

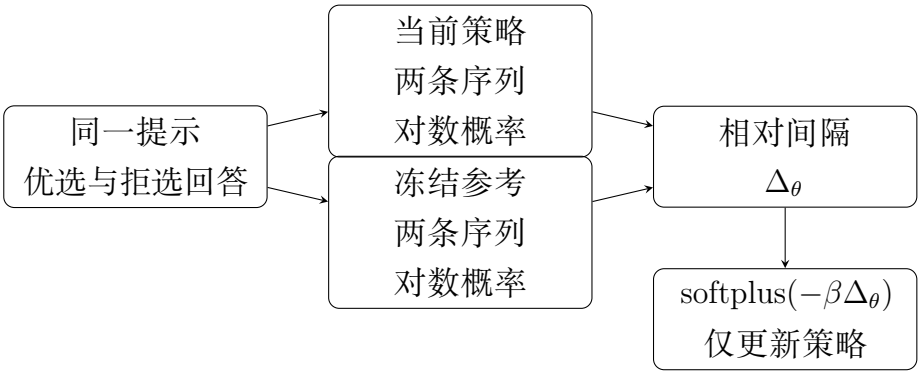


图 25.1: DPO 的数据与梯度边界．参考分支提供固定概率基线，不能随策略更新漂移.

25.5　在线 RLHF 的模型分工

参考策略（Reference Policy）规定相对偏离的基准；**旧策略**（Behavior Policy）在本节指产生当前批次轨迹的行为策略．前者可以跨多次更新保持固定，后者随采样轮次变化．策略概率比用于控制复用样本时的更新，参考 KL 用于约束行为漂移，两者职责不同.

典型 Actor–Critic 链路中，策略生成回答，奖励模型给出序列评分，价值模型估计前缀状态的未来回报，参考模型提供 KL 基线．价值估计不是回答质量评分：它依赖当前策略和剩余过程，主要用于构造优势及降低估计方差．奖励模型可以在策略优化阶段冻结，但其输入分布会随策略变化.

序列 KL 可以用链式法则写成策略访问前缀下的条件 KL 之和：

$$D_{\mathrm{KL}}(\pi_\theta(y \mid x)\|\pi_{\mathrm{ref}}(y \mid x)) = \mathbb{E}_{y\sim\pi_\theta} \sum_t \log \frac{\pi_\theta(y_t \mid x, y_{<t})}{\pi_{\mathrm{ref}}(y_t \mid x, y_{<t})}. \tag{25.13}$$

单个采样词元的对数比可以为负；只有在相应分布下取期望才得到非负 KL．若轨迹来自旧策略，直接平均这些值估计的是旧分布下的对数比，除非作出近似或正确分布校正，不能将其标为当前策略的精确 KL.

将终局奖励与逐词元 KL 惩罚结合，可构造供第二十四章《强化学习基础》策略优化算法使用的回报．对数比中的当前策略参数也影响采样分布；实际代理目标通常在旧轨迹上冻结部分统计量并限制更新幅度，应明确其为近似优化过程，而非把采样损失视作式 (25.3) 的逐项精确梯度.

算法 25.1　离线 DPO 的一次参数更新

输入：偏好对批次、固定参考、策略、$\beta > 0$、版本化模板与分词器．**输出：**更新后的策略及目标统计.

1. 校验每对共享同一完整提示、标签有明确含义，两回答的目标跨度与终止协议有效；剔除或单独记录截断后失去偏好依据的记录.

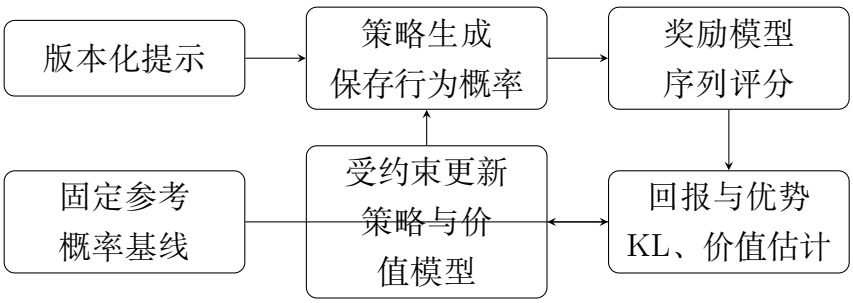


图 25.2: 在线奖励优化的职责．参考、奖励、价值与策略是逻辑角色，不要求部署为四个完全独立的服务．

2. 在冻结参考上计算两条回答的目标词元对数概率和．若使用缓存，键必须绑定参考权重、模板、分词器及精确词元序列．

3. 在当前策略上计算相同目标位置的对数概率和，依式 (25.10) 形成间隔与稳定的 Softplus 损失．

4. 按事先声明的提示或偏好对权重聚合，反向仅更新策略；累积与并行归约保持该权重口径，而非擅自改成词元平均．

5. 记录损失、间隔、两回答对数概率、长度及有效对数；定期在独立提示上自由生成，并与冻结基线进行盲比较．

参考概率缓存只有在输入与参考完全固定时才有效．使用共享基座与适配器实现参考时，关闭某个适配器不一定得到预期参考版本；还需确认基座未被更新、适配器组合一致、随机层处于确定的评分模式．

算法 25.2　奖励模型与在线策略的分阶段更新

输入：比较数据、提示分布、初始策略及独立评估集；**输出：**带来源与评估记录的策略版本．

1. 按提示和来源划分比较数据，拟合奖励模型，并评估排序、校准及关键切片的误差．

2. 固定本轮奖励和参考版本，从已标识的行为策略采样，保存回答、行为概率、终止原因和各奖励分量．

3. 依据声明的回报、优势和约束算法更新策略；限制样本复用范围，记录概率比、KL 估计口径和裁剪比例．

4. 在独立标准下检查质量、长度、拒答和原有能力．若奖励增加而独立质量下降，停止扩大优化强度并分析候选分布漂移．

5. 若重新收集偏好或更新奖励模型，建立新版本与新的训练阶段；保留旧版本以便解释行为变化与恢复.

25.6　GRPO 及方法选择

组相对策略优化（Group Relative Policy Optimization，GRPO）为同一提示生成 G 条回答，以组内奖励构造相对优势 [139]. 选择总体标准差约定时，

$$\bar{r} = \frac{1}{G}\sum_i r_i, \qquad s = \sqrt{\frac{1}{G}\sum_i (r_i - \bar{r})^2}, \qquad A_i = \frac{r_i - \bar{r}}{s + \epsilon}. \tag{25.14}$$

各优势之和为零；奖励全部相同时，相对优势全部为零. 组内标准化消除了绝对偏移，却同时重加权不同问题，不能将不同组的优势直接比较为绝对质量.

例如奖励 $(0, 1, 1, 2)$ 的均值为 1，标准差为 $\sqrt{\frac{1}{2}}$；忽略数值稳定项时优势为 $(-\sqrt{2}, 0, 0, \sqrt{2})$. 换成 $(10, 10, 10, 10)$，尽管绝对奖励更高，组内没有排序信号. 若奖励函数只识别格式，组内训练也只会提供格式差异. 组内均值包含自身样本所带来的基线偏差、零方差组、长度归约和裁剪目标，在第二十六章《可验证推理训练》中完整推导.

表 25.2: 反馈学习方法的监督对象、计算角色与适用边界.

方法	反馈对象	主要计算角色	关键边界
SFT	高质量示范回答	策略前后向	模仿示范；数据覆盖限制可学行为.
DPO	离线成对偏好	策略与固定参考评分	无需每次更新在线生成，但受候选覆盖和偏好模型限制.
奖励模型加策略优化	比较数据及在线奖励	策略、奖励、参考；可有价值模型	采样与评分代价较高，奖励外推可能被利用.
GRPO	同提示成组奖励	策略、评分器、可选参考约束	省去独立价值模型，但增加成组生成且可能无相对信号.

表 25.2 中的方法不是必须依次执行的固定流水线. 是否需要示范微调取决于起始模型是否已有任务行为；是否需要在线采样取决于离线候选覆盖和评分可靠性. 参数高效微调改变更新子空间，可与上述多种目标组合，并不构成另一类反馈来源.

25.7　奖励偏差及评估边界

奖励投机（Reward Hacking）指策略利用评分机制的缺陷提高分数，却没有改善真正的任务目标. 若代理奖励为 $\widehat{r} = r + e$，其中 r 是目标效用、e 是评分误差，则优化 $\widehat{r}$ 可以同时增加 e. 原分布下误差均值很小，不保证经过策略选择后的误差仍小.

偏好评估应同时固定候选生成预算、比较基线和呈现方式. 胜率需要报告比较数、平局处理、提示分组及不确定性；重复比较同一提示不能简单当作独立扩大样本. 长度、语言、任务类别和高严重度失败应分开观察，平均胜率不能替代各项行为边界.

对齐结论必须绑定反馈范围

奖励模型准确率、DPO 间隔、在线奖励和独立任务质量分别测量不同对象. 只有在明确提示分布、评价标准、生成条件和参考基线后，才能说明某项行为得到改善. 损失下降和奖励上升都是训练证据，不能单独证明广泛能力或安全性的提升.

第二十六章　可验证推理训练

一个回答是否正确、生成它的步骤是否可靠，以及训练能否增加这种回答的概率，是三个需要分别回答的问题．最终数值正确可能来自偶然抵消；流畅的推导可能包含错误前提；奖励上升也可能只是策略找到了验证器的漏洞．可验证推理训练的目标，是将任务定义、反馈证据与策略更新连接起来，并明确这种连接能够支持多强的结论．

第二十四章《强化学习基础》已经解释回报、策略梯度与信用分配，第二十五章《人类反馈学习》讨论学习型奖励和离线偏好．本章集中研究程序或环境能够核验的反馈，并以组相对策略优化为例，推导从候选生成到参数更新的具体计算．训练改变模型参数，测试时计算改变一次求解投入的资源；二者可以结合，但评估时必须分别记录．

26.1　可验证任务及监督对象

26.1.1　可验证对象

可验证奖励强化学习（Reinforcement Learning with Verifiable Rewards，RLVR）利用答案检查、程序测试、形式化证明检查或环境状态等产生奖励．可验证性始终相对于任务规范和检查器成立．程序通过有限测试，表示满足这些测试，而不自动证明对所有输入正确；一个形式证明被内核接受，仍以形式化命题准确表达原问题、所用公理与实现可信为前提．

设问题为 x，模型输出为 $y=(z,a)$，其中 z 是生成的中间步骤文本或结构化轨迹，a 是最终答案．验证器可写为

$$V_\nu(x,y,e)\longrightarrow(v,\mathcal{E}),\tag{26.1}$$

其中 ν 标记解析器与验证规则版本，e 是环境状态，v 是判定，$\mathcal{E}$ 是可复核证据．判定至少应能区分通过、不通过和无法判定；超时、解析失败与逻辑错误是不同事件，即使训练中最终映射成相同标量，也应分别保存原因．

验证器（Verifier）可以是确定性规则，也可以是具有误差的学习模型．确定性只意味着在相同输入与环境下行为可复现，不意味着规则覆盖完备．数学答案检查需要定义等价关系、定义域、单位和数值容差；代码检查需要约束执行环境、测试输入、

超时和副作用；工具任务需要检查真实状态转移，而非仅检查模型声称“操作成功”的文字.

本章策略推导

对固定问题 x，回答由有限长度的离散词元组成，结束符及截断规则固定. 除特别说明外，同组候选在给定 x 后从同一冻结行为策略独立同分布采样. 验证奖励不对模型参数直接求导. 重要性采样要求目标策略在行为策略支持集之外没有待估计的概率质量；涉及 KL 时，参考分布在目标分布支持集上为正.

符号	含义与形状
x, y_i, T_i	问题、第 i 条回答及其有效生成词元数.
$G, B, V, T_{\max}$	每题候选数、问题批量、词表大小、填充后最大长度.
h_{it}	回答第 t 个生成词元之前的可见历史 $(x, y_{i,<t})$.
$\pi_\theta, \pi_{\text{old}}, \pi_{\text{ref}}$	当前策略、冻结行为策略和冻结参考策略.
$r_i, \bar{r}, s_r, A_i$	标量奖励、组均值、组标准差与组内相对优势.
ρ_{it}, w_i	词元条件概率比与完整序列概率比.
M_{bit}	生成位置掩码，形状 $B \times G \times T_{\max}$；有效位置为 1.
ℓ_{bit}	已选词元对数概率，与掩码同形；完整 logits 为 $B \times G \times T_{\max} \times V$.
$\delta, \beta, \varepsilon$	裁剪宽度、KL 系数与标准差分母稳定项.

提示和工具观察可以影响后续生成，但通常不作为模型采样动作计入策略损失. 结束符是否计入、超长回答如何截断、分布式批量如何归约，都属于目标定义；填充长度不得进入有效词元分母.

26.1.2 结果及过程监督

结果监督（Outcome Supervision）根据最终答案或整条完成的结果给出反馈，**过程监督**（Process Supervision）在中间步骤提供更细粒度标签. **结果奖励模型**（Outcome Reward Model，ORM）与**过程奖励模型**（Process Reward Model，PRM）分别学习这两类信号. 过程监督研究明确区分了最终结果标签与步骤标签，不能用一种标签的性能结论替代另一种. [140]

设结构化任务为“从 7 开始，加 5，再乘 2”. 可靠执行给出状态 $7 \to 12 \to 24$. 另一候选先错误写成 $7 + 5 = 11$，再写出答案 24. 只核对终点时两者都可能得分 1；逐步复算时第二条的首步已失败. 相反，某条过程完全有效但最终答案被格式解析器误读，也可能遭到错误拒绝. 需要分别观察任务正确性和解析器行为.

过程奖励用于强化学习时，还需要把步骤标签映射到可归因的动作. 设轨迹分为 J 个步骤，第 j 步结束后得到奖励 $r_j^{(p)}$，则一种折扣回报为

$$R_j = \sum_{u=j}^{J} \gamma^{u-j} r_u^{(p)}, \qquad 0 < \gamma \leqslant 1. \tag{26.2}$$

第 j 步产生的动作可以影响该步及后续结果，却不能改变已经发生的前序奖励. 在适当因果和基线条件下，使用后续回报有助于去掉与当前动作无关的历史噪声. 将 R_j 分配给该步骤的生成词元时，还需声明步骤边界及其长度权重. 把每个步骤的即时分数直接当作所有先前词元的优势，与把未来步骤分数累加为回报，并不是同一信用分配方案. 过程标签本身也不是优势：它未扣除给定历史下的预期结果，不能仅通过改名获得价值函数解释.

过程标签可以判断局部算术是否合法、推理是否由前提推出，或者该前缀是否仍可完成任务. 这些标签不是同一对象. 特别是，局部合法步骤可能把搜索带入死路，而一个已包含错误的前缀之后也可能出现局部正确运算. 标注规范必须说明是“该步局部有效”，还是“截至该步整体有效”.

若步骤 j 的监督标签为 $u_j \in \{0,1\}$，预测有效概率为 $q_\phi(u_j = 1 \mid x, z_{\leqslant j})$，可用二元交叉熵训练

$$\mathcal{L}_{\mathrm{PRM}} = -\frac{1}{N_s} \sum_j [u_j \log q_\phi + (1 - u_j) \log(1 - q_\phi)], \tag{26.3}$$

其中 N_s 为有确定标签的步骤数；不确定标签应按既定规则屏蔽或单独建模. 将各步概率相乘作为整条轨迹有效概率，需要相应条件概率解释；把独立训练的局部评分直接相乘，通常只是聚合启发式，且会引入长度效应. 取最小值、平均值或累积和也各自改变选择准则.

稀疏奖励（Sparse Reward）常在回答结束后才出现. 当所有生成词元共享终点奖励时，信用分配难以区分必要推导、无关叙述与错误步骤. 过程反馈可以提高时间分辨率，但错误的密集反馈同样可能更快地强化不良行为. **思维链**（Chain of Thought，CoT）是生成文本的组织方式，不是独立网络层，也不存在由这个名称唯一确定的损失函数. 输出文本不能等同于模型内部计算的完整记录. [1]

26.2　拒绝采样及推理数据

26.2.1　筛选分布偏移

拒绝采样微调（Rejection Sampling Fine-Tuning，RFT）先生成候选，再保留满足接受条件的回答用于监督训练. 这里讨论按验证事件筛选的条件分布，不假定它等同于任意目标分布的通用拒绝采样算法.

令 $a(x, y) \in \{0, 1\}$ 为接受判定，候选分布为 $\pi(y \mid x)$，接受率为 $Z_a(x) = \sum_y \pi(y \mid x) a(x, y)$. 当 $Z_a(x) > 0$ 时，贝叶斯公式给出

$$\pi_{\mathrm{acc}}(y \mid x) = \frac{\pi(y \mid x) a(x, y)}{Z_a(x)}. \tag{26.4}$$

[1] 本章的“轨迹”在纯文本场景指可见生成序列，在工具场景还包含动作与观察. 公开可复算的证明或执行证据可以支持审计，但自然语言中间说明本身不能保证忠实反映内部计算.

因而筛选数据只在原生成分布支持的候选中重新分配概率. 策略从不生成的正确方法，不能仅靠这一次筛选出现. 每个成功样本所需候选数的期望为 $\frac{1}{Z_a(x)}$，低接受率问题会消耗更多生成与验证预算.

验证器准确性还与基线正确率共同决定保留数据质量. 设候选正确概率为 p，正确样本通过概率为 a，错误样本通过概率为 b，则

$$\Pr(\text{correct} \mid \text{pass}) = \frac{pa}{pa + (1-p)b}. \tag{26.5}$$

例如 $p = 0.1, a = 0.9, b = 0.05$，保留数据正确率仅为 $\frac{0.09}{0.09 + 0.045} = \frac{2}{3}$. 即使误通过率看似不大，低基线正确率也会使错误样本占据显著比例. 训练把质量提升寄托于验证器时，必须独立估计误通过和误拒绝，而不能只记录总体通过率.

26.2.2 经验证的蒸馏及监督区域

推理蒸馏（Reasoning Distillation）让学生学习教师产生的答案、轨迹或选择偏好. 这里的蒸馏以经验证数据为载体，与对齐 logits 或隐藏状态的表示蒸馏不同. 教师更大、轨迹更长、措辞更自信，均不能代替任务验证.

对目标序列 $y = (z, a)$，令 $m_t^{(z)}$ 和 $m_t^{(a)}$ 分别标记过程与答案，监督目标可写为

$$\mathcal{L} = -\frac{1}{N} \sum_t [\lambda_z m_t^{(z)} + \lambda_a m_t^{(a)}] \log \pi_\theta(y_t \mid x, y_{<t}), \tag{26.6}$$

这里取 $\lambda_z, \lambda_a \geqslant 0$，并定义 $N = \sum_t [\lambda_z m_t^{(z)} + \lambda_a m_t^{(a)}] > 0$，即按加权有效目标量归一化；$N = 0$ 的样本不产生该项更新. 若改用固定词元数作分母，权重还会改变整体梯度尺度，属于另一种明确的目标配置. 只生成答案的样本、过程与答案共同监督、过程可见但只监督答案，是三种不同的条件与目标配置. 第三种情况下过程仍影响答案预测，不能说“过程不参与训练计算”. 独立样本装箱后还需保持注意力可见性边界.

数据行应保存问题来源、题族标识、教师版本、采样配置、解析器和验证器版本、答案、可执行证据、各奖励分量、接受原因与内容摘要. 若每题只保留成功候选，低成功率难题更容易从数据集中消失；因此应同时保存尝试数和失败类型，按问题而非仅按保留序列统计覆盖. 只保留一种“标准路径”也会缩窄解法多样性.

26.3 组内相对优势

26.3.1 均值及方差约定

组相对策略优化（Group Relative Policy Optimization，GRPO）对同一问题生成一组回答，以组内奖励构造相对信号，省去独立价值模型的一种常见做法由 DeepSeek-

Math 提出. [139] 省去价值模型并不消除生成、验证、参考策略和训练稳定性的成本.

本章采用分母为 G 的组内方差:

$$\bar{r} = \frac{1}{G}\sum_{i=1}^{G} r_i, \qquad s_r^2 = \frac{1}{G}\sum_{i}(r_i - \bar{r})^2, \qquad A_i = \frac{r_i - \bar{r}}{s_r + \varepsilon}. \tag{26.7}$$

这里 $G \geqslant 2$, $\varepsilon > 0$. 组内必有 $\sum_i A_i = 0$; 当 $s_r > 0$ 且忽略稳定项时, $G^{-1}\sum_i A_i^2 = 1$. 若使用分母 $G-1$ 的样本方差, 标准差扩大为 $\sqrt{\frac{G}{G-1}}$ 倍, 优势缩小为 $\sqrt{\frac{G-1}{G}}$ 倍. 两种实现并不数值等价, 尤其在小组中差异明显.

对全组奖励相等的情形, 分子均为零, 按式 (26.7) 得到全零优势. 这类**零方差组** (Zero-variance Group) 不提供组内排序信号; 它可能全对, 也可能全错. 若目标含独立 KL 正则, 该组仍可能有正则梯度. 丢弃零方差组会改变问题采样权重, 应记录其比例与丢弃规则, 不能把没有奖励梯度的样本简单称为 "无用问题".

二元奖励且单候选正确率为 p 时, 独立采样的零方差概率为

$$\Pr(s_r = 0) = p^G + (1-p)^G. \tag{26.8}$$

当 p 接近 0 或 1 时, 大量组无法产生相对区分. 扩大组规模可能改善混合组比例, 但增加生成成本; 提高温度可能增加探索, 也可能降低正确率和破坏格式. 应在采样成本与有效奖励组比例之间比较, 而不是假定更大组总是更好.

26.3.2　组均值基线偏差

为了看清基线作用, 先去掉标准差归一化、裁剪、长度归一化与 KL. 令

$$g_i = \nabla_\theta \log \pi_\theta(y_i \mid x), \qquad g = \mathbb{E}[r_i g_i], \tag{26.9}$$

并在采样策略等于当前策略处讨论期望. 由概率归一化, $\mathbb{E}[g_i] = 0$. 给定问题后候选独立, 故对 $j \neq i$ 有 $\mathbb{E}[r_j g_i] = \mathbb{E}[r_j]\mathbb{E}[g_i] = 0$. 然而组均值包含 r_i 本身, 因此

$$\mathbb{E}[\bar{r} g_i] = \frac{1}{G}\mathbb{E}[r_i g_i] + \frac{1}{G}\sum_{j\neq i}\mathbb{E}[r_j g_i] = \frac{g}{G}, \tag{26.10}$$

$$\mathbb{E}[(r_i - \bar{r}) g_i] = \left(1 - \frac{1}{G}\right) g. \tag{26.11}$$

固定 G 时这是梯度整体缩放, 方向仍相同, 但不能称为未经修正的无偏估计. 如果组大小变化, 该缩放还会改变不同问题的相对权重.

留一基线 (Leave-one-out Baseline, LOO Baseline) 定义为 $b_{-i} = (G-1)^{-1}\sum_{j\neq i} r_j$. 它在条件独立假设下与第 i 个动作独立, 满足

$$\mathbb{E}[(r_i - b_{-i}) g_i] = g, \qquad r_i - b_{-i} = \frac{G}{G-1}(r_i - \bar{r}). \tag{26.12}$$

但重新除以由整组奖励决定的 s_r 后，分母与当前样本相关，前述无偏推导不再直接成立. 标准化还把同样绝对奖励差映射为不同幅度，改变问题之间的训练权重. 不能用“基线不改变期望”一句话替代对样本依赖和分母的检查.

例如 $G = 2$ 且奖励不同时，忽略 ε 后优势总为 $+1, -1$，无论奖励差为 1 还是 0.001. 这个变换保留顺序，却抹去了差值大小. 因此标准化是优化设计选择，不是恢复真实价值函数的证明. 关于长度和奖励标准差归一化所引入的偏置，已有专门分析与变体研究. [141]

26.4 概率比、裁剪及 KL

26.4.1 序列比及词元比

自回归分解给出

$$\pi_\theta(y_i \mid x) = \prod_{t=1}^{T_i} \pi_\theta(y_{it} \mid h_{it}). \tag{26.13}$$

在同一停止规则和支持集条件下，完整序列的重要性比为

$$w_i = \frac{\pi_\theta(y_i \mid x)}{\pi_{\text{old}}(y_i \mid x)} = \prod_t \rho_{it}, \qquad \rho_{it} = \exp(\ell_{\theta,it} - \ell_{\text{old},it}). \tag{26.14}$$

对固定函数 $f(y)$，才有完整分布变换恒等式

$$\mathbb{E}_{y\sim\pi_\theta} f(y) = \mathbb{E}_{y\sim\pi_{\text{old}}}[w(y)f(y)]. \tag{26.15}$$

单个 ρ_{it} 仅修正给定历史下的动作分布，不修正此前历史的分布；将多个词元比取平均不能替代它们的乘积.

序列比随长度累积，可能具有很大方差. 词元级近端目标把旧策略采样的历史作为局部优化条件，是有意构造的代理目标，而不是无条件等价的序列重要性采样. 即使采用完整 w_i，组内归一化优势依赖其他候选，变换整个组分布还需要处理其他样本；不能将其与固定奖励的单序列恒等式混为一谈.

还需区分模型原始 softmax 与实际生成策略. 温度缩放改变条件概率，截断采样改变支持集. 若生成使用温度或核采样，却用另一分布的对数概率充当行为概率，式 (26.14) 就不再是实际采样比. 特别是行为分布删除的词元在目标分布中仍有正概率时，完整无偏重加权无法从已采样数据恢复那些位置. 实现应明确自己优化的策略、行为策略和允许的近似.

26.4.2　裁剪目标的分段结构

定义 $\mathrm{clip}(u,l,h)=\min(\max(u,l),h)$，取 $0<\delta<1$．一种按回答平均的 GRPO 代理目标为

$$J(\theta)=\mathbb{E}\frac{1}{G}\sum_{i}\frac{1}{T_i}\sum_{t}\left[L_{it}^{\mathrm{clip}}-\beta K_{it}\right], \tag{26.16}$$

其中

$$L_{it}^{\mathrm{clip}}=\min\left(\rho_{it}A_i,\mathrm{clip}(\rho_{it},1-\delta,1+\delta)A_i\right). \tag{26.17}$$

优势、行为概率与参考策略参数在更新中冻结．实际最小化的损失为 $\mathcal{L}=-J$．当 $A_i>0$ 时，

$$L_{it}^{\mathrm{clip}}=A_i\min(\rho_{it},1+\delta); \tag{26.18}$$

当 $A_i<0$ 时，

$$L_{it}^{\mathrm{clip}}=A_i\max(\rho_{it},1-\delta). \tag{26.19}$$

正优势动作的概率比过大时，继续增大不再提高该项；负优势动作的概率比过小时，继续减小不再提高该项．相反方向仍保留惩罚．裁剪不是把所有概率比强制限制在区间，也不保证更新后实际 KL 一定小于某个阈值．

在未裁剪且非边界的位置，对已选词元对数概率的局部导数为 $A_i\rho_{it}$，再乘相应归约权重．其他参数仍通过 softmax 和共享网络耦合，因此某位置被裁剪不意味着其输出概率在整次参数更新中绝对不变．

26.4.3　KL 值及采样估计条件

在固定历史 h 下，令 $p(v)=\pi_\theta(v\mid h)$、$q(v)=\pi_{\mathrm{ref}}(v\mid h)$．精确的正向 KL 为

$$D_{\mathrm{KL}}(p\|q)=\sum_{v}p(v)\log\frac{p(v)}{q(v)}. \tag{26.20}$$

它需要词表求和．DeepSeekMath 使用了下述非负采样形式．[139] 令 $u(v)=\dfrac{q(v)}{p(v)}$，考虑单样本量

$$\kappa(v)=u(v)-1-\log u(v). \tag{26.21}$$

由 $\log u\leqslant u-1$ 可知 $\kappa(v)\geqslant 0$．若 $v\sim p$ 且两分布具有适当共同支持，

$$\mathbb{E}_p\kappa=\sum_{v}p(v)\frac{q(v)}{p(v)}-1+\sum_{v}p(v)\log\frac{p(v)}{q(v)} \tag{26.22}$$

$$=D_{\mathrm{KL}}(p\|q). \tag{26.23}$$

这就是该非负估计式的期望依据；采样来自当前分布是关键条件．若实际词元来自冻结行为分布 b，通常 $\mathbb{E}_b\kappa\neq D_{\mathrm{KL}}(p\|q)$．固定历史下，可在支持集充分时写为 $\mathbb{E}_b[(\frac{p}{b})\kappa]$；旧历史分布与当前历史分布的差异仍未因此消失．

数值上令 $p=(0.6,0.4)$、$q=(0.5,0.5)$，则两种词元的 κ 分别约为 0.015655 与 0.026856. 当前分布加权得到 0.020136，等于精确 KL；若按行为分布 $(0.5,0.5)$ 平均，则约为 0.021256，已经不同.

还必须区分无偏的数值估计与无偏的梯度. 因为采样分布随参数变化，

$$\nabla_\theta \mathbb{E}_{v\sim p_\theta}\kappa_\theta(v)=\mathbb{E}_p[\nabla_\theta\kappa_\theta(v)+\kappa_\theta(v)\nabla_\theta\log p_\theta(v)]. \tag{26.24}$$

对冻结采样词元只求 $\nabla\kappa$，会遗漏第二项. 若使用行为分布下的 $(\frac{p}{b})\kappa$ 并对概率比一并求导，固定历史处才可恢复对应完整期望的梯度. 实际训练可以有意采用局部近似，但必须说明采样分布、停止梯度位置和更新轮数，不能把“非负 KL 估计器”直接称为精确 KL 正则的所有实现.

例 23.1

令同一问题生成四条回答，奖励为 $(1,1,0,0)$，有效长度为 $(2,4,2,4)$. 这里奖励为构造输入，不对应真实模型测量. 组均值 $\bar r=0.5$，分母为 4 的方差为

$$s_r^2=\frac{4\times 0.5^2}{4}=0.25,\qquad s_r=0.5. \tag{26.25}$$

忽略仅用于稳定的 ε，优势为 $(1,1,-1,-1)$. 若改用样本标准差 $\sqrt{\frac{1}{3}}$，优势则为 $\pm\frac{\sqrt{3}}{2}$. 留一基线对应未标准化优势为 $(\frac{2}{3},\frac{2}{3},-\frac{2}{3},-\frac{2}{3})$，与当前标准化优势又是不同对象.

设 $\delta=0.2$，暂令 $\beta=0$ 以单独展示裁剪. 各生成位置的概率比和结果如下：

回答	优势	词元概率比	裁剪项平均
1	$+1$	$(1.3,0.9)$	$\frac{1.2+0.9}{2}=1.05$
2	$+1$	$(1.1,1.1,1.1,1.1)$	1.1
3	-1	$(0.7,1.1)$	$\frac{-0.8-1.1}{2}=-0.95$
4	-1	$(0.9,0.9,0.9,0.9)$	-0.9

因此按回答平均的目标为

$$J=\frac{1.05+1.1-0.95-0.9}{4}=0.075,\qquad \mathcal{L}=-0.075. \tag{26.26}$$

第一条的首词元因正优势且比值超过 1.2，该奖励项对其已选对数概率的局部导数为零；第二个词元导数为 $\frac{0.9}{4\cdot 2}=0.1125$. 第三条首词元因负优势且比值低于 0.8，也落在常数分支；第二个词元的导数为 $-\frac{1.1}{4\cdot 2}=-0.1375$. 第二条每个位置导数为 $\frac{1.1}{16}=0.06875$，第四条每个位置为 $-\frac{0.9}{16}=-0.05625$. 最小化损失时这些导数再取负号.

四条完整序列比却分别是

$$(1.3\cdot 0.9,\ 1.1^4,\ 0.7\cdot 1.1,\ 0.9^4)=(1.17,1.4641,0.77,0.6561). \tag{26.27}$$

它们不能由表中平均值替代. 若改为对全部 12 个有效词元平均，裁剪项总和为 $2.1+4.4-1.9-3.6=1$，目标为 $\frac{1}{12}$；若统一除以 $GT_{\max}=16$，目标为 $\frac{1}{16}$. 三个目标共享同一候选和奖励，却给出不同的权重与梯度尺度.

再加入相同历史条件下的 KL 后，每个有效位置需按选定估计方式计算 K_{it}，然后乘 β 与相同归约权重；不能把刚才的二元词表 KL 数值无条件复用到全部位置，因为每个历史的策略分布不同. 将任务奖励、裁剪和正则分别计算，能够防止一个最终标量掩盖实现差异.

26.5　长度、探索及奖励投机

26.5.1　长度归一化偏差

按回答平均、按整个批量有效词元平均与使用固定分母，分别对应

$$\frac{1}{G}\sum_{i}\frac{1}{T_i}\sum_{t}L_{it},\qquad \frac{\sum_{i,t}L_{it}}{\sum_{i}T_i},\qquad \frac{1}{GT_0}\sum_{i,t}L_{it}, \tag{26.28}$$

其中 T_0 是预先固定的常数. 第一个目标让每条回答的归约总权重相等，但同一优势对应的单词元权重与长度成反比；第二个目标按实际词元数汇总，随机分母还随批次变化；第三个目标避免以样本长度作分母，但需要独立控制梯度尺度和生成预算.

在纯序列终点回报 $J=\mathbb{E}[r(y)]$ 的策略梯度中，$\nabla\log\pi(y)=\sum_{t}\nabla\log\pi(y_t\mid h_t)$，没有自动出现 $\frac{1}{T_i}$. 引入长度平均因此改变了相对于该原始目标的样本权重. 对于相同负优势，长回答的每词元负向权重变小，可能造成不期望的长度动力学；实际变化还取决于停止符、奖励和参数耦合，不能仅由一个分母预测所有训练结果.

奖励中显式加入成本，例如 $r'=r_{\text{task}}-\lambda T_i$，是在定义新的目标. 它可能控制冗长，也可能使模型过早停止必要求解. 相比事后含糊地说“鼓励简洁”，应明确总生成上限、截断奖励、超时处理和成本权重；高分但缺少最终答案的超长样本不能悄悄从统计中消失.

26.5.2　奖励投机及分布变化

奖励投机（Reward Hacking）指策略通过满足评分规则而偏离实际任务目标. 答案解析器只提取最后一个数字时，模型可能输出多个互相矛盾的答案；测试集覆盖不足时，程序可能针对少数测试硬编码；过程奖励若偏好特定连接词，模型可能增加无效步骤；工具奖励若只检查返回文本，策略可能学会伪造成功声明.

一种分层设计是先检查必要约束，再对有效候选评分：

$$r(y)=\begin{cases} r_{\text{invalid}}, & \text{格式或执行契约不满足},\\ r_{\text{task}}(y)+\lambda_p r_{\text{process}}(y)-\lambda_c c(y), & \text{必要约束满足}. \end{cases} \tag{26.29}$$

但约束本身也可能不完备，线性权重也不能把不相容质量维度变成真值. 安全或权限要求若是不可违反条件，应在环境执行边界阻断，不依赖期望奖励中的小罚分阻止越界.

探索不足（Insufficient Exploration）会使策略反复产生相似候选，表面组规模较大，实际解法覆盖很小. 应同时观察奖励分量、零方差组比例、候选重复率、熵、回答长度和独立正确率. 训练奖励持续上升而独立正确率不变，可能来自验证漏洞、问题难度分布变化或选择性丢弃失败组.

生成和更新分离时，排队中的候选可能来自多个旧策略. **策略滞后**（Policy Lag）扩大后，词元比偏离 1，裁剪比例与重要性权重方差可能上升. 解决问题需要记录真实行为版本、限制样本陈旧程度和更新复用次数，而不只是增大裁剪区间. 奖励、解析器或环境版本改变后，相同文本的分数也可能改变，因此旧分数与新规则不可无条件混用.

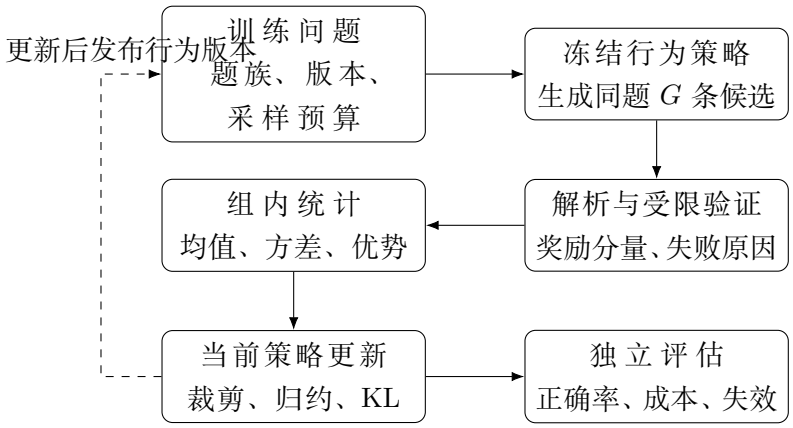


图 26.1: 可验证推理训练的数据与策略循环. 独立评估不向训练回流隐藏答案.

算法 26.1 版本化组相对策略更新

输入：问题集、初始策略、冻结参考策略、验证器版本、G、生成预算、裁剪宽度、KL 方案与归约规则. 输出：候选策略及与之对应的训练证据. 状态：当前参数、行为版本、待处理候选和优化器状态.

1. 固定一轮行为策略及其实际采样分布，对每题独立生成 G 条候选，保存有效词元、停止原因、行为对数概率与版本.

2. 在隔离环境中验证候选，保存奖励分量和无法判定原因；不得把验证器异常自动解释为候选正确.

3. 按问题计算均值与声明的方差. 零方差组按预定策略处理，保留统计. 将奖励、优势和行为概率从本次自动微分图中分离.

4. 对当前策略重新计算已生成位置的条件概率，构造词元比、分段裁剪项和选定的 KL 项. 以有效掩码及统一归约口径聚合，执行有限次更新.

5. 记录裁剪比例、概率比、长度、奖励、KL 和吞吐；样本超过允许陈旧程度或更新预算后不再复用．保存策略及验证器、模板、采样配置的版本关系．

6. 按预定训练预算结束，并在隔离题集上比较冻结基线和候选策略．若继续下一轮，发布新的行为版本并重新采样．

不变量：同组问题相同；概率分母对应实际行为版本；填充和环境观察不冒充生成动作；评估答案不进入训练奖励输入；失败与超时有显式记录．

26.6　测试时计算及采样指标

26.6.1　推理强度及硬预算

推理强度（Reasoning Effort）是推理模型在一次请求中采用多少内部推理工作的离散控制量．它通常以低、中、高等档位暴露，但档位集合、默认值和实现含义由模型及接口版本决定；同名档位不能跨模型换算为相同词元数、浮点操作数或思考深度．提高档位常会增加推理词元、延迟与费用，也可能改善复杂任务的成功率，但不是逐题单调改进的保证．

推理强度需要与三个容易混淆的控制量分开．输出上限是硬约束：以 OpenAI Responses API 为例，`max_output_tokens` 同时约束内部推理词元和可见输出词元；可见答案的详略由输出详细度等参数控制；温度、核采样等参数控制采样分布．推理强度也不等于候选数、搜索深度、工具调用次数或可见思维链．闭源接口即使报告推理词元，也通常不会返回完整内部轨迹；开源模型是否暴露轨迹则取决于模型与模板，不能由参数名称推断．

接口字段同样不是通用协议．OpenAI 在 2026 年的 Responses API 中使用 `reasoning.effort`，Chat Completions 使用 `reasoning_effort`；具体模型支持的档位仍须查阅该模型版本的能力说明 [142]．例如 Responses 请求可以表达为：

```
{
  "model": "<pinned-model>",
  "input": "<task>",
  "reasoning": {"effort": "low"},
  "max_output_tokens": 4096
}
```

这里的 `low` 是服务方解释的策略档位，并不承诺固定生成多少推理词元．若模型不支持该值，客户端或网关应得到明确错误，不能把静默回退后的结果记作低档实验．

评估推理强度时，应固定模型快照、提示模板、工具、采样设置、输出上限和题集，只改变档位．每个档位重复运行，至少记录最终正确率、可验证完成率、可见输出词元、推理词元（若接口提供）、首词元与端到端延迟、费用、超时和截断原因．若

高档位在输出上限内消耗更多内部词元，可见答案反而可能更早截断；这属于预算竞争，需要与能力不足区分.

26.6.2 候选覆盖及最终选择

测试时计算（Test-time Compute）是在参数固定时，为单次求解投入更多生成、验证或搜索. 延长单条轨迹、生成多候选、调用工具与扩展搜索树改变的计算路径不同. 训练使单候选正确率提高，与测试时增加候选数提高覆盖概率，应在相同评估预算下分别比较.

自一致性（Self-Consistency）对多条生成轨迹的答案进行聚合，常以规范化后的多数答案作为输出. [80] **多候选择优**（Best-of-N）则依赖评分器或验证器选择候选. 多数票可以集中到共同错误上，择优评分也可能偏爱错误答案. 因此候选集中“存在正确解”与系统“返回正确解”是不同指标.

若单候选正确概率为 p，固定策略独立生成 k 条，至少一条正确的概率为

$$P_k = 1 - (1-p)^k, \qquad P_{k+1} - P_k = p(1-p)^k. \tag{26.30}$$

式中递减的是理想独立覆盖的边际增益. 共享搜索路径、去重采样、束搜索与自适应提示通常不符合该独立模型. 实际输出还需要选择器识别正确解；没有可用选择器时，覆盖率不能作为服务返回准确率.

若每题成功率为 p_x，题集平均覆盖率是 $\mathbb{E}_x[1-(1-p_x)^k]$，一般不等于 $1-(1-\mathbb{E}_x p_x)^k$. 难度异质性使先平均 pass@1 再套公式产生另一层错误. 应先逐题估计，再按预先定义的问题权重汇总.

26.6.3 pass@k 估计的组合推导

前 k 次采样通过率（Pass at k，pass@k）通常衡量 k 条独立候选中至少一条通过任务检查的概率. 代码模型评测提出从每题 $n \geqslant k$ 条候选构造组合估计器. [44] 设其中 c 条通过，从已有 n 条中均匀无放回取 k 条，全部失败的子集数为 $\binom{n-c}{k}$，总子集数为 $\binom{n}{k}$，因此

$$\widehat{\text{pass@}k} = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}. \tag{26.31}$$

当 $n-c<k$ 时失败组合数为零；当 $c=0$ 时估计为零. 无放回选择是估计器内部的组合操作，不表示原始模型必须无放回生成.

说明其无偏性的一种方法是对所有大小为 k 的索引子集定义指示量 I_S，表示该子集中全部候选失败. 式 (26.31) 中的失败比例就是这些指示量的平均. 如果原始 n

条在固定问题与策略下独立同分布，那么任意固定子集都有 $\mathbb{E}[I_S] = (1-p)^k$，故

$$\mathbb{E}\left[\frac{1}{\binom{n}{k}}\sum_{|S|=k} I_S\right] = (1-p)^k. \tag{26.32}$$

线性期望不要求不同子集彼此独立，但要求原始抽样契约成立．这一区别解释了为何组合估计有效，而任意相关搜索结果不能直接套用同样的无偏结论．

例如 $n=10, c=3, k=2$，估计为

$$1 - \frac{\binom{7}{2}}{\binom{10}{2}} = 1 - \frac{21}{45} = \frac{8}{15} \approx 0.5333. \tag{26.33}$$

直接将 $\frac{c}{n} = 0.3$ 代入 $1-(1-p)^2$ 会得到 0.51，两者不同．后者在有限样本下具有插值偏差．计算大组合数时可使用

$$\frac{\binom{n-c}{k}}{\binom{n}{k}} = \prod_{j=0}^{k-1} \frac{n-c-j}{n-j} \tag{26.34}$$

或其对数形式，避免显式阶乘．

指标还必须声明验证器、生成温度、词元上限和工具预算．若在首次成功后停止采样，最终 n 受结果影响，就不再是前述固定样本数设计．相关采样可以报告固定算法与固定预算下的经验覆盖率，但应按题或独立运行计算不确定性，不把每条相关轨迹视为独立样本．pass@k 也不等于验证器择优准确率，更不是执行任意 k 次就能保证成功．

26.6.4　统一计算预算

设第 i 个候选生成成本为 $c_{g,i}$，验证成本为 $c_{v,i}$，工具成本为 $c_{u,i}$，总预算 C 应满足

$$\sum_{i=1}^{k} (c_{g,i} + c_{v,i} + c_{u,i}) \leqslant C. \tag{26.35}$$

成本可以计词元、设备秒或货币，但同一约束中必须使用可比较单位；延迟、峰值内存和调用次数通常还需单独限制．候选并行生成可降低墙钟延迟，却不会消除总设备工作量．验证器较昂贵时，增加候选的成本也不能只按生成词元计．

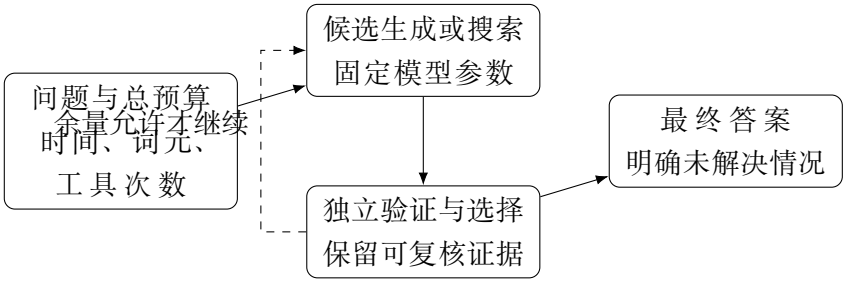


图 26.2: 测试时计算以剩余预算约束继续生成与验证，输出质量由选择结果而非候选数量决定.

自适应预算可以在不确定时追加候选，但必须避免把隐藏测试答案作为停止依据. 通过验证器后提前结束是实际系统的一种策略，应直接测量该策略的正确率、耗时和失败率；不能把其变长采样日志再解释为固定 k 的无偏评测. 单位正确答案成本可按总消耗除以正确返回数统计，同时报告无法回答和超时的比例.

26.7　工具任务中的轨迹及信用分配

工具任务将文本生成扩展为环境交互. 设历史为 h_t，策略选择动作 a_t，环境按 $P(e_{t+1} \mid e_t, a_t)$ 转移并返回观察 o_{t+1}. 完整交互轨迹概率包含策略与环境两部分：

$$p_\theta(\tau) = p(e_0) \prod_t \pi_\theta(a_t \mid h_t) P(e_{t+1}, o_{t+1} \mid e_t, a_t). \tag{26.36}$$

当环境转移不依赖待训练参数时，对数概率梯度只保留策略动作项：

$$\nabla_\theta \log p_\theta(\tau) = \sum_t \nabla_\theta \log \pi_\theta(a_t \mid h_t). \tag{26.37}$$

工具输出作为观察影响后续动作，但不是由策略采样产生的动作；直接把工具返回文本加入策略损失，会错误地归因模型控制不了的内容.

如果终点才有任务奖励，早期检索、计算和修正动作共享稀疏反馈. 过程奖励可以根据状态约束、有效工具调用或可核验子目标提供信号，但“调用越多得分越高”会鼓励无效调用；“每步执行成功”也不能保证完成用户目标. 动作成本与必要权限应分开建模，环境需要保证策略无法自行扩大授权范围.

沙箱验证（Sandbox Verification）限定工具执行的文件、网络、CPU、内存、时间与副作用范围. 训练中的大量探索不能直接作用于真实业务系统；应使用具有明确状态和转移规则的受控任务环境. 离线轨迹评估适合分析已有行为，但未覆盖动作的环境后果无法由日志凭空推断. 真实系统验证需在授权范围内逐步建立证据，不能从离线奖励上升直接推定部署可用.

环境非确定性、工具版本变化、检索内容变化和限流都会改变轨迹分布. 同一动作在不同时间失败，不一定由策略质量导致. 记录应包含初始状态、工具版本、请求参数、观察、错误类别、终止原因和奖励版本，以便区分策略失败、环境失败和验证器失败.

26.8　数据污染及独立评估

测试污染（Test Contamination）不仅是训练集出现完全相同题目，还包括答案、题目改写、同模板实例、教师生成时可见的测试材料，以及反复根据测试结果调参.

只做字符串去重不能证明题族独立；只把测试文件放在另一个目录也不能阻断奖励服务读取答案.

应先按题源、生成模板或任务族划分，再生成候选、标注过程和构建蒸馏数据. 验证器开发集与最终隐藏评测集也应分开，避免反复修规则实际上适配了测试答案. 教师版本和已知训练来源需记录；当无法排除教师接触过测试题时，应明确证据边界，并使用新的、独立构造或时间后移任务补充评估. [2]

训练时保留原始奖励及其分量，评估时使用独立任务标准. 推荐同时报告答案正确率、过程首错定位、解析失败、无法判定、零方差组比例、长度分布、pass@k、选择器返回正确率与总成本. PRM 的分类准确率或排序指标描述其标签任务，不能自动替代最终求解准确率；自然语言轨迹与参考文本的相似度也不适合作为逻辑正确性的唯一标准.

同一基座经训练后 pass@1 提高，可以支持固定单样本预算下的行为改善；只在更大 k 下提升，则还需分析采样覆盖与选择机制；奖励提升但独立指标不升，则不能宣称推理能力提升. 按任务、难度、语言、长度和工具类型分层，保留分子、分母与不确定性，才能判断平均变化是否掩盖了某类任务退化.

可验证奖励的证据边界

奖励定义优化方向，验证器定义可检查范围，采样策略决定能发现哪些候选，归约与概率比决定如何更新参数. 只有将这四项明确分开，并以独立问题和固定预算评估最终返回结果，才能判断训练是否改善了预期推理任务.

[2] 去污染措施提供的是降低污染风险的证据，不是未知预训练语料完全不含测试材料的证明. 报告应区分已检查的重叠方式与无法观察的来源.

第四部分

训练推理系统

第二十七章　模型规模及资源预算

一个模型能否在设备上运行，以及能否在期限内完成训练，是两个不同问题．前者取决于峰值状态和工作区，后者取决于总工作量与有效处理速度．只给参数量，既不能唯一确定显存，也不能唯一确定设备数．本章从模块形状出发，建立参数、计算、存储与期限之间可以逐项解释的关系．

本章负责推导工作负载需要多少资源．下一章讨论 GPU、网络和存储怎样供给这些资源，分布式训练章再解释状态如何切分、通信怎样执行．以下预算均保留明确假设；构造数值例题不作为任何具体硬件的实测性能．

假设与适用范围

除另行说明外，考虑自回归 Transformer，查询总宽度满足 $h_q d_h = d$，键和值等宽；线性层不含偏置，前馈层采用三个矩阵的门控结构．将一次乘法与一次加法计为两个浮点操作．参数存储、缓存、激活和工作区分别计量，所有内存先换算为字节后再求和．

27.1　计量对象及单位

符号	含义
L, d, V, d_f	层数、隐藏宽度、词表大小、前馈中间宽度．
h_q, h_{kv}, d_h	查询头数、键值头数、每头宽度．
B, T, D	一次计算的序列批量、序列长度、整个训练过程消费的词元数．
P, F	参数数与浮点操作数，二者不能混用．
b_w, b_g, b_c	权重、梯度、缓存每元素字节数．
N, S, u	设备数、时间预算（秒）、按指定口径定义的有效利用率．
R_{peak}, β	单设备峰值操作率、有效数据传输带宽．

浮点操作数（Floating-point Operations，FLOPs）表示完成给定工作量所需的操作数量；FLOP/s 才是每秒操作率．参数数是独立可学习标量的数量．共享同一矩阵两次使用不会增加独立参数，却可能增加计算次数．

十进制 GB 为 10^9 字节，二进制 GiB 为 2^{30} 字节．若模型含 7×10^9 个参数，每参数两字节，权重为 14×10^9 字节，约 13.04 GiB．将“7B”与“7 GiB”混为一谈，

会在预算第一步就产生错误. [1]

峰值还必须带时间语义. 训练过程中权重、梯度、保存激活、通信缓冲和检查点暂存的生存区间不同, 逐项峰值直接相加可能过于保守; 只记录每项平均值又可能漏掉同时存在的瞬时峰值. 初步预算可先给保守上界, 再用执行时间线细化.

27.2 参数量核算

27.2.1 注意力及门控前馈层

查询投影 $W_Q \in \mathbb{R}^{d\times h_q d_h}$, 键值投影 $W_K, W_V \in \mathbb{R}^{d\times h_{kv} d_h}$, 输出投影 $W_O \in \mathbb{R}^{h_q d_h \times d}$. 因此

$$P_{\text{attn}} = dh_q d_h + 2dh_{kv} d_h + h_q d_h d \tag{27.1}$$

$$= 2d^2 + 2dh_{kv} d_h. \tag{27.2}$$

第二行使用了 $h_q d_h = d$; 若结构不满足这一条件, 必须保留第一行. GQA 减少键值投影参数, 但查询和输出两项没有因此消失.

门控前馈采用两条上投影和一条下投影, 矩阵形状分别为 $d \times d_f, d \times d_f, d_f \times d$, 故

$$P_{\text{ffn}} = 3dd_f. \tag{27.3}$$

普通两矩阵前馈则为 $2dd_f$. 比较两种前馈时要固定宽度还是固定参数预算, 应先作选择; 不能把系数二与三的差异忽略后宣称同样规模.

若每层有两组只含缩放参数的 RMSNorm, 末尾还有一组, 则归一化参数为 $(2L+1)d$. 若改为同时含缩放与偏置的 LayerNorm, 需要按实际数量重新计数. 偏置虽常是小项, 精确统计仍应说明是否包含.

27.2.2 输入输出共享及专家层

输入嵌入参数为 Vd. 输出投影若独立, 再增加 Vd; 若与输入共享, 则独立存储只计一次. 设 $\tau = 1$ 表示共享, $\tau = 2$ 表示独立, 在上述同构稠密结构下

$$P = \tau Vd + L(2d^2 + 2dh_{kv} d_h + 3dd_f) + (2L+1)d. \tag{27.4}$$

学习到的绝对位置表、额外任务头或查询键归一化参数不包含在该式中, 存在时必须逐项加入.

对于专家层, 若每个专家为门控前馈, 独立路由专家总参数为 $E \cdot 3dd_e$; 每词元执行的专家矩阵规模近似为 $k \cdot 3dd_e$. 共享专家、路由器和注意力另计. **激活参数影响**

[1] 硬件标称容量、操作系统显示和框架日志可能采用不同单位. 有些界面虽写 GB, 实际计算却用二进制除数, 资源报告应列出原始字节数及换算规则.

本次计算，总参数影响需要保存的模型状态. 它们不能在显存公式中互换. 路由不均衡还会使某设备的实际负载偏离平均值，详见第三十六章《混合专家模型》.

27.2.3　参数量核算

例 15.1

构造模型取 $L=12, d=1024, V=32000, h_q=16, h_{kv}=4, d_h=64, d_f=2816$，输入输出共享，归一化采用上述 RMSNorm 配置. 注意力每层参数为

$$2(1024)^2+2(1024)(4)(64)=2621440. \tag{27.5}$$

前馈每层为 $3(1024)(2816)=8650752$. 12 层主干矩阵共 135266304 个参数，嵌入为 32768000，归一化为 25600，合计

$$P=168059904. \tag{27.6}$$

若只把输出改为独立矩阵，增加 32768000 个参数. 若只把键值头由 4 改成 16，每层增加 $2d(16-4)d_h$ 个参数；其影响不等于把整个模型参数乘四.

27.3　矩阵计算量及训练近似

27.3.1　线性层前向及反向

对 $X\in\mathbb{R}^{n\times a}$ 和 $W\in\mathbb{R}^{a\times b}$，输出 $Y=XW$ 有 nb 个元素，每个元素执行 a 次乘法和约 a 次加法，故前向近似 $2nab$ FLOPs. 该线性层有 ab 个参数，因此每行输入的矩阵计算约为 $2P_W$.

反向需要 $\overline{X}=\overline{Y}W^{\mathsf{T}}$ 与 $\overline{W}=X^{\mathsf{T}}\overline{Y}$，两项各约 $2nab$，前向与反向合计约 $6nab$. 这解释了稠密模型训练中常用 $6PD$ 估计的来源：每词元约执行一次前向与两次同阶矩阵反向.

它不是所有参数都精确执行六次操作. 查表、共享参数的多次使用、冻结模块、优化器更新、专家路由和非矩阵算子都需要单独判断. 尤其在门控网络中，逐元素乘法和激活导数有额外成本；大矩阵占主导时才适合把它们作为次阶项.

27.3.2　注意力的长度平方项

对每条序列，QK^{T} 的计算近似为 $2h_qT^2d_h$，再与值矩阵相乘也约为同量级，因此

$$F_{\text{interaction}}\approx 4BT^2h_qd_h=4BT^2d \tag{27.7}$$

表示常规密集实现的两个主要矩阵乘法，不含投影与 Softmax. 利用因果三角结构的专用实现可能减少实际算术，但仅把未来分数设为负无穷的密集实现不一定节省计算.

令每层主干线性参数为 P_ℓ，输入输出投影的执行参数按实际路径统计，整个前向可以近似写为

$$F_{\text{fwd}} \approx 2BT\left(\sum_{\ell=1}^{L} P_\ell + Vd\right) + 4LBT^2d + F_{\text{elementwise}}. \tag{27.8}$$

输入查表不按一次 Vd 密集矩阵乘法计算，输出全词表投影却需要该项. 权重共享减少存储，不删除输出计算. 长上下文下第二项不能总被 $2PBT$ 吸收.

以 $P_\ell \approx 12d^2$ 的常见量级作粗略比较，注意力交互与该层线性前向之比约为 $\dfrac{T}{6d}$. 长度相对宽度很大时，忽略交互项会系统性低估成本. 该比例来自给定参数近似，不是所有模型统一的转折点.

27.3.3 重计算及有效工作量

激活重计算（Activation Recomputation）通过丢弃部分中间量，在反向时重新执行相应前向，从而用计算换显存. 若原训练前向计算为 F_f、反向为 F_b，额外重算为 F_r，实际计算为 $F_f + F_b + F_r$，不能仍把全部成本记成固定 $6PD$.

FlashAttention 等算法减少注意力中间矩阵的显式存储和数据搬运 [143]，不意味着注意力关系的数学定义发生变化. 使用同一公式计算的两个内核，可以因读写量、重算与硬件利用率不同而具有不同时间. 资源报告应分开模型有效工作量、实际执行工作量与墙钟时间.

27.4 推理显存及并发容量

27.4.1 权重存储及量化开销

未量化权重存储为 Pb_w. 若使用 q 位量化且每 g 个参数共享一个占 b_s 字节的尺度，忽略边界取整时

$$M_w \approx \frac{Pq}{8} + (\frac{P}{g})b_s + M_{\text{other}}. \tag{27.9}$$

M_{other} 包含零点、未量化层、对齐和打包元数据. 某些后端还需临时反量化工作区. 标称四位不能保证总显存恰为 BF16 的四分之一，也不能保证速度更快.

27.4.2 变长请求及键值缓存

对同构 GQA，若第 i 个并发请求已缓存 T_i 个位置，则

$$M_{\text{KV}} = 2Lh_{kv}d_hb_c\sum_{i=1}^{B} T_i. \tag{27.10}$$

当全部请求长度相同才化为 $2LBTh_{kv}d_hb_c$. 这里的长度包含已处理提示与已生成内容，不能只统计最终输出. 共享前缀、滑窗淘汰、潜在状态和跨设备切分会改变实际驻留方式，需替换相应状态模型.

若分页块容量为 p 个位置，且每条请求独立分配整块，则实际保留槽位为 $\sum_i p\lceil \frac{T_i}{p} \rceil$. 内部浪费小于 Bp 个位置，但还存在页表与运行时保留. 共享前缀时不能同时按逻辑引用数和物理页数重复统计.

27.4.3　并发容量约束

设设备可用显存为 M_0，权重及固定运行时占用 M_f，给定长度 T 的单请求缓存为 $m(T)$，工作区上界为 $W(B,T)$，则并发必须满足

$$M_f + Bm(T) + W(B,T) \leqslant M_0. \tag{27.11}$$

若假设工作区固定，可先求缓存约束的整数上界；实际工作区随批处理变化时，需要在候选并发上逐项计算. 平均请求长度只描述典型状态，不能保证长请求同时到达时不越界.

对于前述构造模型,取缓存每元素两字节,单请求每位置缓存为 $2(12)(4)(64)(2) = 12288$ 字节. 缓存 8192 个位置需要 96 MiB，32 个同长请求需要 3 GiB. 若键值头变成 16，其缓存变成四倍；权重、工作区和服务时延仍需分别计算.

27.5　训练状态、激活及分片

27.5.1　每参数字节数来自哪些状态

一种混合精度 Adam 配置保存两字节训练权重、两字节梯度、四字节主权重以及两个四字节矩估计，总计每参数 16 字节. 这个数字来自具体状态表，而不是 Adam 的不可变常数. 梯度累积可能使用四字节，某些实现不保留独立主权重，低精度优化器又会采用其他布局.

状态	本例每参数字节数	用途
计算权重	2	前向与反向的矩阵运算.
梯度	2	本次更新方向.
主权重	4	高精度参数累积.
一阶矩与二阶矩	8	自适应更新统计.

冻结基座不再为基座计算全部参数梯度和优化器状态，却通常仍需保存激活以向可训练适配器传递梯度. 因此 LoRA 或 QLoRA 的显存不能只按可训练参数百分比乘全参数训练显存.

27.5.2 分片降低哪些驻留量

设权重、梯度、优化器相关状态分别为 M_w, M_g, M_o，数据并行组有 N_d 个设备. 理想均匀分片的驻留模型为

$$M_{\mathrm{DDP}} = M_w + M_g + M_o, \tag{27.12}$$

$$M_{\mathrm{Z1}} = M_w + M_g + \frac{M_o}{N_d}, \tag{27.13}$$

$$M_{\mathrm{Z2}} = M_w + \frac{M_g + M_o}{N_d}, \tag{27.14}$$

$$M_{\mathrm{Z3}} = \frac{M_w + M_g + M_o}{N_d.} \tag{27.15}$$

这里按 ZeRO 的分阶段思想区分被分片对象 [144]；具体执行协议在第二十九章《分布式训练》展开. 公式没有包含临时聚合完整层参数的峰值、通信桶、预取和激活，也未保证所有层完全均匀.

若总参数状态 112 GB，八设备理想全分片后平均为 14 GB；但某一执行阶段仍可能临时聚合一个大层. 因此用总状态除设备数只能给驻留基线，不能直接当作单设备峰值.

27.5.3 激活模型的结构前提

简单的线性保存模型可写为 $M_a \approx c_a LBTdb_a$，其中 c_a 表示每层每位置保存多少个隐藏宽度等效元素. 它应由实际计算图或测量校准，而非固定成一个普适常数. 若显式保存注意力概率，还可能增加 $O(LBh_qT^2)$ 元素；使用分块内核与重算后，保存结构发生变化.

峰值训练显存可组织为

$$M_{\mathrm{train,peak}} = M_{\mathrm{resident}} + M_{\mathrm{activation}} + M_{\mathrm{gather}} + M_{\mathrm{communication}} + M_{\mathrm{workspace}} + M_{\mathrm{reserve}}. \tag{27.16}$$

各项同时驻留的时间应按执行顺序分析. 流水并行可能保存多个在途微批的激活；张量并行可能分摊部分张量，也可能复制其他状态. 因此不能对全部分项统一除以设备总数.

27.6 带宽、时间及设备数

27.6.1 算术强度及不可突破的下界

算术强度（Arithmetic Intensity）定义为操作数与相应存储层数据搬运字节数之比 $I = \dfrac{F}{Q}$. 忽略重叠细节时，单个工作负载的耗时至少满足

$$t \geqslant \max\left(\frac{F}{R_{\mathrm{peak}}}, \frac{Q}{\beta}\right). \tag{27.17}$$

一次调用若算术强度低，即使峰值矩阵吞吐很高，也可能受权重或缓存读取限制．提高批量可以复用权重，但同时增加缓存和延迟；性能问题不能只用峰值 FLOP/s 回答.

27.6.2　有效利用率及时间预算

设整个训练所需模型工作量为 F_{model}，N 个设备的指定精度峰值总和为 NR_{peak}．按模型工作量定义有效利用率

$$u = \frac{F_{\mathrm{model}}}{NR_{\mathrm{peak}}t_{\mathrm{wall}}}. \tag{27.18}$$

该定义将通信等待、数据等待、重算和其他开销反映在墙钟时间中．若另一个报告把重算计入分子，它采用不同口径，两个利用率不能直接比较．也不能将已包含通信损失的利用率再乘一次同样的通信折损系数.

在假设 u 对候选设备数近似稳定时，期限 S 要求

$$N \geqslant \left\lceil \frac{F_{\mathrm{model}}}{R_{\mathrm{peak}}uS} \right\rceil. \tag{27.19}$$

扩容后 u 往往变化，因此严格预算需要 $u(N, \text{拓扑}, B, T)$ 的情景模型．显存可行性也依赖具体并行方式，不能总用另一个独立常数卡数表示.

27.6.3　同一模型的训练及推理预算

例 15.2

沿用前面 $L = 12, d = 1024, V = 32000, h_q = 16, h_{kv} = 4, d_h = 64, d_f = 2816$ 的共享嵌入模型，总参数为 $P = 168059904$．扣除归一化参数，参与主要线性映射的参数为 $P_{\mathrm{lin}} = 168034304$．设处理约 $D = 10^9$ 个非填充位置、序列长度 $T = 2048$，忽略最后不满序列的边界修正．若每个独立块采用块内下一词元配对，有效监督量约为 $\dfrac{D(T-1)}{T}$；本段计算量使用处理位置数 D．不使用激活重算，注意力按稠密计算计数，则主要训练计算为

$$F \approx 6P_{\mathrm{lin}}D + 12LTdD = 1.310195712 \times 10^{18}\ \mathrm{FLOP}. \tag{27.20}$$

第二项由每层每序列注意力前向 $4T^2d$ 乘以前反向近似因子 3，再乘 $\dfrac{LD}{T}$ 得到．本例未计 softmax、逐元素运算和优化器更新，故是声明范围内的估算.

假设单设备峰值为 2×10^{13} FLOP/s，该计数口径下端到端有效利用率为 0.3，则运行约 218365.952 秒，即 60.66 小时．72 小时期限下，计算量约束允许一个设备．容量还需独立检查：按每参数 16 字节保存训练状态，占 2.5043 GiB；假设激活、工作区和余量合计 4 GiB，峰值约 6.5043 GiB，小于假定的 16 GiB 可用容量．4 GiB 是构造输入，不能从参数数目推出.

同一模型切换到推理时，两字节权重占 320.55 MiB；四条请求各缓存 2048 个位置，KV 占 $4 \times 2048 \times 12288 = 96$ MiB. 若另预留 512 MiB 工作区，总计约 928.55 MiB. 这个容量预算不提供时延保证；实际吞吐必须使用推理阶段的形状、内核与带宽重新评估. 由此，一组结构参数可以同时约束训练计算、训练驻留和推理缓存，但三个结果依赖不同的运行假设.

27.6.4 资源联合预算

例 15.3

假设某稠密任务含 $P = 7 \times 10^9$ 参数、训练 $D = 10^{11}$ 词元，暂用 $F = 6PD = 4.2 \times 10^{21}$ 操作的基线. 每设备峰值设为 2×10^{14} 操作每秒，模型利用率假设 .4，期限为 10 天，即 864000 秒. 计算约束为

$$N_{\text{compute}} = \left\lceil \frac{4.2 \times 10^{21}}{(2 \times 10^{14})(.4)(864000)} \right\rceil = 61. \tag{27.21}$$

若拓扑要求设备数为八的倍数，可行候选从 64 开始. 这里的 61 不是实测扩展曲线给出的答案；长上下文交互和重算若未包含在利用率口径中，必须另加工作量.

同一任务若采用本章 16 字节状态配置，总状态约 104.31 GiB. 设八路全分片，驻留状态约 13.04 GiB，再假设激活、聚合、工作区和余量共 8 GiB，则单设备约 21.04 GiB，低于假设的 40 GiB 可用容量. 该情景说明显存可能在八设备组内已经可行，而期限仍要求更大的集群. 64 设备的具体数据、张量和流水并行布局，以及其实际利用率，仍需由分布式执行方案确定.

算法 27.1　构建资源可行情景

输入：模型配置、词元预算、长度与并发分布、精度、候选拓扑、期限和容量余量. **输出：**每个候选的分项预算、可行条件和仍待测量的系数.

1. 按实际共享关系计数参数，分别建立总参数、执行矩阵和特殊状态清单.
2. 对各拓扑计算权重、梯度、优化器、缓存、激活及峰值临时量；拒绝违反单设备容量约束的情景.
3. 按线性、注意力交互与重算分解工作量，使用该拓扑的利用率假设估计时间；检查期限约束.
4. 计算数据供给、检查点、CPU 和主机内存需求，检查是否存在上游瓶颈.
5. 分别代入基准、保守和压力假设，记录结论改变的边界，而非只输出一个设备数.

未知测量系数保留为区间，不用虚构的精确小数填补. 最终选型前用目标环境测量校准，方法见算力基础设施章.

27.7　主机、存储及数据供给预算

训练吞吐目标若为每秒 r_t 个词元，物化数据平均每词元占 b_t 字节，则纯输入带宽下界为 $r_t b_t$；索引、压缩、重试和预取会增加其他开销. 若运行时还需解析和分词，应使用对应步骤的实际处理速率估计 CPU 需求，而不直接按 GPU 数量乘固定核数.

检查点大小为 C，允许写入窗口为 Δ，则平均有效写带宽至少为 $\dfrac{C}{\Delta}$. 异步写入不删除数据，只是将等待转移到主机内存、队列和后台 I/O；产生速度长期大于写出速度时，队列仍会无限增长或触发阻塞.

总存储应分开原始数据、清洗数据、词元分片、检查点、评估结果、日志和中间资产. 若保留 K 个完整检查点，每个大小 C，至少需要 KC，另加构建暂存和恢复副本. 分片文件数量影响元数据操作与小文件开销，容量充足不保证读取吞吐足够.

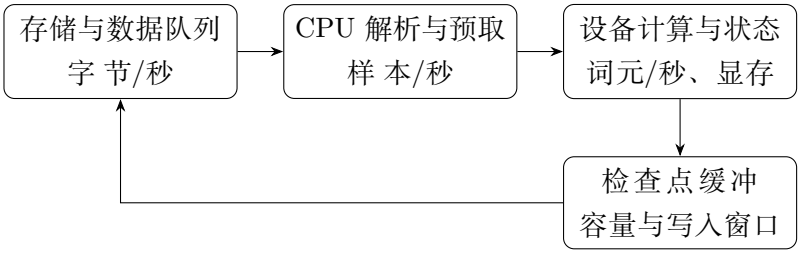


图 27.1: 资源供给与状态持久化形成闭环. 任一阶段的持续处理率不足，都可能限制整条训练路径.

27.8　敏感性及预算表达

预算报告应给出假设变化如何影响结论. 例如固定模型时，普通 KV 缓存随总驻留词元线性变化；密集注意力交互随单序列长度平方变化；模型利用率从 .4 降到 .3 时，在其他条件不变的近似下，所需设备时间增加三分之一. 参数量翻倍则同时影响状态与线性计算，但不必把所有缓存维度也翻倍.

余量要标明用途. 容量安全余量、故障备用设备、维护窗口和负载峰值是不同项；若某系数已经按高分位负载测得，再把同一负载波动完整重复加入，会高估预算. 反过来，不能为了满足费用上限而把未知工作区当成零.

最终预算是可追溯的情景集合：每个情景绑定模型版本、数据量、精度、长度分布、并行配置、有效速度与余量. 它允许读者解释某项变化造成的后果，并指出哪些假设必须在实施时重新测量.

第二十八章　算力基础设施

模型的计算量描述必须完成的工作，基础设施决定这些工作怎样获得数据、执行运算并保存结果. 即使参数和中间状态能够放入显存，训练仍可能因数据准备不及时、通信拥塞或检查点写入过慢而停顿. 反过来，增加设备峰值算力，也不会自动缩短一个受显存带宽限制的解码步骤.

第二十七章《模型规模及资源预算》建立了工作负载的资源需求. 本章建立供给侧的解释框架：先考察单个加速器内部，再沿主机、互连、存储与调度扩展到集群. 分布式训练章进一步讨论张量和优化器状态怎样切分；本章主要分析这些算法所依赖的物理通路及其约束. 所有数值例题均为明确设定输入的工程计算，不代表任何硬件型号的实测结果.

28.1　系统边界及计量单位

性能模型的共同前提

计算工作量固定，数值精度与输出质量要求一致；一次乘法和一次加法合计两个浮点操作. 带宽指指定路径上的有效数据字节率，不能把链路标称比特率直接代入. 除特别说明外，计算模型忽略启动和排队，通信模型单独计入启动延迟；后文将说明这些近似在哪些条件下失效.

符号	含义与单位
F, Q, I	运算量（FLOP）、跨指定存储层传输字节数（byte）、算术强度 $I = \dfrac{F}{Q}$ (FLOP/byte).
R_{peak}, R	与精度和算子匹配的峰值操作率、实际操作率，单位 FLOP/s.
$\beta_{\text{mem}}, \beta_{\text{net}}, \beta_{\text{io}}$	显存、网络和存储路径的有效带宽，单位 byte/s.
$\alpha, \beta_{\text{inv}}$	通信启动时间（s）与每字节传输时间（s/byte）；$\beta_{\text{inv}} = \dfrac{1}{\beta_{\text{net}}}$.
p, m	通信参与进程数与每进程初始消息长度（byte）.
S_c, C, τ, M, R_f	检查点大小（byte）、写入暂停时间、有效计算间隔、作业平均故障间隔、恢复时间，时间均以秒计.
B, d, k, n, s	矩阵批量或矩阵维度，以及每元素存储字节数；含义由对应公式明确指定.

前章以 β 表示带宽，本章仍以带下标的 β 表示字节率．经典通信文献所谓“α–β 模型”中的第二项通常表示带宽的倒数，因此本章专门写为 β_{inv}，避免同一个符号在两个模型中承担相反单位．

十进制 GB 与二进制 GiB 分别为 10^9 与 2^{30} 字节；GB/s 与 Gb/s 还相差每字节八比特．双向链路同时发送和接收的总标称值，也不等于某个单向传输能够使用的带宽．性能报告必须写明方向、协议开销、消息大小和并发条件．

28.2　加速器计算模型

28.2.1　线程、线程束及计算单元

图形处理器（Graphics Processing Unit，GPU）通过大量并发线程执行可分解的计算．程序中的一个线程通常负责若干数据元素，若干线程组成线程块，线程块再分配到片上的计算单元．线程块是资源分配和局部协作的重要边界，不能把软件线程数直接解释为物理算术单元数．

以 NVIDIA CUDA 的抽象为例，**单指令多线程**（Single Instruction, Multiple Threads，SIMT）以**线程束**（Warp）组织线程，通常每束包含 32 个线程；**流式多处理器**（Streaming Multiprocessor，SM）承担线程束调度和执行．线程保有各自寄存器和控制状态，执行时由活跃线程掩码决定哪些通道参与当前指令．同一束内的数据相关分支可能需要分别执行不同路径，从而降低有效通道利用率．独立线程调度并不意味着分支分歧的代价消失．上述名称和束宽属于特定编程体系，其他加速器的线程组结构与调度粒度应按其架构定义理解．[145]

线程束切换可以在一组线程等待内存时发射另一组就绪线程，但前提是仍有独立工作和可驻留资源．设一个 SM 的寄存器容量为 R_{SM} 个寄存器，每线程使用 r 个寄存器，每块 t 个线程，每块共享存储需求为 h 字节，则忽略分配粒度时，驻留块数受到

$$n_{\mathrm{block}} \leqslant \min\left(n_{\mathrm{arch}}, \left\lfloor \frac{R_{\mathrm{SM}}}{rt} \right\rfloor, \left\lfloor \frac{H_{\mathrm{SM}}}{h} \right\rfloor, \left\lfloor \frac{T_{\mathrm{SM}}}{t} \right\rfloor\right) \tag{28.1}$$

约束．其中 H_{SM} 是共享存储容量，T_{SM} 是最大驻留线程数，n_{arch} 是硬件块数上限；$h = 0$ 时去掉对应项．真实硬件还按分配粒度取整，因此这是解释资源约束的上界模型．

占用率（Occupancy）是活跃驻留线程束数与硬件允许的最大驻留线程束数之比．提高占用率可能改善延迟隐藏，但不保证提高性能．如果减少寄存器导致临时值溢出到较慢存储，或者更多线程争用相同带宽，较高占用率反而可能使运行时间增长．必须把占用率、发射效率、访存停顿和有效操作率放在一起解释．

28.2.2 矩阵单元及精度

大规模矩阵乘法具有规则的数据复用结构，专用矩阵乘加单元可用块状指令计算 $D = AB + C$. NVIDIA 将这类单元称为**张量核心**（Tensor Core）；不同平台对输入形状、数据类型、累加精度和稀疏结构的要求不同. 高峰值只有在运算被映射到相应指令、块形状能够有效填充且数据供应充分时才有意义.

以 $A \in \mathbb{R}^{m\times k}$、$B \in \mathbb{R}^{k\times n}$ 为例，主要运算量约为 $2mkn$. 若硬件块尺寸要求对维度补齐到 $\widetilde{m}, \widetilde{k}, \widetilde{n}$，则实现可能执行约 $2\widetilde{m}\widetilde{k}\widetilde{n}$ 个操作. 报告以有效数学操作数为分子时，补齐部分不能算作任务吞吐；以硬件指令操作数为分子时，则必须说明它包含无效填充.

半精度浮点（Half Precision Floating Point，FP16）与**脑浮点 16**（Brain Floating Point 16，BF16）均占两个字节，但指数和尾数分配不同. FP16 具有更细的同量级表示精度，BF16 具有更大的指数范围. 低位宽格式改变的是表示范围、舍入误差、传输字节数和可用指令，不能只看“位数更少”. **八位浮点**（8-bit Floating Point，FP8）还包含不同指数和尾数组合，具体格式必须与缩放规则、累加类型一并记录.

例如，把权重从两字节压缩到半字节，理论上可减少四倍权重读取字节，但实际还需读取缩放因子，并执行解码或反量化. 如果运行时先把整层权重展开成两字节矩阵再调用普通乘法，文件变小并不能保证关键路径上的带宽减少. 另一方面，即便输入采用低精度，某些归约和累加仍应使用更高精度. 数值方法与执行格式必须共同确认，不能以格式名称替代质量验证.

28.2.3 寄存器到显存的层次

高带宽内存（High Bandwidth Memory，HBM）为设备提供较大容量的外部存储，但线程每次从 HBM 取数，通常比复用片上数据付出更高的数据搬运成本. 寄存器保存线程私有的热数据，共享存储供块内协作，片上缓存减少重复外部读取. 容量逐层扩大时，访问成本通常也随之上升；这些层次不能视为等价的“大显存”.

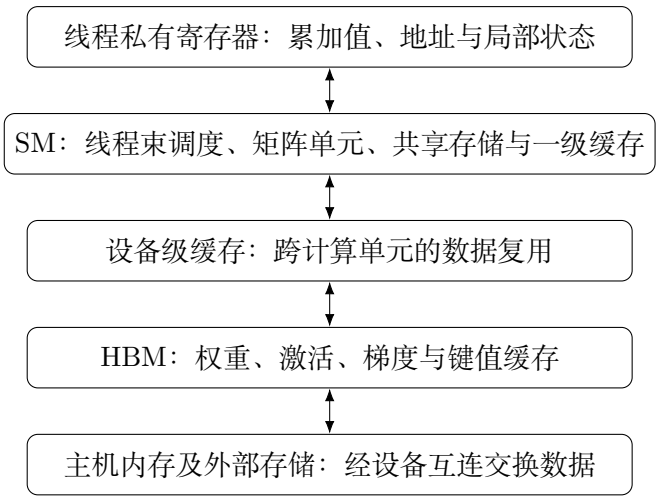


图 28.1: 加速器存储与执行层次. 图中表示逻辑路径，不表示所有架构具有相同缓存组织.

矩阵乘法的分块实现把输入片段载入片上存储，反复参与多个输出元素的计算，再写回结果；其目的首先是复用数据. 相邻线程访问相邻地址有利于**合并访存**（Coalesced Memory Access），而离散访问可能把有效字节分散到多个事务中. 共享存储中的访问也可能出现存储体冲突；加大分块一方面增加复用，另一方面消耗寄存器和共享存储，必须与驻留资源约束共同选择.

注意力中的大中间矩阵说明了这一层次的价值. 将分数矩阵完整写入 HBM 再读回，会产生额外流量. 分块与在线归一化可以避免这些中间写入，保持相同数学运算语义，而主要改变存储访问量. 类似地，算子融合省去中间张量的往返，却可能增加寄存器压力. 基础设施分析需要分别记录“减少了多少操作”和“减少了哪一层的多少字节”.

28.3　Roofline 性能模型

28.3.1　性能上界推导

设一个算子执行 F 个浮点操作，并在 HBM 与计算芯片之间传输 Q 字节. 即使计算单元没有停顿，运行时间也不能小于 $\dfrac{F}{R_{\text{peak}}}$；即使访存完全规则，也不能小于 $\dfrac{Q}{\beta_{\text{mem}}}$. 因此

$$t \geqslant \max\left(\frac{F}{R_{\text{peak}}}, \frac{Q}{\beta_{\text{mem}}}\right), \qquad R = \frac{F}{t} \leqslant \min(R_{\text{peak}}, I\beta_{\text{mem}}), \tag{28.2}$$

其中 $I = \dfrac{F}{Q}$ 称为**算术强度**（Arithmetic Intensity）. **屋顶线模型**（Roofline Model）将这两项上界画在同一坐标系中，区分带宽约束与计算约束. [146]

两项相等时的转折点为

$$I_* = \frac{R_{\text{peak}}}{\beta_{\text{mem}}}. \tag{28.3}$$

当 $I < I_*$ 时，提升峰值算力不会提高该模型中的上界；当 $I > I_*$ 时，仅提升 HBM 带宽也不会抬高计算平台. 实际算子可能远低于上界，因为依赖链、分支、同步、启动和不规则访存尚未计入. 若计算与访存不能充分重叠，时间还会向两项之和靠近.

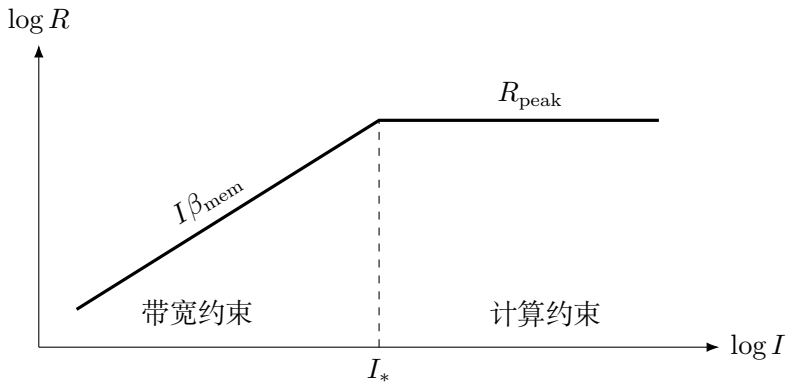


图 28.2: Roofline 的对数坐标示意. 折线是性能上界，不是测量曲线.

本模型必须注明字节跨越哪个层次. 用 HBM 字节计算的强度，不能乘以寄存器带宽；缓存命中减少 HBM 流量后，相应强度会改变. 峰值也必须匹配实际精度和稀疏条件. 将结构化稀疏峰值与稠密算子的操作量混用，会得到没有可比性的利用率. [1]

28.3.2 矩阵乘法批量计算

例 27.1

考虑 $Y = XW$，其中 $X \in \mathbb{R}^{B\times d}$、$W \in \mathbb{R}^{d\times n}$、$Y \in \mathbb{R}^{B\times n}$，每元素 s 字节. 若输入与权重各从 HBM 读取一次，输出写入一次，则理想流量与操作量为

$$Q_{\min} = s(Bd + dn + Bn), \qquad F \approx 2Bdn, \qquad I_{\text{ideal}} = \frac{2Bdn}{s(Bd + dn + Bn)}. \tag{28.4}$$

该流量是特定冷数据边界下的理想值；中间块重复读取会增加 Q，权重已经在缓存中则会改变该边界. 假设 $d = n$ 且 d 远大于 B，权重项主导，得到

$$I_{\text{ideal}} \approx \frac{2B}{s}. \tag{28.5}$$

这解释了小批量矩阵向量乘法与大批量矩阵乘法为何可能受到不同约束：同一份权重被更多行复用，单位搬运字节支持更多运算.

取 $d = n = 4096$、$s = 2$，当 $B = 1$ 时，

$$I_{\text{ideal}} = \frac{2 \cdot 4096^2}{2(4096^2 + 8192)} \approx 0.9995. \tag{28.6}$$

当 $B = 64$ 时，

$$I_{\text{ideal}} = \frac{128 \cdot 4096^2}{2(4096^2 + 128 \cdot 4096)} = \frac{64}{1 + \frac{1}{32}} \approx 62.06. \tag{28.7}$$

设一台抽象设备对该精度的峰值为 100×10^{12} FLOP/s，有效 HBM 带宽为 2×10^{12} byte/s，则 $I_* = 50$ FLOP/byte. 第一个形状的上界约为 2 TFLOP/s，第二个形状的

[1] Roofline 最初用于解释多核架构性能，随后扩展到多个存储层次及不同指令类型. 这里采用两项上界的基本形式，便于明确计算与数据搬运之间的关系.

上界达到计算平台 100 TFLOP/s. 这并未证明批量 64 能取得该性能，只说明在理想复用条件下，它已不再受到同一带宽上界限制.

自回归解码还读取随上下文增长的键值缓存，因此不能只按权重项计算强度. 预填充常有更大的矩阵行维度，但长上下文注意力、掩码和实现方式也会改变流量. 将所有预填充一概称为计算受限、所有解码一概称为带宽受限，均超出了上述推导的前提.

28.4　主机及设备亲和性

28.4.1　CPU 及 NUMA

中央处理器（Central Processing Unit，CPU）承担解压、分词、样本整理、调度和网络控制等工作. CPU 核数并不能独立决定数据准备速度：线程可能竞争内存带宽，解压可能受单线程路径约束，过多加载进程还会增加上下文切换和缓存失效.

非一致内存访问（Non-Uniform Memory Access，NUMA）表示不同处理器访问不同内存节点具有不同距离与代价. 一个工作进程即使绑定在靠近 GPU 的 CPU 上，其数据页仍可能位于另一 NUMA 节点，跨插槽访问因此仍会发生. 处理器绑定、内存分配策略和已有页面位置必须同时考虑；改变分配策略也不等于自动迁移全部已有页面. [147]

高速串行计算机扩展总线（Peripheral Component Interconnect Express，PCIe）连接设备和主机. 拓扑中的根复合体、交换芯片和上行链路决定哪些设备共享带宽. 两个各自具有高标称带宽的设备，可能共同经过一条较窄上行链路；GPU、网卡和本地盘的“总数量”因此不能替代拓扑图.

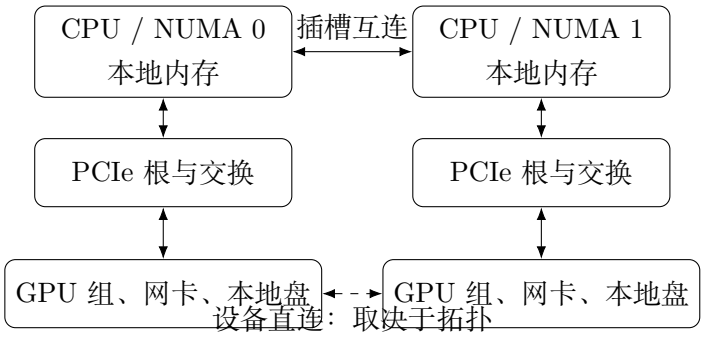


图 28.3: 双 NUMA 节点的设备亲和性示意. 相同设备数量可能形成不同的数据路径.

设备亲和性（Device Affinity）要求把频繁交换数据的进程、内存、GPU 和网络接口放在合适路径上. 设备到设备的直接访问也需要硬件、地址映射、驱动和权限共同支持，不能从“同一台服务器”推定所有设备能够直接互访. GPU 与网卡之间的直接数据通路可以省去主机中转，但仍需建立映射、同步完成事件并遵守设备访问顺序. GPUDirect RDMA 的官方说明明确指出其平台和 PCIe 拓扑限制. [148]

28.4.2 数据加载及重叠

页锁定内存（Pinned Memory）固定主机页的位置，便于设备通过**直接内存访问**（Direct Memory Access，DMA）传输数据. 它可以支持适当条件下的异步拷贝，但并不自动让拷贝与计算重叠. 程序还必须建立不同批次之间的流水，并确保当前计算使用的数据已经完成传输.

若读取、CPU 整理、主机到设备传输和计算分别耗时 t_r, t_p, t_h, t_g，完全串行处理一批需要

$$t_{\text{serial}} = t_r + t_p + t_h + t_g. \tag{28.8}$$

在资源足够独立、缓冲充分且批次间无额外依赖时，稳定流水的批间隔至少为

$$t_{\text{pipe}} \geqslant \max(t_r, t_p, t_h, t_g). \tag{28.9}$$

首批仍须经过完整路径，末批仍存在排空时间. 如果读盘与网络共同竞争 PCIe 上行，或者拷贝与算子共同竞争 HBM，独立性假设不成立，不能直接取最大值作为实际耗时.

设数据消费速度为 r_b 批/s，希望吸收最长 J 秒的短暂供给抖动，缓冲至少需要约 $r_b J$ 批. 缓冲只吸收短时波动：当生产速率长期低于消费速率时，任何有限队列最终都会耗尽. 过深预取也可能占用大量内存，并在取消、抢占或数据顺序变化时增加无效工作.

算法 28.1 有界预取与双缓冲

输入：固定版本的数据分片、批次变换、缓冲上限 K. 输出：按规定顺序交付设备的批次. 状态：有界主机队列、两个设备缓冲区及各自的完成事件.

1. 加载进程读取下一批，完成校验与 CPU 变换；队列满时等待，不继续无限预取.
2. 传输进程等待空闲设备缓冲，将下一批从页锁定主机区异步传入，记录传输完成事件.
3. 计算进程等待该事件后使用缓冲；计算结束记录消费完成事件，之后才允许覆盖.
4. 在另一缓冲传输后继批次，使可独立的传输与当前计算重叠. 源数据耗尽后排空队列；取消或错误时传播终止状态并回收未消费缓冲.

不变量：缓冲被消费前必须传输完成；计算完成前不可复用；队列长度不超过 K；失败批次不可静默跳过而改变训练样本序列.

28.5　互连及集群通信

28.5.1　节点内及节点间网络

节点内可以使用 PCIe 或专用加速器互连，例如 NVLink 体系中的链路与交换结构．专用互连的意义在于改变设备之间的可达路径、带宽和延迟，而不是把多卡变成无成本共享的一块显存．远端访问仍需要传输，某些共享地址空间也不意味着访问延迟一致．

节点间网络通常由网卡、接入交换和汇聚交换构成．**InfiniBand 网络**（InfiniBand，IB）和以太网是不同的网络体系；**融合以太网远程直接内存访问**（RDMA over Converged Ethernet，RoCE）在以太网络上提供 RDMA 传输．**远程直接内存访问**（Remote Direct Memory Access，RDMA）允许设备在预先授权和注册的内存区域之间传输数据，减少常规数据路径上的 CPU 参与，但连接管理、内存注册、完成处理与错误恢复仍然存在．

链路带宽决定持续搬运能力，延迟决定小消息启动成本．**超售比**（Oversubscription Ratio）可在指定交换层定义为下行总带宽与上行总带宽之比；大于 1 意味着所有下行不能同时按线速通过上行．**割带宽**（Bisection Bandwidth）衡量将网络分成两组时跨割能够通过的带宽．讨论双向割带宽时必须说明是两个方向的和还是每方向值，不能用全网端口带宽总和代替跨组吞吐．

相同端口速率下，邻近设备通信、跨交换层通信与全体同时交换可能表现完全不同．专家路由还可能产生多对一突发：平均字节率低于链路容量，并不能排除某一接收队列瞬时溢出．拓扑、路由、消息分布和其他租户负载因此属于通信模型的输入．

28.5.2　拥塞控制及故障域

显式拥塞通知（Explicit Congestion Notification，ECN）通过网络标记让端点感知拥塞并调整发送．**基于优先级的流量控制**（Priority-based Flow Control，PFC）可对指定优先级暂停上游发送．二者职责不同：前者试图调整负载，后者在链路上施加反压．暂停可能把拥塞向上游传播，并影响共享优先级的其他流．RoCE 部署应结合网卡能力、交换缓冲、端到端拥塞算法和具体拓扑设计，不能将“开启 PFC”视为完整方案．是否采用无损配置及其参数应以锁定环境的官方资料为准．[149]

故障域是一次失效能够共同影响的资源集合．一个网络端口、交换机、机架供电或共享存储都可能构成不同故障域．把一个同步训练作业跨越更多设备，可提高资源规模，也扩大了任何单点失效中断整个作业的范围．在线推理的多个完整副本则可以分布于不同故障域，使单组故障不同时影响全部流量；单个张量并行组仍需要组内协作完成请求．

28.5.3　集合通信的时间模型

集合通信（Collective Communication）是一个进程组共同完成的数据交换操作. **全归约**（All-Reduce）让所有参与者得到逐元素归约结果；**归约散播**（Reduce-Scatter）将归约结果分片交给各参与者；**全收集**（All-Gather）将各自分片组成完整结果；**全互换**（All-to-All）允许每个参与者向其他参与者发送不同分片. 这些是数据语义，不指定唯一网络算法. 参与进程的调用次序、数据类型与长度必须满足通信接口约定，否则可能发生挂起或错误. [150]

先考虑一条有效链路传输 m 字节. 将不可随消息长度摊薄的启动成本记为 α，持续传输每字节成本记为 β_{inv}，得到

$$t_{\mathrm{msg}}(m) \approx \alpha + m\beta_{\mathrm{inv}} = \alpha + \frac{m}{\beta_{\mathrm{net}}}. \tag{28.10}$$

小消息满足 $m \ll \alpha\beta_{\mathrm{net}}$ 时受延迟主导，大消息反之. 该线性模型不包含排队和拥塞，参数须从相同路径与消息范围估计，不能把空闲网络的带宽直接用于拥塞时段.

对 p 个进程的环形全归约，每个进程初始有 m 字节，把消息均分为 p 块. 归约散播经过 $p-1$ 轮，每轮发送约 $\dfrac{m}{p}$ 字节；全收集再经过 $p-1$ 轮. 若各链路同时工作，忽略本地归约与拥塞成本，则

$$t_{\mathrm{ring}} \approx 2(p-1)\left(\alpha + \frac{m}{p}\beta_{\mathrm{inv}}\right) \tag{28.11}$$

$$= 2(p-1)\alpha + \frac{2(p-1)}{p} m\beta_{\mathrm{inv}}. \tag{28.12}$$

式中第二项对应每进程发送量，不是全网总字节数. 若归约每字节消耗 γ 秒，还可加入约 $\dfrac{(p-1)m\gamma}{p}$，但计算与网络是否重叠需要进一步分析.

作为对照，考虑不分块的二叉树归约加广播，且 p 为 2 的幂. 每阶段高度为 $\log_2 p$，每层在关键路径上传送完整消息，粗略成本为

$$t_{\mathrm{tree}} \approx 2\log_2 p(\alpha + m\beta_{\mathrm{inv}}). \tag{28.13}$$

它的启动轮数更少，完整消息传输项却可能更大. 分块树、双树与分层实现会改变该模型；这里的比较只能用于解释为何小消息与大消息可能选择不同算法，不能据此宣布树或环普遍更快.

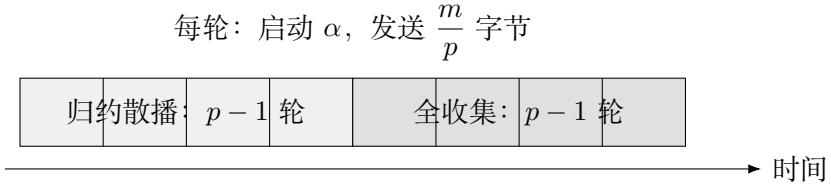


图 28.4: 环形全归约的两阶段时间结构. 格数仅用于示意，轮数由参与进程数决定.

例如令 $p=8$、$m=256$ MiB、$\alpha=5$ 微秒、有效单向带宽 25 GiB/s. 每轮数据为 32 MiB，传输时间为 $\dfrac{32}{25\cdot 1024}=0.00125$ s. 于是

$$t_{\text{ring}} \approx 14(0.000005+0.00125)=0.01757 \text{ s}. \tag{28.14}$$

其中启动只占 70 微秒. 若消息缩为 8 KiB，则每轮仅 1 KiB，全部传输项约 0.534 微秒，70 微秒启动项成为主导. 这是同一参数模型内的构造比较；真实小消息协议可能具有不同参数.

把通信隐藏在反向计算后面，只能隐藏数据已就绪且不阻塞后继计算的部分. 若计算耗时 t_c、通信耗时 t_n、实际重叠时间为 t_o，则

$$t_{\text{step}}=t_c+t_n-t_o, \qquad 0\leqslant t_o\leqslant \min(t_c,t_n). \tag{28.15}$$

最后一个梯度分片的通信往往处于关键路径，不能被已经结束的计算隐藏. 通信内核还可能占用计算资源或显存带宽，因此重叠本身也可能改变 t_c 与 t_n.

28.6　存储系统及数据通路

28.6.1　容量、带宽、请求率及元数据

非易失性存储器快速接口（Non-Volatile Memory Express，NVMe）为高速非易失存储访问定义接口体系. [151] 本地 NVMe 存储常用于数据缓存、权重预热和暂存；它靠近计算节点，但节点失效后未必仍可访问. 分布式文件系统提供路径与文件操作语义，便于共享数据；对象存储以键和对象为基本操作单位，适合不可变数据分片与模型制品. 不同实现的并发、缓存和一致性条件不同，不能把三者仅按容量大小排序.

每秒输入输出操作数（Input/Output Operations Per Second，IOPS）衡量完成请求的速率，带宽衡量完成字节的速率. 若每请求有效负载为 q 字节，设备持续处理能力为 a 次/s，则

$$\beta_{\text{effective}} \leqslant \min(\beta_{\text{io}}, aq). \tag{28.16}$$

例如，吞吐上限为 2 GiB/s 的路径，若只能完成每秒 10000 次 4 KiB 读取，有效吞吐上限为约 39.06 MiB/s. 这里的两个数字是设定条件，说明高顺序带宽不能保证高随机小读吞吐.

元数据（Metadata）包括名称、目录项、权限、大小和分片索引等描述信息. 若每个训练样本都需要打开一个文件，即使样本字节很少，目录查找与打开操作仍可能首先饱和. 将样本组织成可索引的大分片，可以摊薄请求和元数据成本，但也会改变随机访问、恢复位置与负载均衡方式. 全局随机性不必通过“每个样本一次完全随机远程读”实现，可在分片调度与分片内缓冲之间建立可复现的打乱策略.

队列深度也影响并发. 若平均完成延迟为 l 秒，稳定请求率为 a 次/s，则平均在途请求数近似为 al. 为了取得更高吞吐，需要足够在途工作，但无限增大队列会增加

尾延迟和内存占用. 对同步训练，某个节点数据读取的长尾可能让整个进程组在下一次集合通信处等待，所以平均吞吐之外还需关注高分位延迟.

28.6.2 数据集、权重和检查点的不同负载

训练数据通常持续读取，权重分发往往集中在启动时，检查点则周期性产生大写入. 三种流量共享存储或网络时，不能只计算各自平均值. 权重加载还可能形成**启动风暴**（Startup Storm）：大量实例同时请求相同制品，把对象存储出口、节点网络或解压 CPU 推到极限.

分层缓存将权威不可变制品与节点热副本分开. 模型文件先按版本和内容摘要确认，再进行有界并发分发、本地预热及设备加载. 只有校验、加载与必要初始化完成，实例才能进入就绪状态. 缓存命中降低远程读取，但不会消除本地盘到主机、主机到设备的传输时间；预先加载也会消耗容量和带宽，因此需要与任务计划协调.

检查点包含恢复训练所需的模型、优化器、调度器、随机状态和数据位置等信息. 保存权重文件不一定等于保存可恢复的训练状态. 对于分片训练，各文件必须对应同一个逻辑步骤；部分进程写入新步骤、部分仍停在旧步骤，会生成彼此不一致的状态集合.

一致性（Consistency）说明读写的可见性与顺序，**持久性**（Durability）说明确认数据能承受哪些失效，完整性校验则用于发现内容损坏. 这些保证彼此不能替代. 以 Amazon S3 文档为例，单键更新的原子性与强读后写一致性并不提供跨多个键的原子事务. [152] 因此，多分片检查点仍需应用层提交协议，不能因底层对象写入成功就认为整个检查点已经提交.

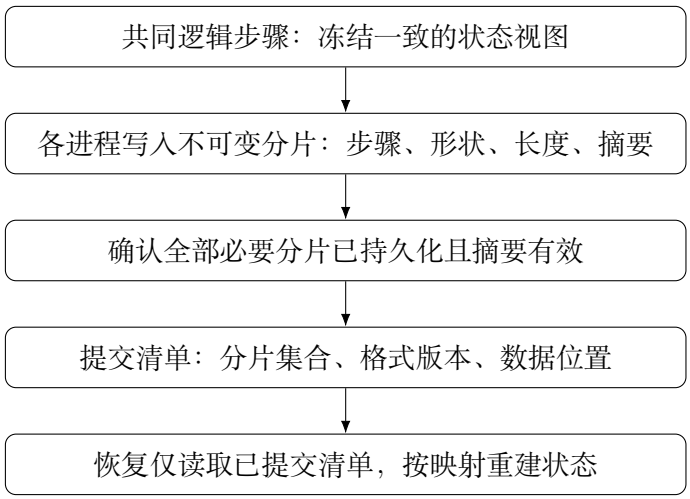


图 28.5: 分片检查点的数据与提交路径. 未提交分片可以回收，不可直接用于恢复.

异步检查点减少训练等待的前提，是先获得稳定快照，再在后台传输. 若后台读取的张量仍被优化器修改，得到的可能是不同时间状态的混合. 快照复制、写时复制或等价的一致性机制都会消耗内存和带宽. 暂存到本地盘可以缩短前台暂停，但在远端持久化之前，不能宣称具有远端存储对应的容灾能力.

28.6.3　存储吞吐量测算

例 27.2

考虑 8 个节点、每节点 8 个加速器的训练作业，共 64 个设备．设每设备有效处理 20000 词元/s，每词元编号 4 字节．为隔离不同成本，先只计算已经分词后的编号读取，不包含掩码、索引和原始多模态数据．总词元率为

$$r_t = 64 \times 20000 = 1.28 \times 10^6 \text{ token/s}, \tag{28.17}$$

对应最小编号读取带宽

$$\beta_{\text{token}} = 4r_t = 5.12 \times 10^6 \text{ byte/s} \approx 4.883 \text{ MiB/s}. \tag{28.18}$$

如果每次远程请求仅获取 4096 个编号，则有效负载 16 KiB，理想请求率为 $\dfrac{1.28 \times 10^6}{4096} = 312.5$ 次/s．改为 64 MiB 的连续分片读取时，按相同长期字节率平均约每 13.11 秒读取一个分片．实际 8 节点并发、预取与缓存使短时请求率不同，但可以看到请求组织与有效字节率是独立决策．

进一步设训练所需全局检查点为 2 TiB，每 30 分钟产生一次，要求 120 秒内完成持久化．它的长期平均写带宽为

$$\overline{\beta}_c = \frac{2048}{1800} \approx 1.138 \text{ GiB/s}, \tag{28.19}$$

而写入窗口内所需带宽为

$$\beta_c = \frac{2048}{120} \approx 17.067 \text{ GiB/s}. \tag{28.20}$$

两者相差 15 倍. 均匀分片时每节点写 256 GiB, 窗口内每节点至少需要约 2.133 GiB/s；若 64 个进程均匀分担，每进程为 32 GiB．以上只计算有效状态字节，不含重试、校验和冗余编码在后端产生的额外流量．

假设所选共享存储路径在相应读写混合条件下的总有效能力被设定为 40 GiB/s，其他作业在该窗口已经占用 24 GiB/s，剩余能力只有 16 GiB/s．即使检查点长期平均带宽远小于剩余能力，120 秒目标仍不能满足：

$$t_{c,\min} = \frac{2048}{16} = 128 \text{ s} > 120 \text{ s}. \tag{28.21}$$

训练编号读取约占 0.00477 GiB/s，计入后下界还会略高；元数据和提交时间也尚未计入．可行调整包括错开共享写入窗口、为检查点预留更高带宽，或者在恢复目标允许时放宽窗口．仅扩大预取缓冲无法改变这项持续传输下界．

如果每个存储请求携带 64 MiB，17.067 GiB/s 约对应 273.07 次/s；如果拆成 1 MiB，则约需 17476.27 次/s．单次检查点的数据对象数量分别为 32768 与 2097152.

后者还显著增加清单、校验和元数据操作数量. 较大对象可以摊薄请求成本，但过大的不可分割请求也可能增加重试成本，必须保留分块校验与可恢复传输能力.

最后考察恢复. 若同一 2 TiB 检查点的存储读取能力为 32 GiB/s，恢复网络路径为 16 GiB/s，聚合设备加载能力为 24 GiB/s，理想充分流水的时间下界为

$$t_{\mathrm{restore}} \geqslant \max\left(\frac{2048}{32}, \frac{2048}{16}, \frac{2048}{24}\right) = 128 \text{ s}. \tag{28.22}$$

若三个阶段必须完全串行，则仅传输约需 $64 + 128 + 85.33 = 277.33$ 秒. 真实恢复还包括资源重新分配、进程组建立、反序列化、状态映射和校验，故不能直接把 128 秒作为恢复承诺. 这一算例表明，训练数据读取、检查点窗口与故障恢复可能对应完全不同的主瓶颈.

28.7　训练及推理集群的组织

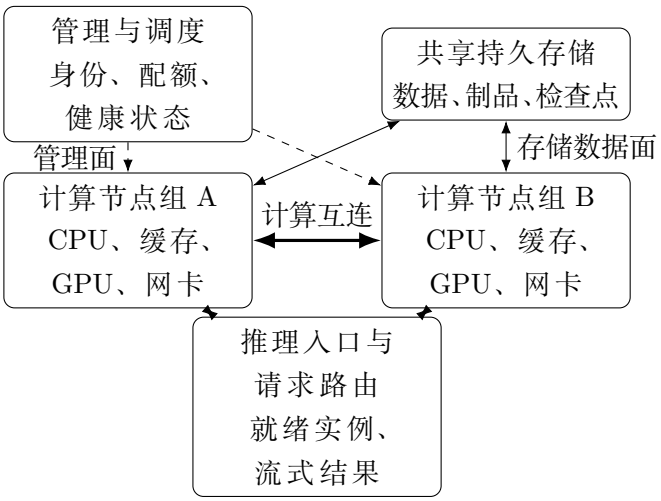


图 28.6: 集群的逻辑网络与数据通路. 管理、计算、存储和服务流量可以物理分离，也可以在共享网络上实施资源与权限隔离.

训练集群中的同步组要求多进程共同前进，数据供给和检查点也与训练步骤相关；推理集群需要进一步处理请求到达、排队、取消、输出长度差异和流式返回. 两者可以共享硬件资源池，但不应因此共享无约束的调度和带宽策略. 长时间训练吞吐目标与在线服务尾延迟目标可能相互冲突.

控制面（Control Plane）负责准入、版本、资源分配和恢复决策，**数据面**（Data Plane）执行模型计算与数据传输. 管理网络故障可能使调度信息暂时不可用，而计算网络故障可能直接打断集合通信；共享同一物理网络时，应识别这种故障相关性. 分离逻辑职责不一定要求每类流量使用独立硬件，但必须有容量与隔离依据.

推理状态还具有不同的寿命. 权重通常按模型版本共享，键值缓存属于具体请求或可严格标识的前缀. **键值缓存**（Key-Value Cache，KV Cache）外置到主机、磁盘

或远端存储后，容量增加但访问路径变长．若迁移 S_{KV} 字节，单纯传输下界为 $\frac{S_{\mathrm{KV}}}{\beta_{\mathrm{net}}}$，还需加排队、序列化和同步．只有节省的重算或排队成本超过新增成本，迁移才可能改善服务目标．

预填充与解码分离（Prefill/Decode Disaggregation，P/D 分离）把前缀计算与逐词元生成分配到不同资源池，可以隔离长提示干扰，但同时增加 KV 传输和跨池协调．分页缓存减少分配碎片，不消除有效 KV 字节；跨实例共享也必须绑定模型、适配器、位置配置、精度、完整前缀与租户权限．基础设施不能仅凭“缓存命中”绕过这些语义条件．

28.8　资源调度、兼容性及隔离

28.8.1　运行兼容矩阵

加速器程序依赖设备指令集、固件、驱动、运行时、通信库、编译器、框架和算子库的共同支持．容器封装用户态依赖，却不自动替换主机驱动，也不保证目标设备具有对应指令．一个模型能够导入框架，并不证明其低精度矩阵核、注意力后端和跨卡通信均已可用．

层次	必须明确的条件	不满足时的后果
设备与固件	指令、精度、内存、互连能力	运算不支持或资源不足
驱动与运行时	锁定版本及相容范围	初始化失败或路径不可用
通信库与网卡	传输后端、拓扑、权限、协议	回退到较慢路径或挂起
框架与算子	形状、布局、精度、掩码支持	图断裂、重编译或低效回退
模型制品	分片布局、量化元数据、版本	加载失败或数值语义改变

国产与国外加速器的迁移都应按照这张依赖关系表进行．源代码层可移植，不等于执行内核和通信库可直接复用；仅替换设备字符串尤其不能证明等价．迁移成本来自算子覆盖、编译后端、数值格式、分布式通信、诊断工具和维护周期．应先建立模型所需运算及张量形状清单，再逐项确认正确性和目标路径性能，而不是按品牌推断成熟度．版本升级应形成新的兼容矩阵与回退制品，避免一次性更换所有层后无法定位回归来源．

28.8.2　国产加速软件栈

以 Ascend Extension for PyTorch 官方历史标签 `v2.5.1-7.1.0` 的兼容矩阵为例，其中一行将 PyTorch 2.5.1、扩展 `torch_npu` 2.5.1.post1 与 CANN 8.2.RC1 对应起来[153]．这是一组可核对的历史版本关系，不代表任意新旧版本可以混装，也不是对当前部署版本的推荐．

这里 PyTorch 表达模型与自动微分图，设备扩展承接设备与算子调用，CANN 提供底层加速软件能力；分布式运行还必须确认适用的集合通信后端. 迁移检查应沿这条调用链展开：框架成功导入，只能证明基础加载路径；目标数据类型与形状下的前向、反向和掩码行为正确，才支持算子等价；实际张量并行配置下的集合通信完成，才支持协同执行. 出现未覆盖算子、主机回退或隐式格式转换时，单算子正确也可能伴随严重吞吐损失. 兼容矩阵因而应记录设备型号、驱动、扩展、底层运行时、通信后端及模型运算范围，并分别记录正确性与性能证据.

28.8.3 成组调度及拓扑放置

成组调度（Gang Scheduling）要求协作任务达到共同启动的最小资源条件后才分配运行，避免部分进程长期占有设备却只能等待其他进程. 固定规模同步训练通常要求整个进程组齐备；允许弹性的算法可以定义其他最小规模，但改变成员后还需重新建立状态与通信关系. Volcano 的 Gang 插件以作业最小可用任务数表达这种准入约束. [154]

调度对象应是资源向量和拓扑约束，而不只是 GPU 数. 一个作业同时要求设备容量、CPU、主机内存、网络路径、临时存储和可用软件栈. 即使全局空闲设备数足够，设备分散在不满足互连要求的节点上，也可能无法满足性能目标. 异构设备共同执行同步任务时，最慢参与者与最弱通信路径可能决定步骤时间；异构分配应与任务分解方式相匹配.

抢占（Preemption）把资源从低优先级作业转交给高优先级作业，但其成本包括检查点保存、后续恢复和丢失计算. 调度器若频繁抢占同一长作业，设备忙碌时间很高，有效训练进度却可能很低. 配额约束资源份额，拓扑放置控制物理路径，抢占策略控制资源转移，三者应共同设计.

算法 28.2 协作作业的资源准入

输入：资源向量、协作规模、兼容矩阵、优先级与恢复策略. 输出：可启动的放置方案，或保留在等待队列中的作业.

1. 过滤不兼容、不健康或无租户访问权限的节点.
2. 在剩余节点中搜索满足设备、主机内存、网络与临时存储约束的完整资源组，并估计跨节点关键路径.
3. 检查配额和优先级；若必须抢占，确认被抢占作业的保存窗口与恢复成本符合策略.
4. 以组为单位预留资源，完成进程启动与通信就绪确认. 部分启动失败时释放本次预留并报告原因，不能无限占有剩余设备.

5. 作业运行期间观察进度与健康；完成时释放资源，失效时按通信组边界终止或进入已定义的弹性恢复流程.

不变量：未经完整准入的资源组不进入协作计算；预留具有超时；恢复不会把旧成员的完成消息误认为新进程组的状态.

28.8.4　租户隔离及观测

多租户隔离（Multi-tenant Isolation）同时包含身份权限、数据边界和资源干扰控制. 访问控制决定租户能读写哪些模型、数据和检查点；资源配额约束容量；调度与限流约束带宽和执行份额. 共享缓存、日志、设备内存和性能指标还可能跨越数据边界，因此隔离不能只依赖容器名称.

整卡分配、硬件资源分区和时间共享具有不同的容量与干扰特征. 可用的硬件分区能力必须按设备和运行时确认；时间共享通常不能提供等同于独占硬件的尾延迟隔离. 观测需要将作业进度与设备、CPU、网络队列、存储请求和错误事件关联起来，并保留租户范围. 单一“GPU 利用率”指标不能说明设备是在做有效计算、等待通信，还是频繁执行短小内核.

28.9　可靠性、供电及散热

28.9.1　作业故障传播

设备失效、网络分区和存储不可用可能通过同步关系放大. 一个参与进程停止，其他进程可能阻塞在下一次集合通信；如果应用只重试其中一个进程，却没有重建共同步骤与进程组，旧请求与新请求可能发生次序不一致. 故障处理必须区分局部可重试操作与需要整组恢复的状态变化.

检查点恢复也需要处理格式和并行布局. 如果恢复设备数不同，原分片可能需要重排；模型权重能够重分片，不代表优化器状态、随机状态和数据位置已经正确恢复. 可恢复性应通过完整恢复路径定义，而不是通过“文件写入成功”定义. [2]

28.9.2　检查点间隔的简化推导

假设作业故障近似服从均匀稳定的泊松过程，平均故障间隔为 M；每进行 τ 秒有效计算暂停 C 秒保存检查点，恢复耗时 R_f. 若 C, R_f, τ 相对 M 较小，忽略保存与恢复期间再故障等高阶项，每单位有效时间的损失比例近似为

$$w(\tau) \approx \frac{C}{\tau} + \frac{\tau}{2M} + \frac{R_f}{M}. \tag{28.23}$$

[2]保存更多副本能提高某些故障下的可用性，但若副本共享同一机架供电、同一元数据服务或同一误删除权限，它们并不是相互独立的故障保护.

第一项是周期性暂停；故障在两次保存之间近似均匀出现，平均丢失 $\frac{\tau}{2}$ 的计算，给出第二项；第三项是故障恢复本身．对 τ 求导，

$$\frac{\mathrm{d}w}{\mathrm{d}\tau} = -\frac{C}{\tau^2} + \frac{1}{2M} = 0, \qquad \tau_* = \sqrt{2CM}. \tag{28.24}$$

二阶导数 $\frac{2C}{\tau^3} > 0$，故为该近似模型中的极小值．它说明保存越昂贵，适宜间隔越长；作业越容易故障，适宜间隔越短．它不是复杂真实系统的通用精确公式．

例如，若暂停保存需要 60 秒，作业平均故障间隔为 24 小时，则

$$\tau_* = \sqrt{2 \cdot 60 \cdot 86400} \approx 3220 \text{ s}, \tag{28.25}$$

约 53.7 分钟．在该间隔处，保存损失与丢失计算两项分别约为 1.86%．恢复项仍需另算．若检查点异步写入，C 应表示前台有效暂停，并另分析后台资源竞争；不能把整个远端传输时间机械地代入暂停项．

若 p 个节点独立且具有相同失效率 λ，任何一个失效都会使作业停止，则作业失效率为 $p\lambda$，平均故障间隔为 $\frac{1}{p\lambda}$．共享供电、网络或软件缺陷会破坏独立性假设，因此不能从单机平均间隔简单外推所有集群．维护窗口、预告式抢占与突发相关故障也需要不同的保存策略．

28.9.3　功率、温度及有效工作

机架的供电与散热容量限制可持续部署密度．加速器峰值功率之和之外，还包括 CPU、内存、网卡、存储、交换设备和电源转换损失．若热量不能及时带走，设备可能降低频率，可持续操作率因此低于短时峰值．功率封顶也会同时改变能耗和完成时间，不能只比较瞬时瓦数．

设系统运行时间为 T、实际功率为 $P(t)$，总用电量为

$$E = \int_0^T P(t)\ \mathrm{d}t. \tag{28.26}$$

电能使用效率（Power Usage Effectiveness，PUE）是设施总能耗与 IT 设备能耗之比．[155] 使用同一统计周期的 PUE 可估计冷却等设施开销，但不能直接表示模型能效．任务能效更应报告完成每单位有效训练词元或合格输出所消耗的能量，并包含失败、重算和空闲分摊．

总拥有成本（Total Cost of Ownership，TCO）包含资源获得、能源、网络存储、运维和停机等成本．对固定任务，单位有效工作成本可以写成总分摊成本除以合格完成量．利用率提升通常改善固定成本摊薄，但若它来自过度共享、提高错误率或增加重算，分母不会按设备忙碌时间同比增长．敏感性分析应分别改变有效吞吐、能源价格、恢复频率和资源占用时间，避免将某个采购单价作为系统经济性的唯一指标．

备用资源、维护窗口和故障演练也是容量设计的一部分. 没有恢复位置的备用检查点，或者没有可用设备的快速重启机制，都不能独立满足恢复目标. 供电、散热、网络和存储的冗余应按共同故障域分析，而不是逐项计数后假定整体已经冗余.

28.9.4　同一集群的期限、容量及费用

例 27.3

继续使用存储算例的 8 节点、每节点 8 设备. 设目标为 1.28×10^{12} 个有效词元，每设备端到端有效速率为每秒 20000 词元，则 64 设备的完成时间为

$$T = \frac{1.28 \times 10^{12}}{64 \times 20000} = 10^{6}\ \mathrm{s} = 277.78\ \mathrm{h}. \tag{28.27}$$

若期限为 12 天，连续设备数下界为 61.73，向上取整为 62，再按完整 8 设备节点放置得到 64. 这里有效速率假定已计入既定通信、检查点及故障损失，不能再次乘同一损失因子. 该速率假设的可行性仍受前述存储约束：必须先修复 120 秒检查点窗口；若为检查点单独保障 20 GiB/s，2 TiB 纯写入时间为 102.4 秒，其余约 17.6 秒供提交等开销，训练读取另行保障. 否则不能将不满足窗口的系统视为已达到假定速率.

假设每节点实际运行功率 8 kW，网络和存储合计 20 kW，则 IT 功率为 84 kW；PUE 为 1.2 时设施功率为 100.8 kW，运行耗电为 28000 kWh. 电价按每千瓦时 0.8 个货币单位计算，电费 22400；设备折旧及维护若按每设备小时 5 个货币单位分摊，则为 88888.89，二者合计 111288.89. 这里没有另计融资、闲置期与人员费用，故不能称为采购总成本. 若有效速率降至 16000，运行时间增至 347.22 小时，64 设备不再满足期限；期限约束给出 78 设备，按节点取整需 80 设备. 新增节点也会改变网络、功率与存储负载，不能只修改分母.

同一物理集群用于服务时需重新给定服务阶段证据. 例如构造 8 个各占 8 设备的副本，假设每副本稳定输出 800 词元/秒、平均回答 400 词元，算术吞吐上界为每副本 2 请求/秒. 目标 12 请求/秒且允许负载率不超过 0.75 时，需要 $\lceil \frac{12}{2 \times 0.75} \rceil = 8$ 个副本. 若每副本可用于缓存的容量为 20 GiB、每活动请求的缓存上界为 0.5 GiB，容量另限制为 40 条活动请求. 以上速率、长度和缓存均为构造输入；它们不能由训练速率推得，也不单独保证尾时延. 排队与调度的联合约束见第三十一章《批处理调度》中的推理调度与容量管理.

28.10　系统性能诊断

测量应按层次回答不同问题. 峰值规格界定硬件上限；算子基准说明特定形状、精度和后端的执行能力；端到端训练或推理结果才反映数据准备、通信、排队与恢复等共同成本. 三种结果可以相互解释，不能相互替代.

对同一个工作负载，首先固定模型、输入长度、有效批量、精度、并行规模和质量要求，再观察计算内核、HBM 流量、主机准备、网络通信、存储读写和空闲时间. 若算子本身接近 Roofline 上界而步骤仍慢，应继续查找算子之间的同步与数据供给；若网络小消息时间占比高，增加链路带宽未必有明显收益；若仅在保存检查点时波动，则应检查窗口资源竞争和快照成本.

瓶颈是路径约束

一次任务能够达到的吞吐受到其必要数据路径共同约束. 容量决定状态能否驻留，带宽决定字节能否及时到达，延迟决定细粒度依赖何时推进，可靠性决定已经完成的工作能否保留. 只有明确工作量、路径和时间窗口，增加资源才有可推导的作用对象.

推理测量还需区分**首词元时延**（Time to First Token，TTFT）和**每输出词元时间**（Time per Output Token，TPOT），声明是否包含请求排队、分词和网络返回. 冷启动、首次编译与稳定执行应分别记录. 缓存命中率需结合节省的有效工作量，吞吐需结合错误率和尾延迟. 训练则应报告有效词元率、步骤分布、通信暴露时间、检查点暂停与恢复损失，不能用补齐词元或反复重算增加的操作量冒充训练进度.

一次可解释的记录至少包括设备与节点拓扑、驱动和库版本、实际算子后端、精度与累加规则、批量和长度分布、数据路径、并行配置、冷热状态与同时运行的负载. 容量公式和状态机可以说明机制；只有在目标环境中取得相应测量，才能把结论提升为该配置的性能证据.

扩展阅读

- Wafer AI, *GPU Performance Engineering Resources*：按 GPU 基础、CUDA 与 PTX、内核优化、性能分析、推理引擎和分布式推理组织的学习资源目录；适合在掌握本章的性能模型后，按实际优化任务继续深入.

第二十九章　分布式训练

模型规模扩大后，训练不再只是选择更新公式的问题：一次前向所需的激活可能超过显存，优化器状态可能无法在单设备常驻，多设备之间的通信又可能抵消并行计算的收益．分布式训练把同一逻辑计算分配到多个执行单元；它首先需要解释计算结果为何仍对应原来的目标，然后才讨论速度和容量．

本章沿全局目标归一化、集合通信、状态分片与模型并行展开，适用于预训练和微调．第二十二章《训练工程优化》讨论微调中的资源取舍与并行方案选用；本章解释跨设备计算如何保持目标、参数更新和恢复状态的一致性．第二十七章《模型规模及资源预算》提供内存和计算量的计数方法，第二十八章《算力基础设施》解释互连与存储，第十九章《训练稳定性》规定一次更新的数值语义．

29.1　不变的目标及可变的执行方式

29.1.1　三个不同的词元计数

训练中至少需要区分：输入的非填充词元数、实际进入算子的词元位置数，以及参与目标函数的有效监督词元数．提示词元可能不参与损失，却仍需前向计算；被因果掩码排除的注意力关系也未必被底层实现跳过．因此不能把“没有标签的位置”一概视为可以删除的计算．

若批内非填充长度为 $\ell_1,\dots,\ell_B$，动态填充到 $\ell_{\max}$ 后，填充比例为

$$\rho_{\text{pad}} = 1 - \frac{\sum_{b=1}^{B} \ell_b}{B\ell_{\max}}. \tag{29.1}$$

这描述输入位置的浪费比例，不直接等于总 FLOPs 的浪费比例．稠密注意力的长度代价近似二次，而前馈层通常随位置数线性增长；不同内核还可能利用变长信息减少实际运算．

长度分桶（Length Bucketing）应与随机化和数据混合共同设计．多副本执行还要平衡各副本的计算负载：仅降低每个本地批次的填充比例，仍可能让持有长序列的副本成为同步等待的瓶颈．分桶的基本算例见第二十二章《训练工程优化》．

序列打包（Sequence Packing）减少短序列尾部空位，但必须保留任务规定的监督及上下文边界．若任务要求样本独立，需同时处理跨样本注意力、位置标识和首个

目标的条件范围；仅插入终止标记并不自动禁止读取前一个样本．数据构造的完整机制见第十八章《训练数据工程基础》．

29.1.2　符号及逻辑设备网格

本章主要符号见表 29.1.

表 29.1: 分布式训练的主要符号与逻辑设备量.

符号	含义
R	数据并行组中的副本数；不是所有场景下的总设备数.
G	一次更新内每个副本的微批次数.
S_{rj}, n_{rj}	副本 r 的微批次 j 的损失和与有效监督词元数.
N	一次全局更新的有效监督词元总数.
P_s, G_s, O_s	完整模型的参数、梯度、优化器状态所占字节数.
q, p	张量并行度与流水并行阶段数.
M, α, β	集合通信的逻辑消息字节数、单轮时延和每字节传输时间.
B_μ, T, d	微批量、序列长度和隐藏维数.

进程序号（Rank）是通信组内的逻辑成员标识．一个物理进程可以属于多个通信组，例如与相同模型分片的其他副本组成数据并行组，与同一层的其他分片组成张量并行组．全局进程序号不能代替这些不同的组内坐标.

29.2　梯度累积及全局归一化

29.2.1　有限和梯度推导

梯度累积（Gradient Accumulation）在参数保持不变的条件下，分多次前向和反向获得一次更新的梯度．设 n_{rj} 由固定标签掩码决定，不依赖参数，则

$$N = \sum_{r=1}^{R} \sum_{j=1}^{G} n_{rj}, \qquad \mathcal{L} = \frac{1}{N} \sum_{r,j} S_{rj}, \tag{29.2}$$

$$\nabla \mathcal{L} = \frac{1}{N} \sum_{r,j} \nabla S_{rj} = \sum_{r,j:n_{rj}>0} \frac{n_{rj}}{N} \nabla \overline{\mathcal{L}}_{rj}, \qquad \overline{\mathcal{L}}_{rj} = \frac{S_{rj}}{n_{rj}}. \tag{29.3}$$

该式来自微分的线性性，不要求各微批次长度相同．简单平均各批次平均损失，使用的是每批等权目标，只有计数相同或梯度发生特殊抵消时才与词元平均一致．$n_{rj} = 0$ 时其损失和与梯度贡献定义为零，不能计算局部平均.

微批次加权的基本算例见第二十二章《训练工程优化》；多副本执行还需让本地损失缩放与跨副本通信的求和或平均语义配合.

29.2.2　求和通信及平均通信

分布式数据并行（Distributed Data Parallel，DDP）令各副本处理不同数据，随后同步梯度．若通信后梯度为各副本梯度的平均，每个副本可反向传播

$$\mathcal{L}_r^{*} = \frac{R}{N}\sum_{j} S_{rj}, \qquad \frac{1}{R}\sum_{r}\nabla\mathcal{L}_r^{*} = \frac{1}{N}\sum_{r,j}\nabla S_{rj}. \tag{29.4}$$

若通信执行求和，则无需分子中的 R．选择哪一种表达取决于实际通信语义，而不是函数名称中的“归约”二字．

例如两副本分别拥有 2 和 6 个目标，损失和的梯度为 2 和 18．各自局部平均再跨副本平均得到 $\frac{1+3}{2}=2$；先乘 $\frac{R}{N}=\frac{1}{4}$ 得到 0.5 和 4.5，再平均才得到 2.5．全局 N 需要在相关数据并行组内归约；张量并行进程往往处理同一批词元，不能把它们再次计入总量．

可以先读取窗口内的标签计数，再依式 (29.4) 反向；也可以累积未归一化的损失和梯度，完成求和后除以 N．后者避免预先知道计数，但未归一化梯度更大，必须在精度设计中考虑溢出风险．裁剪应在最终归一化后执行，因为先裁剪局部梯度再求和通常改变方向．

大批次与累积的等价范围

所有微批次使用同一参数版本、相同样本及监督集合，梯度线性聚合正确，且中间不更新参数或矩状态．在精确算术和相同随机函数实现下，累积与完整批次梯度一致．若层依赖批内统计、随机掩码映射不同或归约顺序不同，不能据此保证逐位一致．

非最终微步可推迟梯度通信，以减少集合操作次数，但必须确保最终同步覆盖全部累计贡献．某些参数只在部分微批次或路由分支中参与计算时，通信调度还必须处理未使用参数和组间顺序一致性．没有本地有效目标的进程也不能自行退出其他成员仍需参与的集合操作．

29.3　混合精度及激活重计算

混合精度（Mixed Precision）将不同运算和状态置于适合的数值格式．它同时影响输入存储、乘法精度、累加精度、输出格式和优化器状态，单个“低精度训练”标签不足以描述这些选择．损失缩放与非有限值处理已在第十九章《训练稳定性》展开；分布式场景需要使同一更新组对跳步作出一致决定，防止各副本参数分叉．

激活重计算（Activation Recomputation）在前向时不保存部分中间结果，在反向需要时重新求值．早期系统工作给出了次线性激活存储的构造 [131]．以下采用均匀链式网络说明成本，不把近似式当作所有计算图的峰值保证．

设有 L 层，每层需保存一个大小为 A 字节的激活单位．每隔 k 层保存边界，反向时重建一个区间．边界约占 $\frac{AL}{k}$，当前区间约占 Ak，从而

$$M_{\text{act}}(k) \approx A(\frac{L}{k} + k). \tag{29.5}$$

该均匀模型的最优区间与计算量算例见第二十二章《训练工程优化》．分布式执行需要按本地模型分片与同时存活的微批次重新计数．流水并行会保留多个在途微批次的边界，张量或上下文切分改变本地激活形状；参数分片下，重算区间还可能再次触发参数收集．每个设备的容量预算因此需要同时计入激活、临时完整参数及通信缓冲．

重算中的集合通信必须由相应通信组按一致顺序参与．若一部分进程重算含集合操作的分支而其他成员跳过，可能造成等待或执行不一致．分片参数释放和预取的时机还决定重算期间的通信开销，不能直接把额外前向计算量当作完整的重算成本．

重计算要求函数可重复执行．若重新生成不同的 Dropout 掩码，反向对应的函数已经改变；若第二次前向重复修改持久缓冲，还可能重复产生副作用．[1]

29.4　集合通信及反向计算的依赖

29.4.1　四种基本操作

全归约（All-Reduce）将各成员的等形状张量按指定算子归约，并将完整结果交给每个成员．**归约散播**（Reduce-Scatter）执行同样归约，但每个成员只获得结果的一片．**全收集**（All-Gather）将各成员的分片收集为完整结果．**全互换**（All-to-All）则由每个成员向所有目标分别发送指定部分．

将 All-Reduce 理解为 Reduce-Scatter 后接 All-Gather，有助于解释状态分片如何省去不需要的副本．它不表示所有实现必须按这两个独立调用执行，也不表示每种拓扑下同一算法最优．

对 R 个成员的理想环式算法，完整消息大小为 M 字节，每个阶段每成员传递约 $\frac{M}{R}$．Reduce-Scatter 和 All-Gather 各需 $R-1$ 轮，因此

$$t_{\text{AR}} \approx 2(R-1)\alpha + 2\frac{R-1}{R}M\beta. \tag{29.6}$$

式中忽略归约计算、链路竞争和协议开销，$\frac{1}{\beta}$ 是该消息路径可利用的有效带宽．小消息可能受轮次时延主导，大消息更易受带宽主导．树式组织改变轮次和负载分布，不能只凭一条消息大小决定实际最优选择．

[1] “梯度检查点”是这一机制的常用别名，但通常保存的是前向边界状态，不是已经计算好的完整梯度．为与持久化训练检查点区别，本书优先使用“激活重计算”．

29.4.2　桶大小及通信尾部

反向沿层逆序产生梯度. 将若干梯度合并为通信桶，可以在前面层仍计算时归约已经就绪的桶. 大桶减少启动次数，却可能推迟首个通信；小桶更早就绪，却增加时延和调度开销. 真正不能隐藏的是最后一批计算之后仍未完成的通信尾部.

若计算时间为 t_c、独立通信时间为 t_m，完全串行时为 $t_c + t_m$，理想完全重叠的下界为 $\max(t_c, t_m)$. 依赖、带宽争用及共享计算单元常使实际时间位于两者之间，甚至因资源竞争超过独立时间之和. 通信重叠是需要度量的优化选择，不是所有工作负载都应强制开启的条件.

29.5　ZeRO 及 FSDP 状态分片

零冗余优化器（Zero Redundancy Optimizer，ZeRO）按状态类别逐步消除数据并行副本之间的重复存储 [144]. 以 R 为分片组大小，持久状态近似为

$$M_0 = P_s + G_s + O_s, \tag{29.7}$$

$$M_1 = P_s + G_s + \frac{O_s}{R}, \tag{29.8}$$

$$M_2 = P_s + \frac{G_s + O_s}{R}, \tag{29.9}$$

$$M_3 = \frac{P_s + G_s + O_s}{R.} \tag{29.10}$$

阶段 1 由各成员更新其拥有的优化器状态和参数片，再使其他成员获得更新后的参数；阶段 2 进一步只保留所需的梯度片；阶段 3 在计算某模块之前临时取得该模块的参数. 它们改变状态所有权，而不是将不同样本训练成互不相关的模型.

完全分片数据并行（Fully Sharded Data Parallel，FSDP）同样围绕按需聚合参数与归约梯度组织全状态分片 [156]. 具体实现可以选择计算后重新分片或暂时保留完整参数；这些策略改变通信与峰值. 不能仅由“FSDP”名称推断一次更新的精确通信量.

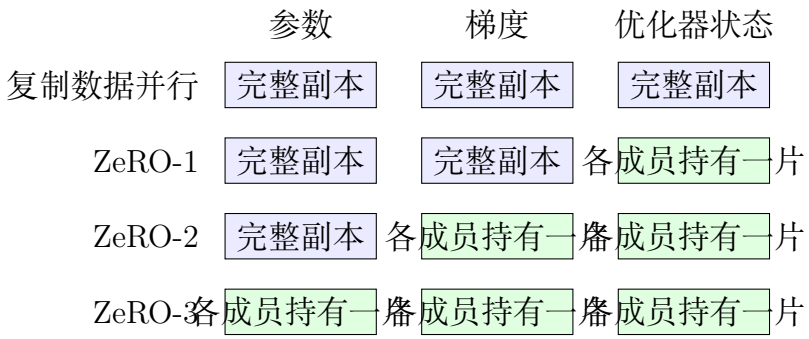


图 29.1: 状态的逻辑所有权. 图中分片不包含前向或反向期间临时聚合的完整模块.

峰值显存还包含激活、当前聚合模块、预取模块、梯度桶、分配器碎片与临时工作区. 设最大聚合模块为 U 字节，则即使持久状态很小，至少仍需为该模块的计算表示留出空间；同时预取下一模块还会增加驻留量. 过粗的分片单元可能使“全分片”仍然无法执行.

29.5.1　容量通信测算

例 28.1

设某模型的 $P_s = 12$ GiB、$G_s = 12$ GiB、$O_s = 72$ GiB，分片组 $R = 8$. 四种持久状态分别为 96、33、22.5 和 12 GiB. 若另有 10 GiB 激活与 6 GiB 工作区，阶段 2 近似为 38.5 GiB；阶段 3 若需要 4 GiB 聚合参数和 4 GiB 预取，则约为 36 GiB. 该构造例说明：持久状态从 22.5 降到 12 GiB，不代表峰值也下降 10.5 GiB.

对阶段 3，若每层参数在前向后释放、反向前重新聚合，那么一个微批次可能需要两次参数收集；若为减少通信而跨阶段保留参数，则显存增大. 梯度累积也未必消除这些逐微批次的参数通信. 精确通信量必须从参数生命周期和实际调度计算，不能只乘一个固定“ZeRO 倍率”.

卸载（Offload）把部分状态或更新计算移到主机内存、CPU 或存储. 它与分片阶段是不同维度. 假设每步必须通过某链路传输 Q 字节且有效带宽为 b，仅这段搬运就有 $\dfrac{Q}{b}$ 的时间下界. 预取能隐藏一部分等待，但不能消除总带宽约束；当数据管线和卸载共享链路时还需合并预算.

29.6　张量并行的代数结构

29.6.1　列切分、行切分及非线性

张量并行（Tensor Parallelism，TP）将一个算子的张量维度分配到多个设备. 令 $X \in \mathbb{R}^{n\times d}$、$W \in \mathbb{R}^{d\times f}$. 沿输出列切分为 $W = [W_1 \ \cdots \ W_q]$，则

$$XW = [XW_1 \ \cdots \ XW_q]. \tag{29.11}$$

每个成员获得一组输出特征. 若下一运算逐元素进行，则可直接在分片上计算，无需立即收集完整结果.

再令第二层权重 $V \in \mathbb{R}^{f\times d}$ 沿输入行分成 $V_i \in \mathbb{R}^{\frac{f}{q}\times d}$，有

$$Y = \phi(XW)V = \sum_{i=1}^{q} \phi(XW_i)V_i. \tag{29.12}$$

第一层的列切分和第二层的行切分可以配对，仅在部分和合并处通信. 这是 Transformer 层内并行的一种重要组织方式 [157]. 它依赖 ϕ 不混合分片特征；若中间运算

是沿完整特征轴的归一化，则还需归约统计量．通常 $\phi(\sum_i Z_i) \neq \sum_i \phi(Z_i)$，因此不能任意移动非线性和归约的位置．

对门控前馈层，门分支和数值分支须按相同中间特征索引切分，先在本地相乘，再由输出投影产生部分和．反向时，列并行各分片对输入梯度的贡献也需要相加；只画前向通信会低估总成本．

例 28.2

取 $X = (1, 2)$，

$$W_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad W_2 = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}, \quad V_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad V_2 = \begin{pmatrix} -1 \\ 1 \end{pmatrix}.$$

采用逐元素 ReLU，两个本地隐藏向量为 $(1, 2)$ 和 $(3, 1)$，均无需截断；输出部分和为 $1 + 4 = 5$ 和 $-3 + 1 = -2$，最终输出为 3．单设备将隐藏拼成 $(1, 2, 3, 1)$ 再乘完整 V，同样得到 3．

若输出的上游梯度为 1，本地输入梯度分别为 $(1, 2)$ 与 $(-2, 0)$，相加得到 $(-1, 2)$．因此即使前向输出可以通过一次求和得到，反向仍存在输入梯度的跨成员汇总．

注意力头也可按组切分，但 GQA 的键值头数可能少于张量并行度．此时需要改变分片粒度或复制某些键值头，并计算重复状态和梯度归约；不能在整数头数无法整除时简单向下取整．

29.7　流水并行及调度

流水并行（Pipeline Parallelism，PP）沿层深度划分模型，以微批次在阶段之间传递激活和反向梯度．GPipe 展示了以微批次组织层间流水训练的方式 [158]．以下推导采用同步更新和均衡阶段，不包含过时权重的异步方案．

设有 p 个阶段、m 个微批次，每阶段一次前向耗时 f、反向耗时 b．采用先完成全部前向、再完成全部反向的填充—排空调度，前向占 $(m+p-1)f$，反向占 $(m+p-1)b$，总时间为

$$t_{\text{pipe}} = (m + p - 1)(f + b). \tag{29.13}$$

每阶段有效工作为 $m(f + b)$，因此理想利用率与气泡比例为

$$U = \frac{m}{m + p - 1}, \qquad \rho_{\text{bubble}} = \frac{p - 1}{m + p - 1}. \tag{29.14}$$

当 $p = 4, m = 8$ 时 $U = \dfrac{8}{11} \approx 72.7\%$．增加 m 减少气泡比例，却可能增加待反向激活，或者迫使固定全局批量下的微批量变小，从而降低算子效率．

一前向一反向调度（One-Forward-One-Backward，1F1B）在预热后交替处理就绪的前向与反向，可降低同时保留的激活数．它并不自动消除所有启动和排空开销；交

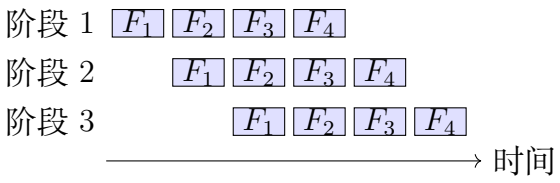


图 29.2: 三个均衡阶段处理四个微批次的前向流水. 每个微批次必须等待上一阶段，形成启动与排空空位；反向具有反向依赖.

错虚拟阶段等方法还会改变通信次数与调度依赖. 保持同步语义时，一次全局更新中的微批次必须使用一致的参数版本，待贡献完整后再更新.

阶段划分应依据计算与内存，而非只平均层数. 嵌入、词表投影、不同序列长度及 MoE 负载可能造成显著差异. 稳态节拍受最慢阶段约束；某阶段算力空闲可能是在等待依赖，不能据此认定其设备性能不足.

29.8 序列、上下文及专家并行

序列并行（Sequence Parallelism，SP）常指将某些逐位置激活和运算沿序列轴分片；**上下文并行**（Context Parallelism，CP）则通常需要处理跨设备的注意力上下文. 不同系统的命名范围存在差异，应以实际切分对象和通信过程为准. 序列并行与选择性重算的组合见原论文 [159]；Megatron Core 对上下文并行的具体切分与通信定义见其官方文档 [160].

若查询按序列分片，而每个查询仍应读取全部合法键值，则必须传递键值块，或等价地交换注意力计算所需的信息. 不能只在各自局部片段计算 Softmax 后将输出相加，因为局部分母不同.

设一个查询对两个键块的分数分别为 $z^{(1)}$ 和 $z^{(2)}$. 每块保存最大值 a_i、指数和 $l_i = \sum_j \exp(z_j^{(i)} - a_i)$ 与未归一化值和 $o_i = \sum_j \exp(z_j^{(i)} - a_i) v_j$. 取 $a = \max(a_1, a_2)$，则

$$l = e^{a_1 - a} l_1 + e^{a_2 - a} l_2, \tag{29.15}$$

$$o = e^{a_1 - a} o_1 + e^{a_2 - a} o_2, \qquad y = \frac{o}{l}. \tag{29.16}$$

这是把相同全局指数和分块后重组的恒等式，为跨块精确注意力提供数学基础. 实际系统仍须处理因果掩码、全掩码块、反向梯度和数据移动；全掩码块应作为零贡献显式处理，避免无穷值相减.

专家并行（Expert Parallelism，EP）将 MoE 专家分布在不同成员上，按路由结果分发词元，在专家计算后把输出送回并加权组合. 若 N 个词元各选择 k 个专家，隐藏维数为 d，每元素 b_e 字节，则一次分发与返回的逻辑载荷约为 $2Nkdb_e$，另有索

引、路由权重和容量填充等开销．该式计入本地路由的逻辑数据，不等于每张网卡的实际流量．

当少数专家接收过多词元时，最大负载决定阶段耗时．增加专家设备数可能降低每成员参数量，却增加路由通信及小矩阵计算．路由损失和容量策略见第三十六章《混合专家模型》；本章关注这些选择在分布式执行中的负载后果．

29.9　混合并行及故障恢复

29.9.1　通信拓扑映射

在各维度互相独立的简单网格中，总设备数为 $D = Rpq$．每个数据副本由 pq 个设备共同执行一次模型，只有数据并行维 R 增加不同样本的并发量．若每副本每微步有 B_μ 个样本、累积 G 步，则全局样本数为 $RB_\mu G$，不能再乘 p 和 q．

例如 32 个设备采用 $R = 2, p = 4, q = 4$，每副本微批量 2，累积 8 步，全局样本数为 32．若每个样本恰有 1023 个有效目标，则一次更新包含 32736 个目标；长度变化时改用式 (29.3) 中的实际计数．专家组或上下文组若复用了已有网格维度，不能在 D 上重复乘其组大小．

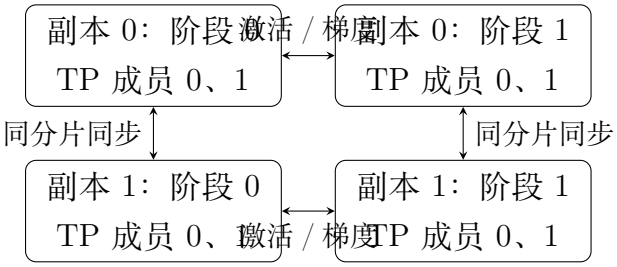


图 29.3: 八设备逻辑网格：两数据副本、两流水阶段、每阶段两张量成员．每个框对应两个不同物理设备，纵向同步匹配同一参数分片．

层内高频通信宜优先映射到适合其带宽和时延需求的互连域，但这是成本判断，不是固定的“节点内只能 TP”规则．参数分片、专家流量或大规模上下文交换也可能跨节点；必须使用第二十八章《算力基础设施》的链路约束计算关键路径和超售影响．

29.9.2　一致更新及分布式检查点

所有相关成员应对有效计数、非有限梯度、更新版本和停止条件达成一致．一个成员独自跳过集合操作可能使其他成员永久等待；一个成员更新而另一个成员跳步，会使副本分叉．错误传播须进入明确的组级取消或重启路径．

分布式检查点将第十九章《训练稳定性》的一致性边界扩展到整个逻辑模型．清单应记录每片的逻辑参数名、全局形状、偏移范围、数据类型与摘要，以及数据组和

随机状态. 恢复时先验证全部必需片段，再按目标布局读取. 改变分片布局是张量重分布；改变数据并行度还可能改变样本次序、归约顺序和批量语义，二者不能混为同一操作.

算法 29.1　按全局有效目标执行同步更新

输入：固定模型与资产、合法通信组、一个累积窗口；**输出：**各逻辑参数的一致新版本.

1. 确认各副本对应同一参数版本，为窗口中的有效标签计数；在数据并行组中求和得到 N.
2. 若 $N=0$，所有成员一致跳过；否则按通信采用求和或平均的约定确定损失系数.
3. 参数保持不变，逐微批次执行分片前向、反向与必要激活重计算；依赖允许时重叠通信.
4. 完成全部梯度贡献和组内归约，恢复未缩放梯度，统一判定有限性. 异常时按共同策略跳步或停止.
5. 对唯一逻辑参数计算全局范数，按同一系数裁剪；状态所有者执行优化器更新，并同步所需参数片.
6. 全组提交成功更新计数和已消费数据位置；到达保存边界时写出同一代的状态分片及完整清单.

29.10　扩展效率及效果判断

强扩展（Strong Scaling）固定总工作量并增加设备数. 若同一任务在 D_0 与 D 个设备上耗时 t_{D_0} 和 t_D，相对加速比为 $\dfrac{t_{D_0}}{t_D}$，效率为

$$E=\frac{\dfrac{t_{D_0}}{t_D}}{\dfrac{D}{D_0}}. \tag{29.17}$$

弱扩展（Weak Scaling）随设备数增加总工作量以保持每设备工作量近似不变. 若扩大的是全局批量，优化轨迹也可能变化，因此弱扩展吞吐不能独自证明达到同等质量的时间减少.

若 8 设备某固定预算耗时 100 分钟，16 设备耗时 60 分钟，则加速比为 $\dfrac{5}{3}$，效率约为 83.3%. 若模型不能在单设备容纳，应使用可运行的多设备基线并明确 D_0，不能虚构单设备耗时. 还应同时报告有效目标词元吞吐、实际序列长度、更新次数、精度、峰值显存和恢复成本.

优化成立需要两条证据：数学与数值行为仍满足声明的训练语义；在相同质量目标和资源约束下，时间或成本确有改善．更高设备利用率可能来自额外重计算，更多处理词元可能来自填充，较低显存也可能换来更长通信等待．应从端到端关键路径解释这些指标之间的关系．

29.11　分布式训练框架的工程落点

PyTorch FSDP、DeepSpeed 和 Megatron-LM 把分布式机制落实到不同的接口与工作流．FSDP 围绕模块参数和训练状态的分片组织计算；DeepSpeed 提供 ZeRO 及相关训练运行能力；Megatron-LM 及 Megatron Core 面向大规模 Transformer 训练，组合张量、流水线等并行机制 [161, 162, 163]．项目提供某种能力不等于该能力适用于任意模型、硬件与版本组合．

FSDP2 采用逐参数分片表示，并通过 `fully_shard` 组织模块的参数收集与重新分片；阅读实现时应把它与传统 FSDP 包装器区分，配置和状态导出方式不能机械互换 [161]．DeepSpeed 的 ZeRO 阶段则对应优化器状态、梯度和参数的不同切分范围．仅训练很少量 LoRA 参数时，优化器状态节省可能有限；不能把全参数训练中的显存收益比例直接搬到适配器训练．

Accelerate 可以组织启动、设备与训练后端配置，并与上层训练器衔接 [164]．它与 FSDP 或 DeepSpeed 之间存在编排与执行的关系．一次运行应明确由谁管理优化器、梯度累积、学习率调度和混合精度，避免在多处配置不一致的批量或重复更新．

迁移应从一个可容纳的单设备或较少设备基线出发，固定有效全局批量、监督词元计数和更新次数，再引入分片．首轮验证关注损失、梯度或参数更新差异以及检查点恢复；性能比较随后测量有效词元吞吐、通信等待、峰值显存和保存恢复时间．对于无法建立单设备基线的大模型，应公开可运行的最小分布式基线，而不是用虚拟单卡数字计算加速比．

框架产生的分片检查点用于恢复训练，部署制品用于加载推理，两者可能具有不同格式．将分片状态汇总为完整权重时，要预算主机内存、存储和集合通信，并检验权重、配置、分词器与适配器的完整性．对恢复的验收应继续一次真实更新并检查数据进度；仅恢复模型权重无法说明训练状态已经连续．

第三十章　增量解码

自回归生成每次只增加少量词元，但新的词元需要读取已经形成的上下文．如果每一步重新执行整个前缀，模型会反复计算相同的位置表示．键值缓存利用因果结构保留这些中间结果，把后续计算集中于新增位置；与此同时，缓存随序列增长，成为并发、显存与状态一致性的核心约束．

本章先证明缓存复用的数学条件，再讨论不同注意力结构的容量、分页寻址和前缀共享．第六章《Transformer 网络结构》定义网络层，第三十七章《长上下文建模》讨论长距离能力．本章侧重执行状态；算子优化和投机生成由第三十三章《推理计算优化》展开，服务调度由第三十一章《批处理调度》展开．

30.1　预填充及增量解码

30.1.1　两个阶段的输入输出

预填充（Prefill）对已知提示的全部位置执行因果前向，形成各层键和值．**增量解码**（Incremental Decoding）随后只处理新加入的词元，通过历史缓存读取前缀信息．它改变执行方式，不改变自回归概率分解．

提示包含 P 个词元，编号 $0,\dots,P-1$．预填充最后位置的输出分布预测第一个生成词元 y_0；采样出 y_0 后，才把它送入位置 P 的增量前向，得到预测 y_1 的分布．因而输出 G 个词元通常需要一次预填充与 $G-1$ 次后续单词元前向，而不必为最后一个输出再执行一次模型．[1]

30.1.2　张量形状

设批次大小 B，隐藏维数 d，层数 N，查询头数 h_q，键值头数 h_{kv}，键维数 d_k，值维数 d_v．预填充一层有

$$H \in \mathbb{R}^{B\times P\times d},\quad Q \in \mathbb{R}^{B\times h_q\times P\times d_k},\quad K \in \mathbb{R}^{B\times h_{kv}\times P\times d_k},\quad V \in \mathbb{R}^{B\times h_{kv}\times P\times d_v}. \tag{30.1}$$

[1]若服务需要把完整最终回答状态留下用于续聊，可能额外处理最后输出词元．请求返回了多少词元与缓存实际覆盖到哪个位置应分别记录．

若已经缓存 L 个词元，新加入 q 个词元，则 Q_{new} 的序列轴为 q，K, V 的可读取序列轴为 $L+q$. 单词元解码时 $q=1$，每个查询头的注意力分数形状为 $[B, 1, L+1]$. 实际运行时可采用不同轴顺序或打包布局，逻辑维度不变.

设 $g(i)$ 将查询头 i 映射到其键值头，新增位置 $r \in \{0, \dots, q-1\}$ 的注意力为

$$o_{r,i} = \sum_{j=0}^{L+r} \frac{\exp(\frac{q_{r,i}^{\top} k_{j,g(i)}}{\sqrt{d_k}} + b_{L+r,j})}{\sum_{u=0}^{L+r} \exp(\frac{q_{r,i}^{\top} k_{u,g(i)}}{\sqrt{d_k}} + b_{L+r,u})} v_{j,g(i)}. \tag{30.2}$$

b 表示合法的位置偏置，旋转位置编码则通常已经作用于查询与键. 新块内部仍需因果约束 $j \leqslant L+r$，不能因为使用缓存而让较早新词元读取较晚新词元.

30.1.3　符号及一致性假设

符号	含义
P, G, L, q	提示长度、输出词元数、已缓存长度、新增位置数.
$N, d, h_q, h_{kv}, d_k, d_v$	网络层数、隐藏维数及注意力维度.
b_k, b_v	单个键或值元素占用的字节数.
$\mathcal{C}^{(\ell)}$	第 ℓ 层的键值缓存及其位置元数据.
$p_b, \mathcal{B}$	每个物理块容纳的词元数与逻辑块到物理块的映射.
r_c, d_C, d_R	潜变量缓存维数、内容查询/键维数与解耦位置分量维数.

缓存等价的条件

模型参数、适配器、位置规则和可见性规则固定；网络处于推理模式，旧位置的计算不依赖未来词元. 层归一化和前馈变换按位置进行，不以不断扩展的整段序列重新计算全局统计. 缓存没有被有损修改，并保留与参考全量前向一致的逻辑位置和输入条件.

30.2　键值缓存等价性

30.2.1　层级因果性证明

键值缓存（Key-Value Cache，KV Cache）保存各层历史位置的 K, V. 设在长度 L 的前向中，第 ℓ 层历史位置 $j < L$ 的隐状态为 $h_j^{(\ell,L)}$. 现在追加词元扩展为 $L+q$，需要证明

$$h_j^{(\ell,L)} = h_j^{(\ell,L+q)}, \qquad j < L. \tag{30.3}$$

第一层之前，旧词元及位置编号相同，嵌入相同. 假设某层输入的历史状态相同，则其历史 Q, K, V 相同；因果掩码保证旧位置 j 只读取 $0, \dots, j$，新增位置不进入求和. 于是注意力输出相同，逐位置残差、归一化和前馈结果也相同. 对层数归纳，式 (30.3) 成立.

计算新增位置时，使用缓存的历史键值与重新计算整个前缀所得到的历史键值相同，新位置本身按同一公式计算，故新增隐状态与输出分布也相同．这是缓存等价的依据，不是因为历史文本“通常不变”而作出的近似．

历史查询通常不必保存，因为以后只需要新位置发出的查询；历史键和值仍将被后续查询读取．跨层不能只保存最后一层缓存：第 ℓ 层新查询需要第 ℓ 层的历史键值，其表示与其他层不同．

30.2.2 等价性条件

双向编码器旧位置会读取新增文本，通常不能直接沿用这套追加证明．对编码器—解码器模型，如果编码器输出固定，则交叉注意力的编码器键值可以预先计算；若源文本改变，则这些缓存失效．

推理时仍启用随机丢弃、随长度改变位置缩放、修改适配器或依赖全序列统计的操作，都会破坏旧状态不变的前提．量化缓存、选择性淘汰或近似压缩也会引入额外误差，应作为不同执行模式说明．

数学等价不要求不同内核逐位相同．浮点求和顺序、精度与归约布局可能带来误差；输出概率接近时，贪心最大值也可能发生离散分叉．验证应先比较同一输入下的分数或概率，再区分数值误差与掩码、位置错误，不能只凭生成文本相同或不同判断缓存正确性．

例 24.1

考虑单层单头、$d_k = 1, d_v = 2$ 的构造．前两个位置的缓存为

$$k_0 = 0,\ k_1 = \log 2, \qquad v_0 = (1, 0),\ v_1 = (0, 2). \tag{30.4}$$

第三个新位置的查询 $q_2 = 1$，新键 $k_2 = \log 3$，新值 $v_2 = (3, 3)$．无位置偏置时，分数为 $0, \log 2, \log 3$，指数为 $1, 2, 3$，归一化权重为 $\frac{1}{6}, \frac{2}{6}, \frac{3}{6}$．所以

$$o_2 = \frac{1}{6}(1, 0) + \frac{2}{6}(0, 2) + \frac{3}{6}(3, 3) = \left(\frac{5}{3}, \frac{13}{6}\right). \tag{30.5}$$

全量前向会为第三行生成完全相同的分数；缓存前向复用前两行键值即可．它不需要重新计算前两行注意力输出．若错误漏掉新键值，得到 $(\frac{1}{3}, \frac{4}{3})$，这不是数值精度差异，而是可见集合改变．

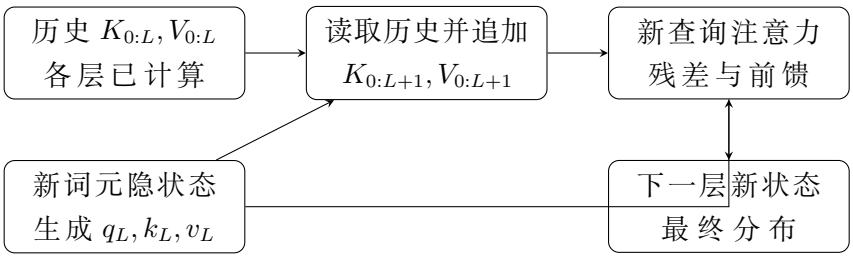


图 30.1: 单层增量计算的数据流．新查询读取全部合法键值，历史查询和历史前馈输出不再重算．

30.3　复杂度及带宽

30.3.1　重复前缀及缓存执行

忽略层数等固定系数，以密集网络常见维度关系计，长度 L 的全量前向约为 $O(Ld^2 + L^2 d)$．如果生成每个词元都重算整个前缀，G 次预测消耗约

$$\sum_{g=0}^{G-1} O\big((P+g)d^2 + (P+g)^2 d\big). \tag{30.6}$$

使用缓存后，预填充一次，后续 $G-1$ 次单词元前向各为 $O(d^2 + (P+g)d)$，总量为

$$O(Pd^2 + P^2 d) + O\big(Gd^2 + (GP + G^2)d\big). \tag{30.7}$$

因此无初始长提示时，朴素重复前缀的注意力累计量可达三次方，而缓存把它降到二次方；单步解码的密集注意力仍随上下文线性增长，不能称为常数时间．

例如提示 4 个词元、输出 3 个词元．全量方案分别处理长度 4、5、6，累计线性变换位置 15 个；缓存方案处理 4 个提示与 2 个新增位置，共 6 个．按因果合法分数对计，前者为 $10 + 15 + 21 = 46$，后者为 $10 + 5 + 6 = 21$．这是运算对象计数，不是端到端时间加速比．

30.3.2　缓存容量及计算量

单步解码需要读取历史键值．若主要限制是显存带宽，减少缓存元素有助于降低读取成本；但查询头数不变时，注意力分数与输出仍按查询头计算．预填充可以对许多查询并行执行，单词元解码更容易受参数与缓存读取影响，二者不能共用一个固定吞吐常数．

缓存只避免重算，不会消除新词元的网络层计算，也不会消除全上下文注意力读取．实际耗时还取决于批次、内核复用、页布局、并行通信和工作区；第三十三章《推理计算优化》讨论算子，第三十一章《批处理调度》讨论调度．本章容量式不替代实际负载下的延迟分析．

30.4　MHA、GQA 及 MQA 的缓存

30.4.1　头共享结构

多头注意力（Multi-Head Attention，MHA）为每个查询头提供独立键值头，$h_{kv} = h_q$．**多查询注意力**（Multi-Query Attention，MQA）让所有查询头共享一个键值头，

$h_{kv} = 1$[165]. **分组查询注意力**（Grouped-Query Attention，GQA）在两者之间按组共享键值 [24].

若 h_q 能被 h_{kv} 整除，每组有 $\dfrac{h_q}{h_{kv}}$ 个查询头. 共享键值并不意味着查询头输出相同，因为各查询向量仍不同，从而产生不同注意力权重. 这些是不同参数化结构；不能把已有 MHA 键值头直接丢弃后声称数学等价. 转换需要与模型训练方式及质量评估结合.

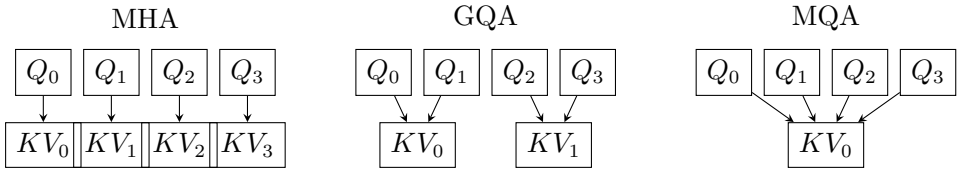


图 30.2: 四个查询头下的键值共享关系. 查询头数相同，缓存键值头数依次为 4、2、1.

30.4.2 统一容量公式

若请求 b 缓存长度为 L_b，各层同形状，则裸缓存容量为

$$M_{KV} = N \sum_{b} L_b h_{kv} (d_k b_k + d_v b_v). \tag{30.8}$$

当 $d_k = d_v = d_h, b_k = b_v = b_e$、所有请求长度 L 相同时，退化为 $2NBLh_{kv}d_h b_e$. 它不包含页表、分配碎片、量化比例、临时空间和模型权重. 分层采用滑窗或不同头数时，应分别求和.

取 $N = 32, h_q = 32, d_h = 128, b_e = 2, L = 8192, B = 1$. 每词元 MHA 缓存为 $2 \times 32 \times 32 \times 128 \times 2 = 524288$ 字节，即 512 KiB. GQA 取 8 个键值头，每词元 128 KiB；MQA 每词元 16 KiB. 于是

结构	键值头数	单请求裸缓存	四请求裸缓存
MHA	32	4 GiB	16 GiB
GQA	8	1 GiB	4 GiB
MQA	1	128 MiB	512 MiB

这里 GiB 为 2^{30} 字节. 相同比例描述缓存容量，不代表模型总显存或吞吐按同样比例变化.

张量并行下，若键值头均匀分片，每卡缓存可相应减少；若键值头少于并行设备数而运行时复制，则不能直接除以并行度. 例如 MQA 的一个头可能在多个设备上分别驻留. 容量规划需使用实际每卡存储头数，并与模型及内核布局一致.

30.5　低秩键值及潜变量缓存

30.5.1　联合表示及矩阵吸收

键值头共享减少独立头数，另一种思路是让键和值由低维共同变量生成. 对单位置列向量 $h_t \in \mathbb{R}^d$，设

$$c_t = W_D h_t \in \mathbb{R}^{r_c}, \quad k_{t,i} = U_{K,i} c_t, \quad v_{t,i} = U_{V,i} c_t, \tag{30.9}$$

其中 $U_{K,i} \in \mathbb{R}^{d_k \times r_c}$，$U_{V,i} \in \mathbb{R}^{d_v \times r_c}$. 不含位置旋转时

$$q_{t,i}^{\top} k_{j,i} = (U_{K,i}^{\top} q_{t,i})^{\top} c_j. \tag{30.10}$$

可以先把当前查询投到潜空间，再与缓存 c_j 点积，不必为每个历史位置展开完整键. 输出也有

$$\sum_j a_{t,j,i} v_{j,i} = U_{V,i} \sum_j a_{t,j,i} c_j. \tag{30.11}$$

若第 i 头输出投影块为 O_i，可进一步合并 $O_i U_{V,i}$. 这种**矩阵吸收**（Matrix Absorption）利用线性结合与矩阵乘法结合律，改变计算顺序而保留所定义低秩结构的数学结果.

低秩约束本身不等于任意原始 MHA 的无损压缩. 模型若按该结构训练，可以精确执行自身的潜变量形式；若对已有高秩权重作事后低秩近似，则还需分析近似误差. 缓存变小也可能增加某些潜空间计算，必须结合维度与执行阶段权衡.

30.5.2　位置旋转的非交换约束

若对完整键加入旋转 R_j，查询也按位置 t 旋转，则点积为

$$q_{t,i}^{\top} R_t^{\top} R_j U_{K,i} c_j. \tag{30.12}$$

中间旋转依赖历史位置 j，一般不能用一个与 j 无关的固定矩阵吸收到当前查询投影中. 矩阵通常不交换次序；忽略这一点会把缓存压缩推导变成错误等式.

多头潜在注意力（Multi-Head Latent Attention，MLA）在 DeepSeek-V2 中采用联合键值压缩与解耦位置分量 [166]. 概念上可把分数写为内容项和位置项之和：

$$s_{t,j,i} = \frac{(q_{t,i}^{C})^{\top} U_{K,i} c_j + (q_{t,i}^{R})^{\top} k_j^{R}}{\sqrt{d_C + d_R}}. \tag{30.13}$$

内容项采用矩阵吸收，旋转位置项单独保留. 分母与该结构的查询、键总维度和训练尺度一致，不能任意换成潜变量维数. 这里只给出缓存相关关系，具体查询压缩与模型规格应遵循对应架构.

若每层每位置保存一个 r_c 维潜变量和一个共享 d_R 维位置键，统一元素字节数 b_e 时，裸容量为

$$M_{\text{latent}} = N \sum_b L_b (r_c + d_R) b_e. \tag{30.14}$$

例如取 $r_c = 512, d_R = 64, N = 32, b_e = 2, L = 8192$，得到 288 MiB．该构造不能直接与不同模型的性能比较；这里只说明缓存由哪些维度决定．

30.6　逻辑位置及物理分页

30.6.1　页表不改变注意力位置

分页键值缓存（Paged KV Cache）把序列的连续逻辑位置映射到可不连续的物理块．PagedAttention 将此思路用于模型服务中的缓存管理 [167]．若每块容纳 p_b 个词元，逻辑位置 t 对应

$$j = \lfloor \frac{t}{p_b} \rfloor, \qquad o = t \bmod p_b, \qquad \mathrm{physical}(t) = (\mathcal{B}[j], o). \tag{30.15}$$

j 为逻辑块编号，o 为块内偏移，$\mathcal{B}$ 是请求页表．位置编码仍使用逻辑位置 t，不能使用物理块编号．搬迁一个块只需更新映射与数据位置，不应重新解释序列顺序．

设块大小 4、逻辑长度 10，页表为 $(7, 2, 9)$．位置 6 映射到物理块 2 的偏移 2，位置 9 映射到物理块 9 的偏移 1．三个物理块共有 12 个槽位，最后两位无效，注意力必须根据逻辑长度排除它们．

图 30.3: 逻辑块映射到不连续物理块．页表解决存储布局，逻辑位置和有效长度决定模型可见的内容．

30.6.2　碎片及容量预留

没有共享时，请求长度 L_b 需要 $\lceil \frac{L_b}{p_b} \rceil$ 个块，分配槽位与有效长度之差小于 p_b．对 B 条请求，尾部内部碎片少于 Bp_b 个词元槽位．块太大增加尾部浪费，块太小增加页表和寻址开销，选择应结合内核与长度分布．

缓存按需增长减少一次预留最大输出长度的浪费，但不能消除后续增长风险．接纳请求时应考虑未来输出预算及可用块；正在执行的内核读取旧映射期间，块不能被提前回收．页表修改、数据拷贝与设备执行之间需要明确顺序和完成边界．

30.6.3　批内变长

对长度分别为 $L_1, \ldots, L_B$ 的请求，每个新查询仅能读取自身有效前缀．连续批次中的行号只是当前执行槽位，不能代替稳定请求身份．请求完成后压缩批次、交换行

或加入新请求时，缓存映射必须跟随请求，而不能默认沿用原行对应的物理块.

填充到批内最大长度的实现与分页变长实现可以表达同一逻辑计算；前者需要屏蔽无效位置，后者需要正确页表与长度. 它们的缓存分配量和算子效率不同，但语义不应改变.

30.7　前缀共享及缓存生命周期

30.7.1　计算前缀共享

前缀缓存（Prefix Cache）复用多个请求相同前缀产生的键值. 因为旧状态只依赖前面的内容，相同模型、相同逻辑位置与相同条件下的精确词元前缀可以共享. 相同的一段后缀若前文不同，深层键值通常不同，不能仅按局部文本重复复用.

共享通常从完整块边界开始. 前缀块的身份应绑定前一块身份与本块词元内容，使同一局部块出现在不同前缀后时不被误判为相同计算状态. 内容摘要是索引手段，仍需要足够可靠的身份方案与兼容性元数据.

完整命中提示缓存时，若仅保存 K, V，可能没有可直接产生首个新词元的最后隐状态或输出分数. 实现可以保留相应输出状态，或重新处理最后一个提示位置并精确控制缓存截断与追加. 不能将最后词元再追加一次而造成重复位置，也不能把“所有 KV 已存在”误解为“首词元分布必然已保存”.

30.7.2　引用计数及写时复制

共享块在被多个请求引用时应保持不可变. **引用计数**（Reference Counting）记录有多少活动使用者持有该物理块；释放一个请求只减少其引用，计数归零后才具备回收条件. 若块仍被设备内核读取，还需等待相关执行完成.

当多个分支共享未满块又要追加不同词元时，需要**写时复制**（Copy-on-Write，CoW）：为修改分支分配新块，复制有效前缀，再追加. 否则一个请求会覆盖另一个请求的缓存. 已满共享块无需复制，后续分支直接分配各自的新块即可.

例如每块 4 个词元，三条请求共享 8 个前缀词元，各自拥有 3 个后续词元. 独立存储需要 $3 \times \lceil \frac{11}{4} \rceil = 9$ 块；共享后需要 2 个公共块加 3 个私有尾块，共 5 块. 有效独特缓存词元为 $8 + 3 \times 3 = 17$，分配 20 槽位；节省的是重复前缀，并未减少私有后缀的计算内容.

30.7.3　缓存驻留及淘汰

活动请求使用的缓存不能因“最近没访问”就直接删除. **缓存淘汰**（Cache Eviction）通常针对没有活动引用但为未来复用保留的块；若要抢占活动请求，应保存可恢复映

射并换出，或释放后按同一输入重新计算．前者消耗传输，后者消耗计算，两者的语义与普通缓存命中不同．

缓存可经历空闲、分配写入、可共享驻留、活动引用、无引用可淘汰、回收等状态．取消与超时也必须走释放路径，并以请求资源所有权防止重复递减．前缀复用率高但长期占满显存时，驻留策略会挤压活动解码；如何在任务之间分配资源由调度章节展开．

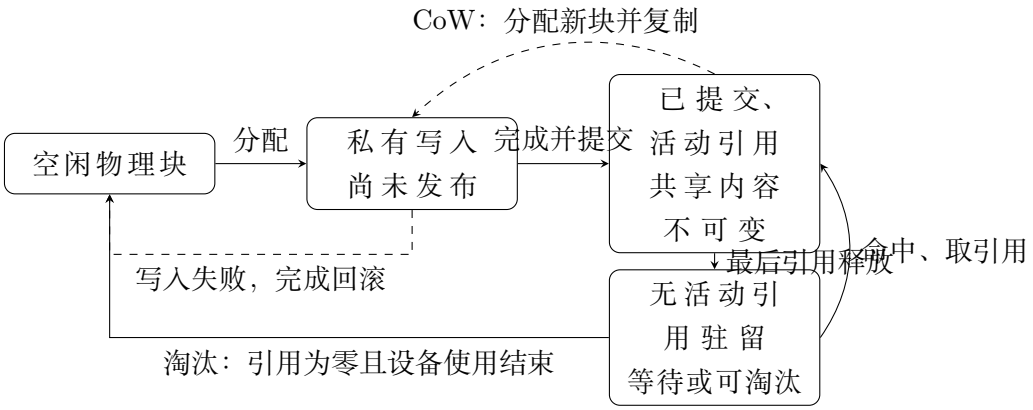


图 30.4: 缓存块生命周期．CoW 边表示创建新的私有副本，原共享块仍保持原状态；取消与正常结束都按所有权释放一次引用．未完成的设备访问禁止复用物理存储．

算法 30.1　前缀缓存的引用与追加

输入为计算身份、词元前缀、请求身份及所需新增容量；输出为请求页表和可继续执行的私有尾部．

1. 在允许共享的命名空间内查找最长兼容完整前缀，逐块验证计算身份与有效范围，取得引用后固定映射．
2. 为未命中部分分配块，执行预填充并提交有效长度；只有计算完成的内容可发布为可共享状态．
3. 每次追加前判断尾块是否有空位及是否被共享；共享且需要修改时先复制有效内容到私有块．
4. 写入各层新键值，在成功完成后推进逻辑长度；失败时回滚未提交分配，不发布部分完成状态．
5. 请求终止后按所有权恰好释放一次引用；待无引用且设备使用结束后，块可保留驻留或由淘汰策略回收．

不变量是任何活动页表只能指向兼容且已完成的有效数据，共享块不被原地覆盖，逻辑长度只描述已提交状态．

30.8　滑窗、淘汰及位置编码约束

30.8.1　历史状态淘汰条件

模型若原本采用包含当前位置的窗口 w，未来查询只读取最近 w 个键值，则窗口之外的历史键值不再作为该层的直接输入，可在保持必要元数据的条件下回收．删除全注意力模型的旧键值，则改变式 (30.2) 的求和集合，是近似策略而非精确缓存管理．

滑窗模型的高层新状态可能包含通过中间位置传递的更早信息．缓存中仅保留最近键值，不等于从原始文本只取最近 w 个词元重新运行整网：后者可能失去这些键值形成时曾使用的上下文．精确性应相对于相同逐步滑窗计算定义．

混合全局层与滑窗层时，缓存保留范围按层不同．用于前缀匹配的累计逻辑长度也不等于当前驻留位置数，应同时保存历史进度和各层可见范围．

30.8.2　环形存储的逻辑位置

环形缓存（Ring Buffer）用位置 $t \bmod w$ 复用物理槽位．槽位编号是存储坐标，位置编码仍应使用模型定义的逻辑位置 t．把槽位 0 当成绝对位置 0，会改变旋转相位或绝对嵌入．

如果缓存保存已经旋转的键，移动物理块不需要重新旋转；若改变逻辑位置，情况不同．纯相对旋转在特定条件下可以通过一致变换分析重定位，但普通网络还包含深层上下文状态，不能只改键相位就宣称任意裁剪与拼接等价．

长度扩展方案若根据当前长度动态修改旋转频率，旧键以及形成旧键的深层状态可能不再对应新规则．最稳妥的缓存复用前提是一次请求内固定位置配置；需要动态改变时应明确重建范围，而不能仅更新配置数字．

30.9　缓存身份及隔离

30.9.1　缓存失效依赖

缓存是给定计算条件的中间结果．其身份至少应覆盖权重及结构修订、适配器组合、精度与缓存格式、位置配置、注意力规则、精确前缀和外部条件．对多模态或交叉注意力模型，还需绑定媒体特征、处理器和编码器状态；同样文本标记不保证图像条件相同．

分词器与模板决定如何得到词元前缀，适合在制品身份中统一绑定．温度与采样阈值若仅作用于输出分布而不改变前向，本身不改变已有前缀 KV；但它们影响后续采样结果及恢复随机轨迹，因此仍属于请求生成状态．区分这两类依赖可以避免缓存键过宽或过窄．

模型名称、可变分支名和目录路径不是充分身份．更新模型而继续使用旧缓存会得到混合版本计算；同一个请求在解码中切换修订也不具有普通续算语义．无损搬迁到相同制品通常只改变位置，跨不同缓存布局则需要明确定义格式转换．

30.9.2　多租户共享边界

缓存包含由用户内容计算出的状态，应受请求或授权共享域控制．即使两个租户碰巧拥有相同词元前缀，也不意味着系统已获准共享其私有计算状态．命名空间、前缀来源和授权域应进入查找边界；公共固定前缀可以在明确规则下共享，私有内容不应被摘要命中自动扩大访问范围．

引用计数只保证内存生命周期，不提供权限隔离．缓存命中造成的时间差、日志或错误信息也可能泄露前缀存在性，应避免向无权请求暴露可查询的私有前缀信息．物理块重新分配后，旧内容必须在访问与有效长度上不可见，不能依赖“下次大概会覆盖”．

缓存正确性的三个层次

数学层保证保存的状态等价于相同前缀计算；存储层保证页表、有效长度与生命周期一致；隔离层保证只有授权请求能够复用状态．三者缺一不可，较高命中率不能替代任何一层的正确性．

30.10　增量执行过程

算法 30.2　带缓存的自回归生成

输入为非空提示、冻结模型、生成上限 $G > 0$、停止规则与空或兼容前缀缓存；输出为最多 G 个词元及最终缓存进度．

1. 确定提示词元、逻辑位置和缓存身份，复用合法前缀并处理未缓存部分，取得提示末位置的输出分布．
2. 根据生成策略选择下一个词元并记录输出；若达到结束条件或输出上限，终止生成．
3. 为该新词元分配可写位置，按层计算新查询、键和值，查询所有合法历史位置并完成残差与前馈；提交缓存及逻辑长度．
4. 从最终层取得下一词元分布，返回步骤 2；批内其他请求使用各自页表、长度及终止状态．
5. 正常结束、取消或失败时释放请求持有的资源；若保留续聊状态，明确缓存是否已经覆盖最后输出词元．

每次循环至多增加一个已输出词元，缓存只覆盖已经执行前向的位置；不把“已采样”误认为“已写入各层 KV”.

时间指标也应遵循阶段语义. 首词元延迟从约定入口到首个可见输出，通常包含排队与预填充；后续词元间隔描述增量阶段及其排队、传输影响. 若 TTFT 已含排队，端到端时间写为 $\mathrm{TTFT} + (G-1)\mathrm{TPOT}$，不能重复加排队. 缓存命中减少某些预填充计算，但不保证所有请求的首词元延迟同比下降.

30.11　缓存计量及服务边界

缓存增长、头共享和容量计数描述状态存储的成本；内核分块、编译与并行效率在后续章节展开. 请求身份、模型修订、取消和资源释放决定缓存生命周期与隔离边界. 容量算例用于推导字节数，不能称为实测显存或吞吐.

第三十一章　批处理调度

单个请求的算子变快，并不意味着同时到达的一组请求都能及时完成．服务系统还必须决定何时接纳请求、哪些请求共同执行、长提示如何穿插，以及资源不足时谁应等待．调度改变的是请求共享资源的方式，其收益与到达过程、长度分布和服务目标有关．

第三十章《增量解码》给出可复用状态及其容量，第三十三章《推理计算优化》给出一次执行的成本．本章将二者放入随时间变化的工作负载中．部署、连接协议与实例生命周期见第三十二章《模型服务架构》．

31.1　请求、时间及容量的计量边界

本章主要符号见表 31.1.

表 31.1: 批处理调度的主要符号与计量对象.

符号	含义
a_i, b_i, c_i	请求 i 到达、首次获得执行和终止的时间.
$u_{i,j}$	客户端收到请求 i 的第 j 个输出词元的时间.
P_i, O_i	输入长度与实际输出长度；$O_i^{\max}$ 为接纳时约定的输出上限.
W_i, T_i, S_i	首次执行前等待时间、系统逗留时间、理想队列模型中的服务时间.
$N(t), N_q(t)$	时刻 t 系统内和等待队列内的请求数.
λ, μ, ρ	队列模型中的到达率、单服务台服务率和利用率.
B, K, M	单轮新计算词元预算、活动序列上限和可用缓存字节预算.
q_i, s_i, m_i	单轮为请求分配的新位置数、已有上下文长度及缓存字节数.
d_i, w_j	请求截止时间与租户 j 的调度权重.

首词元时延（Time to First Token，TTFT）与**词元间时延**（Inter-Token Latency，ITL）分别为

$$\mathrm{TTFT}_i = u_{i,1} - a_i, \qquad \mathrm{ITL}_{i,j} = u_{i,j} - u_{i,j-1} \quad (j \geqslant 2). \tag{31.1}$$

请求生成至少两个词元时，其**平均输出词元时间**（Time per Output Token，TPOT）

定义为

$$\mathrm{TPOT}_i = \frac{u_{i,O_i} - u_{i,1}}{O_i - 1}. \tag{31.2}$$

因此最后一个词元到达的时延恰为 $\mathrm{TTFT}_i + (O_i - 1)\,\mathrm{TPOT}_i$. 这里的 TTFT 已包含到首次词元之间的排队，不能再次加上 $W_i = b_i - a_i$. 只有一个词元时 TPOT 不定义；无输出的失败请求必须进入失败统计，不能以零时延代替.

客户端时间还包含网络和缓冲，工作进程内的执行时间不包含这些部分. 二者都可记录，但必须说明观测位置. 流式协议可能一次传输多个词元；若只能观察数据块，应称为块间隔，而不是把它冒充逐词元 ITL. [1]

吞吐也有不同口径：请求完成数、输入词元数、输出词元数和满足服务目标的完成数. 预填充与解码的计算和搬运成本不同，$\lambda\mathbb{E}[P_i + O_i]$ 仅表示词元流量，不能直接视为统一的设备需求. 缓存命中又会使输入长度与实际计算长度不同.

31.2　连续批处理

31.2.1　静态批处理的容量浪费

静态批处理（Static Batching）在批次开始时确定成员，通常在当前批次退出后才加入下一组请求. 已完成序列可以提前返回结果，但空出的执行位置若不能补入新请求，仍会降低资源利用率. 不能把静态批处理一概等同于禁止提前返回.

考虑一个仅用于计算的调度模型：所有预填充均已完成，一轮解码固定耗时 10 毫秒，最多容纳两条序列. 三个请求同时到达，剩余输出长度分别为 $A = 4, B = 1, C = 1$. 静态顺序先选 A, B，B 在 10 毫秒完成，A 在 40 毫秒完成，C 在 50 毫秒完成；平均完成时延为 $\frac{100}{3}$ 毫秒. 若在第二轮将 C 加入空位，则完成时间变为 40、10、20 毫秒，平均为 $\frac{70}{3}$ 毫秒.

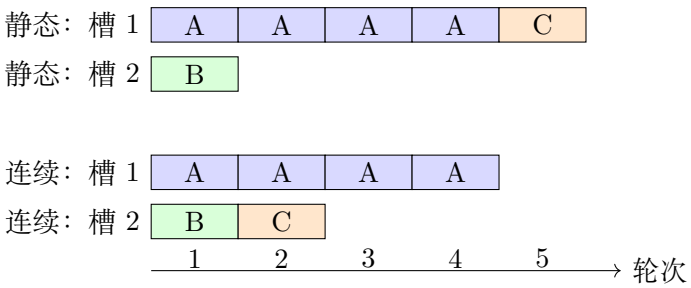


图 31.1: 在固定轮时模型下，连续批处理及时利用完成请求释放的槽位.

[1] 同一请求的终止控制消息可能晚于最后一个词元；因此完整协议时延 $c_i - a_i$ 还可包括收尾开销.

连续批处理（Continuous Batching）在迭代边界调整活动成员. Orca 将这种决策落实为迭代级调度，并根据算子特性选择批处理方式 [168]. 真实系统的轮时会随批大小、上下文长度和算子形状变化，故图 31.1 只证明该构造中的收益，不是吞吐预测.

31.2.2 长度分桶及等待窗口

若内核把同批 n 条序列填充到最大长度 $P_{\max}$，线性层与显式完整平方注意力矩阵的长度利用率分别为

$$\eta_{\text{linear}} = \frac{\sum_i P_i}{nP_{\max}}, \qquad \eta_{\text{attention}} = \frac{\sum_i P_i^2}{nP_{\max}^2}. \tag{31.3}$$

使用变长分块内核后，不能继续拿这两个比率当作实际计算浪费.

长度分桶（Length Bucketing）把相近长度放入同一执行组，减少形状差异. 但等待相近请求会增加时延. 动态聚批应同时设置最大等待时间与最大批量：任一条件触发就执行. 仅为填满批次持续等待，会在低流量下恶化 TTFT. 桶边界附近的长度抖动还可能造成低占用桶，应与第三十三章《推理计算优化》中的形状分桶中的编译缓存共同设计.

31.3 分块预填充及多资源预算

31.3.1 分块预填充的上下文边界

分块预填充（Chunked Prefill）将一个提示的新增位置分多轮处理，每块仍读取合法历史状态；它改变执行时序，不必改变模型可见上下文. SARATHI 研究了预填充分块与解码共同组批的执行方式 [169]. 具体分块长度须服从设备、批次和服务目标，不能把论文中的某个配置变成普遍最优值.

在已有 s_i 个位置时处理 q_i 个新位置，因果注意力所涉及的键查询配对数为

$$q_i s_i + \frac{q_i(q_i+1)}{2}. \tag{31.4}$$

推导是将第 r 个新查询可见的 $s_i + r$ 个键对 $r = 1, \dots, q_i$ 求和. 每块重读历史会产生搬运与启动开销；块越小，交互机会越多，但不保证总时间越短. 解码通常每序列新增一个位置，预填充则可新增许多位置，因此一个简单预算是

$$\sum_i q_i \leqslant B, \qquad |\mathcal{A}| \leqslant K, \qquad \sum_{p \in \mathcal{P}_{\text{live}}} \text{bytes}(p) + M_{\text{workspace}} \leqslant M. \tag{31.5}$$

其中 $\mathcal{P}_{\text{live}}$ 是实际驻留物理页集合，共享前缀只计一次. 临时工作区要在可用容量中扣除，不能把设备总显存直接当 KV 预算.

仅限制 B 仍不够：同样的新词元数在不同 s_i 下有不同注意力成本．调度器还需使用经测量校准的批次成本估计 $\widehat{t}(\{q_i, s_i\})$，在交互目标允许的轮时内选择组合．该函数一般不是各请求独立成本之和，因为共同执行会改变权重重用和有效带宽．

假设与适用范围

以下资源算例假设普通非共享 KV，每词元缓存为 64 KiB，页包含 16 个词元；设备分区已扣除权重与工作区，可用 KV 为 64 MiB．实际缓存公式应依据模型层数、键值头数、精度和并行分片确定，不能跨模型复用这个常数．

每页为 1 MiB．两条活动序列长度分别为 257、255，需 $17 + 16 = 33$ 页．若为它们各解码一步，长度变为 258、256，仍为 33 页．此轮另处理一个新请求的 256 词元提示块，需要 16 页，共 49 页；若该请求声明最终最多使用 1024 位置，则其完整上界需 64 页，与已有 33 页合计 97 页，不能在 64 页容量下承诺三者同时完整驻留．

这说明**本轮可执行与完整请求可接纳是两个判断**．系统可选择保守地预留完整上界，也可动态增长并提供可兑现的抢占、换出或等待策略．不能既按平均输出长度超额接纳，又声称所有请求都有无抢占完成保证．页级管理减少碎片及支持共享，但不创造物理容量 [167]．

31.3.2　预填充及解码的优先级

始终优先解码可以降低活动请求的词元间隔，却可能使等待提示长期得不到首词元．始终优先新提示又可能使活动输出停顿．可为预填充保留最低服务份额，并在交互轮时预算下分配剩余额度；当多个目标冲突时，应记录超额负载和拒绝，不能用一个平均吞吐数掩盖取舍．

31.4　排队关系的推导及适用条件

31.4.1　Little 关系来自驻留面积

Little 关系联系长期平均人数、吞吐与驻留时间 [170]．下面从样本路径面积说明其机制．对观察区间 $[0, H]$，系统内请求数可以写成指示函数之和：

$$N(t) = \sum_{i} \mathbf{1}\{a_i \leqslant t < c_i\}. \tag{31.6}$$

交换有限求和与积分，得到

$$\int_0^H N(t)\,dt = \sum_{i} |[a_i, c_i) \cap [0, H]|. \tag{31.7}$$

右侧为每个请求在观察窗内的驻留时间. 若系统稳定，长期到达率存在，边界截断对单位时间的贡献趋于零，则除以 H 并令 $H \to \infty$，得到

$$\overline{N} = \lambda \mathbb{E}[T], \qquad \overline{N}_q = \lambda \mathbb{E}[W_q]. \tag{31.8}$$

第二式对等待状态重复相同面积论证. LLM 请求可能在多轮之间反复等待，此时 W_q 应包括所有被统计的等待区间；它与首次执行前的 W_i 不必相等.

式 (31.8) 不要求泊松到达或指数服务，但要求人数、吞吐和时间来自同一系统边界与同一请求总体. 若将被拒请求计入到达率，却只对接纳请求计算驻留时间，等式失去对应关系. 例如稳定系统每秒完成 20 个请求，平均驻留 0.4 秒，则平均系统内请求数为 8；这不能推出 p99 驻留时间，也不能推出队列必须恰好设置为 8.

31.4.2 M/M/1 模型解释高利用率风险

假设与适用范围

设到达为速率 λ 的泊松过程，单服务台独立服务时间服从速率 μ 的指数分布，先到先服务、无限等待空间且无放弃，$\rho = \dfrac{\lambda}{\mu} < 1$. 该模型用于解释机制；连续批处理的服务率随活动集合变化，不直接满足这些条件.

令稳态系统人数概率为 π_n. 相邻状态的概率流平衡给出 $\lambda \pi_n = \mu \pi_{n+1}$，因此 $\pi_n = \rho^n \pi_0$. 利用几何级数归一化得

$$\pi_n = (1-\rho)\rho^n, \qquad \mathbb{E}[N] = \sum_{n \geqslant 0} n(1-\rho)\rho^n = \frac{\rho}{1-\rho}. \tag{31.9}$$

结合式 (31.8)，有

$$\mathbb{E}[T] = \frac{1}{\mu - \lambda}, \qquad \mathbb{E}[W_q] = \frac{\rho}{\mu - \lambda}. \tag{31.10}$$

后式从总时间减去平均服务时间 $\dfrac{1}{\mu}$ 得到. 当 ρ 接近 1，等待呈非线性增长；把设备长期推到饱和不能视为无代价的效率提升.

还可以推导完整逗留时间尾部. 独立泊松到达在到达前看到的状态分布为时间稳态分布：直观上，任一短区间发生到达的概率 $\lambda\, dt$ 不因当时状态改变，因此不会额外偏向某状态. 到达看到 n 个请求时，自身完成需等待 $n+1$ 个独立指数服务阶段；其中正在服务者的剩余时间利用无记忆性处理. 于是对 $z \geqslant 0$，

$$\mathbb{E}[e^{-zT}] = \sum_{n \geqslant 0} (1-\rho)\rho^n \left(\frac{\mu}{\mu + z}\right)^{n+1} \tag{31.11}$$

$$= \frac{\mu - \lambda}{\mu - \lambda + z}. \tag{31.12}$$

这是速率 $\mu-\lambda$ 的指数分布的拉普拉斯变换，故

$$\Pr(T>t)=e^{-(\mu-\lambda)t},\qquad t_{0.99}=\frac{\log 100}{\mu-\lambda}. \tag{31.13}$$

若 $\mu=100$ 请求每秒，$\lambda=80$，平均逗留 50 毫秒，p99 约 230 毫秒；到达率升至 95 时，二者分别为 200 和 921 毫秒．这个构造只改变到达率，就产生明显尾部增长．

关键结论

批处理的请求共享服务轮次，长输出又使驻留长度分布发生变化．不能把上述 μ 换成一次离线词元吞吐，就声称获得了真实服务 p99．实际系统应使用按长度、阶段和负载测得的服务曲线，必要时以到达轨迹驱动离散事件模型，并用独立观测校准．

31.5　准入、期限及背压

准入控制（Admission Control）发生在承诺接受任务的边界．至少要核查模型与租户配额、请求体与长度上限、缓存增长、队列空间和期限．可以用

$$\widehat{W}_i+\widehat{S}_i+\Delta_i\leqslant d_i-a_i \tag{31.14}$$

作为风险判断，其中 Δ_i 为估计误差和协议开销余量．分位数也不能随意相加得到相同分位数的总和；要么直接估计联合总时延，要么使用明确的概率上界．[2]

背压（Backpressure）把下游容量不足传递给上游．它必须贯穿入口、有界等待队列、活动序列和输出缓冲．如果只限制设备并发，慢客户端仍可能使主机缓冲无限增长．若输出缓冲以每秒 r_g 字节生成、每秒 r_c 字节消耗，则积压近似按 r_g-r_c 增长；当 $r_g>r_c$，容量 H_b 的空缓冲将在 $\dfrac{H_b}{r_g-r_c}$ 秒后填满．持续消费不足时应暂停、取消或终止，并传播资源释放．

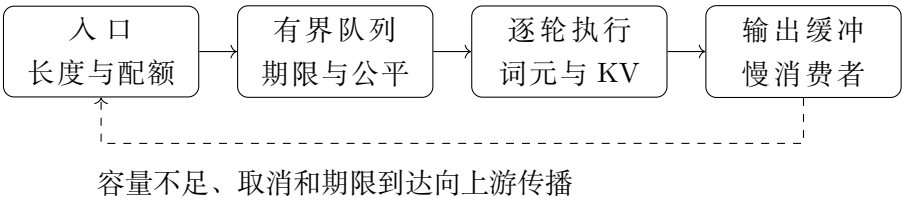


图 31.2: 每一层均有有限预算，释放信号与容量信号形成反馈．

突发到达时，队列吸收的是暂时差额，而不是永久不足．若一段时间内输入速率为 λ_b、有效完成速率为 $\mu_b<\lambda_b$，持续 τ 秒，则新增积压粗估为 $(\lambda_b-\mu_b)\tau$．这是流体近似，忽略随机波动；必须另外考虑突发形状和请求工作量．扩容有加载延迟时，不能把尚未就绪的设备计入即时服务能力．

[2]若 $\Pr(W>w)\leqslant\epsilon_1$、$\Pr(S>s)\leqslant\epsilon_2$，则不要求独立也有 $\Pr(W+S>w+s)\leqslant\epsilon_1+\epsilon_2$．这来自并集界，而不是分位数可加性．

31.6　公平性及可恢复的抢占

31.6.1　公平调度的分配单位

先到先服务（First-Come, First-Served，FCFS）简单且可解释，但长请求可能造成**队首阻塞**（Head-of-Line Blocking，HOL Blocking）. 按最短作业优先在已知独立作业时长等条件下改善平均等待，但输出长度未知，估计误差和长作业饥饿会削弱其适用性.

租户公平可以按请求数、词元数或计算成本定义. 每请求一票会偏向长请求，每词元一票又忽略长上下文解码与短上下文解码的成本差异. 应选可测量、能核算的成本单位，并公开权重的业务含义.

一种实现是为租户 j 维护信用额 D_j，每轮增加 w_jQ，执行后减去计费成本 c. 信用不足的候选等待后续轮次；允许有限结转可以容纳较大任务，但应限制无限积累造成的突发. 预测成本用于选择，实际成本用于事后纠偏，二者差额不能长期由其他租户承担. 共享批次成本难以精确归因，实践中需固定一致的分摊口径，而不能声称存在唯一物理真值.

老化（Aging）随等待增加优先级，可缓解饥饿，但不能在持续超载下保证每个请求都按时完成. 截止时间优先也不是无条件最优：本系统同时具有缓存驻留、多资源约束和不可抢占设备片段，不满足许多经典单处理器期限结论的前提.

31.6.2　抢占成本

抢占（Preemption）应发生在安全的迭代边界. 已完成输出保持不变，未提交候选与执行状态按明确规则处理. 若丢弃缓存，恢复需重算已提交前缀；若换出，则需传输及恢复页映射，并确认模型修订、位置和缓存格式一致.

设重算成本为 C_r，换出和换入共需移动 $2m$ 字节，有效传输带宽为 b，另有同步成本 C_s. 粗略比较 C_r 与 $\dfrac{2m}{b}+C_s$ 有助于选择，但链路竞争与恢复时设备占用还可能改变结果. 频繁抢占会产生反复重算，使观测到的忙碌增加而有效完成减少. 应记录重算词元与原始词元的比例，并设置冷却或最小执行份额.

算法 31.1　计算过程

单轮有界调度. 输入为等待队列、活动请求、物理页状态、租户信用和本轮预算.

1. 在安全边界处理完成、取消和到期事件，释放引用与预留；已发布终止状态不得重新入队.
2. 增加各租户信用，更新活动解码与等待预填充的期限、年龄和成本估计.
3. 为交互解码与等待预填充分别保留最低服务份额；按权重、期限及年龄选择候选.

4. 对候选确定 q_i，同时检查新词元、序列槽、物理页、工作区和轮时预算．无法容纳时考虑其他可行候选，避免无限队首阻塞．

5. 对本轮计划原子预留页与状态版本．若预留失败，撤销该计划并重新选择，不以部分成功的预留直接执行．

6. 执行一轮；按实际进度提交缓存和输出，按实际计费规则扣减信用，保留尚未完成者．

7. 记录排队、批形状、资源、重算和输出阻塞；若不存在可行计划，等待资源事件或执行明确的拒绝、抢占策略，禁止空转．

31.7　尾延迟及有效容量

请求级 TPOT 平均值会掩盖单次长停顿，也会与把所有 ITL 汇总后的均值不同：后者让长输出拥有更大权重．应分别报告请求级分布与词元级分布，并说明权重．输入长度、输出长度、缓存命中、租户和模型修订都可能形成不同的时延切片．

开放环负载（Open-Loop Load）按独立于完成速度的计划提交请求，能够显露过载积压；**闭合环负载**（Closed-Loop Load）在固定并发用户收到响应后再提交，其到达率会在服务变慢时自行下降．两种模型都可有意义，但回答不同问题．若原目标是固定外部到达流，测试却在每个响应后才继续，就会遗漏拥塞期间本应到达的请求，这种测量偏差常称为**协调遗漏**（Coordinated Omission）．不能只用闭合环的低尾延迟证明外部突发下的稳定性．

将满足质量与时延目标的成功请求数除以观察时长，可定义该目标下的**有效吞吐**（Goodput）．例如一个构造性窗口内共提交 1000 个请求，拒绝 100 个、失败 50 个、成功 850 个，其中 700 个满足所有目标，窗口 100 秒，则提交率 10、成功吞吐 8.5、有效吞吐 7 请求每秒．若只在成功者中报告满足率，得到 $\frac{700}{850}$；若从所有提交者看，则是 70%．二者必须注明分母．

调度策略比较应保持模型、生成参数、输入轨迹与输出预算一致；随机生成导致工作量不同，应多次重复或使用可解释的受控长度实验．报告原始到达计划、被接纳请求、失败与取消，不把超时样本删除后再计算漂亮的 p99．质量变化带来的短输出也不能冒充纯调度加速．

第三十二章　模型服务架构

模型服务不是把一次前向传播包装成网络函数. 它需要在并发、取消、故障和版本变化下，持续保证请求使用正确模型，输出具有明确终止语义，资源最终被释放. 部署架构的核心是将概率计算嵌入可解释的状态管理与变更流程.

第三十一章《批处理调度》讨论排队与批调度，本章讨论组件职责、制品身份和服务生命周期. 第三十章《增量解码》定义请求执行状态；这里进一步研究这些状态在网关、工作进程和控制面之间如何保持一致.

32.1　制品身份及运行配置

32.1.1　模型制品构成

模型制品（Model Artifact）是共同定义模型计算行为的一组版本化对象，包括权重、结构配置、分词器、特殊词元、会话模板、适配器及量化参数. 对于多模态模型，还包括处理器与相关编码器. 模型名称或目录路径不足以唯一重建行为，因为它们可能指向不断变化的内容.

不可变制品（Immutable Artifact）以固定内容与摘要标识. 一份清单将所有组成文件及其依赖关系绑定在一起；改变模板或适配器组合，也应形成新的身份. 内容摘要保证比对字节一致，不自动证明来源真实性；来源验证可以由受控注册表、签名和构建记录补充.

运行配置描述制品怎样部署，例如精度、并行布局、内核、设备类型和最大资源预算. 流量比例、副本数、租户配额与路由策略属于服务控制配置. 三者分别版本化，可以让同一模型制品在不同环境晋级，而不把环境名称混进权重包.

32.1.2　制品兼容性

加载成功只说明当前运行时能够解释某种文件布局，不保证配置组合正确. 张量并行需要检查查询头、键值头及层维度的分片方式；键值复制会改变每卡容量；量化格式需要对应内核；适配器需要匹配基座修订. 设备总数满足并行度，并不证明这些关系成立.

服务修订可以表示为

$$\mathcal{R} = (a, e, c, p), \tag{32.1}$$

a 是制品身份，e 是运行时环境身份，c 是执行配置，p 是接口与策略版本．请求进入执行前固定 $\mathcal{R}$，直到完成或明确终止．发布新修订影响后续路由，不应让正在解码的请求突然读取新权重或新模板．

32.1.3　符号及适用范围

符号	含义
t_a, t_f, t_l	请求进入测量边界、首输出与末输出时刻．
d_r, Δ	请求截止时刻与当前剩余时间．
N_o, M_{KV}, u_{net}	输出词元数、需迁移缓存字节数与有效传输带宽．
R, q, u	就绪副本数、给定负载下单副本容量与目标利用系数．
$T_b, D(t), Q(t)$	副本启动提前量、需求速率与有效总容量．
A_{retry}	请求的期望尝试次数，即重试放大系数．

服务分析的口径

容量与延迟均针对固定制品、硬件、运行时和请求长度分布．构造数例只演示计算关系，不代表压测结果．请求状态有明确所有者，网络消息可能延迟、重复或丢失；跨进程状态更新不被假定为天然原子．

32.2　服务拓扑及组件分工

32.2.1　数据面及控制面

数据面（Data Plane）处理请求与输出；**控制面**（Control Plane）管理期望副本、修订、路由与生命周期．前者追求低延迟与局部故障可隔离，后者追求变更一致与最终收敛．把每次请求都同步绑定到注册表或配置中心，会把控制服务短暂不可用放大成全部推理不可用．

网关验证身份、请求大小、配额与协议；路由器解析模型别名并选择合法副本；调度器管理排队、执行批次和资源准入；工作进程加载模型、执行计算和维护缓存．它们可以在小规模系统中合并部署，但职责及状态所有权仍需清楚，不能因进程合并而混淆语义．

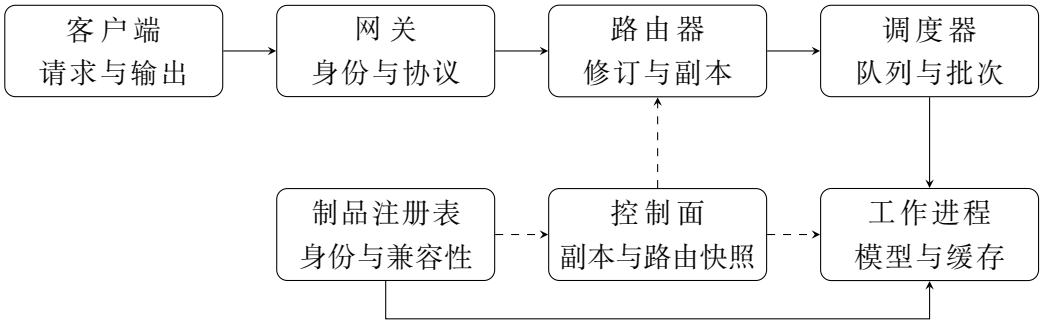


图 32.1: 模型服务的职责拓扑. 实线表示请求或制品数据路径，虚线表示控制配置；流式输出沿请求链路返回.

32.2.2 路由快照及请求粘性

路由器可以使用已校验、可原子替换的本地配置快照. 快照包含修订、端点、健康与故障域信息. 旧快照允许在控制面短暂故障时继续服务，但其可用期限、失效策略和撤销机制必须定义，不能无限信任过期授权或已退役副本.

一个请求固定模型修订和执行所有者；有状态的解码需要访问对应缓存. 如果迁移，应采用显式状态转移协议，而不是每个词元重新随机选择副本. 相同会话的不同请求是否需要粘性取决于状态是否可重建及前缀缓存策略，不能以用户身份直接绑定到永久单点.

32.2.3 故障域及共享依赖

故障域（Failure Domain）是可能共同失效的一组资源，如设备、节点、机架、可用区、修订或共享存储. 两个副本若共享同一块故障存储或同一错误配置，不一定形成独立冗余. 架构分析需要追踪共享依赖，而不只数进程数量.

租户或长短任务池的分隔可以限制资源争用；分池过细又会降低复用效率. 合理边界由故障影响和工作负载差异决定. 服务层不要求所有请求使用相同队列，跨池转移必须服从剩余容量和权限，不能让局部过载自动灌满健康池.

32.3 请求状态及流式协议

32.3.1 能力协商及有效推理强度

推理强度进入服务接口后，既是模型行为参数，也是容量维度. 网关应根据已经绑定的模型修订校验允许档位，并把请求值、默认值和最终生效值分别记录. 模型别名切换后，支持集合或默认档位可能改变；因此缓存键、幂等摘要、审计记录和负载归类都应包含最终生效的推理强度. 服务无法支持请求值时应在排队前返回明确错误，不能静默降档后仍保留原请求标签.

容量规划不能只按可见答案长度分组．内部推理会占用解码步骤，并与可见输出共享设备时间；某些接口还让二者共享输出词元上限．即使两次响应都只有相同长度的答案，不同推理强度也可能产生不同的服务时间、超时率和单位请求成本．单副本容量 q 应在固定模型、档位、输入长度、输出上限和工具配置的负载分布上测量，不能把低档位压测结果直接用于高档位流量．

对于会话式服务，运行中改变推理强度还可能影响前缀缓存或服务方保存的推理状态．是否能够复用已有前缀、参数从哪一轮开始生效，应由具体协议定义．OpenAI 当前接口在 Responses 与 Chat Completions 中使用不同字段，并对部分模型提供会话内配置更新；这些行为属于版本化接口能力，不应被推断为所有 OpenAI 兼容服务都支持 [142]．

32.3.2　请求状态机

请求通常经历接收、等待、执行和终止．执行可包含预填充、状态传输和解码；终止原因至少区分正常完成、取消、超时和失败．状态应记录单调版本或更新序号，以拒绝迟到消息覆盖更晚状态．

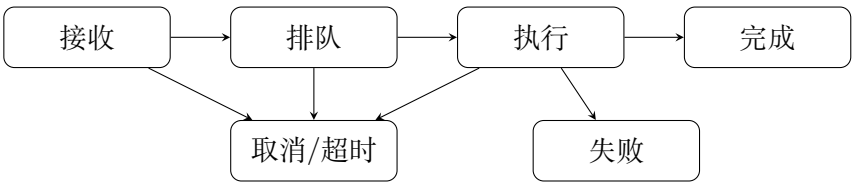


图 32.2: 请求状态机．终止状态不重新进入执行；重试必须具有明确尝试身份与既有请求的关联．

取消和正常完成可能同时到达．系统应通过条件状态更新选择一个终止结果，例如只允许从非终止状态写入终止状态；输掉竞争的事件读取已经确定的结果．资源释放还需使用独立的幂等所有权记录，避免两个处理路径都递减缓存引用．

32.3.3　流事件及终止语义

服务端发送事件（Server-Sent Events，SSE）定义了单向文本事件流机制 [171]，适合服务器连续发送增量内容．应用仍需定义事件类型、请求身份、序号、修订、结束原因和用量．传输机制不自动提供生成任务的完整状态协议．

增量事件可能是多个词元形成的文本片段，也可能包含结构化调用字段．不能假定一个网络事件恰好对应一个模型词元，更不能假定分词字节片段已经构成完整字符．解码和协议组装应保证客户端看到合法的增量内容．

正常终止事件表示生成按指定原因结束；连接关闭或长时间无消息只表示传输状态不确定．响应头已经发出后，服务器通常不能再用一个新 HTTP 状态替换当前响

应，因此应用流内错误与终止原因必须可识别．最终用量记录也要区别已计算、已发送和已确认消费的内容．

SSE 的事件身份与重连机制不能自动重放语言模型计算．若服务支持恢复，需要持久化事件序列或完整生成状态，并声明保留窗口；否则重连只能查询既有请求状态或发起新请求．[1]

32.3.4 取消、期限及剩余预算

截止期限（Deadline）约束从请求开始到终止的总时间．若请求截止时刻为 d_r，当前时刻为 t，剩余预算为

$$\Delta = d_r - t. \tag{32.2}$$

进入队列、状态迁移、重试和执行前都应重新检查，不能为每一跳重新分配完整超时．跨机器绝对时钟存在偏差，需要统一时间约定与容差，或传播明确的剩余预算并计入传输开销；进程内计时宜使用单调时钟．

客户端断连不保证设备计算立即停止．取消信号需要抵达调度与执行所有者，阻止后续生成，并在设备安全边界释放资源．一个已启动的内核可能需要完成，故取消生效时间应区分“停止接收新工作”和“全部设备资源可回收”．

32.3.5 幂等及重试边界

幂等性（Idempotency）指重复执行具有相同预期效果，HTTP 对相关语义有明确区分 [172]．随机生成使用同一请求内容，不天然给出相同输出；相同随机种子也不跨任意设备和执行顺序保证逐字一致．

应用幂等键可将同一逻辑请求映射到已有记录．该记录应绑定请求内容摘要、租户和修订：相同键但不同内容不能静默复用．幂等保留期过后，服务如何处理再次到来的键也应明确．

重试安全性取决于输出与副作用是否已经发生．尚未输出且旧执行可确认终止时，可以按预算重试；旧执行状态未知时，新的尝试可能形成重复计算．已经输出部分文本后静默重新生成，可能出现重复或分叉，通常应返回中断或按显式恢复协议续传．涉及工具写操作时，工具自身还需幂等或事务机制，模型请求幂等不能自动保护外部副作用．

[1] 网络层可能重复投递应用事件．客户端按请求身份与单调事件序号去重，可防止显示重复文本；它不等价于证明服务器只计算了一次．

32.4　工作进程的生命周期

32.4.1　三类健康信号

模型工作进程会经历创建、加载、预热、就绪、排空和终止．**启动探针**（Startup Probe）判断初始化是否完成；**存活探针**（Liveness Probe）判断进程是否还能推进；**就绪探针**（Readiness Probe）判断是否可以承接新流量．Kubernetes 文档明确区分这些信号 [173]．

加载权重耗时较长，不应因尚未完成启动而反复触发存活重启．队列增长或暂时过载也不等于进程死亡；过敏的存活策略会在最需要容量时制造重启．就绪信号可以在排空、关键依赖失效或容量不可接受时撤销，而不必销毁进程．

预热不仅是调用一次模型．它应覆盖目标执行形状、缓存格式和关键内核路径，形成启动资源边界．预热成功证明特定路径可执行，不证明真实长度分布的容量或质量达标．

32.4.2　优雅下线

排空（Draining）先停止新流量，再处理已有请求，最后释放模型．路由更新存在传播延迟，因此工作进程本地也应拒绝新接纳或返回明确重选信号，避免只依赖远端端点更新．

排空期间保留已有请求的修订与缓存．宽限期不能无限延长；超过预定截止后取消剩余请求，并报告结束原因．关闭连接、释放设备缓存、写入最终用量和终止进程之间要有明确顺序．缩容控制器应将排空副本从未来容量中扣除，而不是等进程真正退出才发现容量已经减少．

算法 32.1　工作进程的上线与下线

输入为不可变服务修订、目标资源与期限；输出为可接流量的实例或明确失败结果．

1. 校验制品与执行配置兼容性，拉取完整文件并验证身份；失败则保持流量隔离．
2. 加载权重与必要处理器，分配运行资源，完成指定路径预热；完成前不发布就绪状态．
3. 发布就绪并承接请求，持续区分存活、可用容量和关键依赖状态．
4. 收到维护或缩容指令后撤销就绪，本地停止新请求，保留在途执行与取消处理能力．
5. 等待在途请求完成或达到宽限期，终止剩余请求，确认设备资源释放并冲刷

必要状态后退出.

不变量是未准备完成的副本不承接正常流量，进入排空后不扩大在途请求集合.

32.5 预填充及解码分离

32.5.1 分离的动机及条件

预填充—解码分离（Prefill–Decode Disaggregation，P/D 分离）把两个计算阶段分配到不同资源池. 预填充处理较长输入，解码反复生成少量位置；二者对并行度、批次和带宽的需求不同. DistServe 研究了这一分离及其面向阶段延迟目标的资源分配 [174].

分离可以降低同一设备上阶段相互干扰，并独立调整资源比例，但新增了缓存传输、网络排队和状态所有权切换. 小模型、短提示或有限网络下，新增成本可能超过收益. 它是架构选择，不是所有服务必须采用的固定形态.

32.5.2 缓存传输代价

从预填充节点传送 M_{KV} 字节到解码节点，有效带宽 u_{net}，则无重叠的传输时间至少为

$$T_{\text{transfer}} \geqslant \frac{M_{KV}}{u_{net}} + T_{\text{setup}}. \tag{32.3}$$

T_{setup} 包含必要连接、描述符与同步开销. 协议、拥塞、布局转换和接收写入会增加实际时间. 若部分传输与计算流水重叠，端到端新增时延不能简单等于二者相加，但总网络工作量仍然存在.

例如缓存每词元 128 KiB、提示 8192 词元，则 $M_{KV} = 1$ GiB. 有效带宽 25 GiB/s 时，仅字节传输下界为 40 毫秒；若每秒接收 50 条这种请求，就产生 50 GiB/s 缓存流量，单条 25 GiB/s 链路无法长期承载. 缩短单请求传输延迟与满足总带宽预算是两个不同问题.

32.5.3 迁移所有权

需迁移的不只是缓存字节，还包括逻辑长度、页映射、模型修订、位置规则、已生成词元、采样状态和输出序号. 若预填充端已经产生首词元，解码端必须知道该词元是否已经执行前向及是否已向客户端发送，避免重复生成或漏处理.

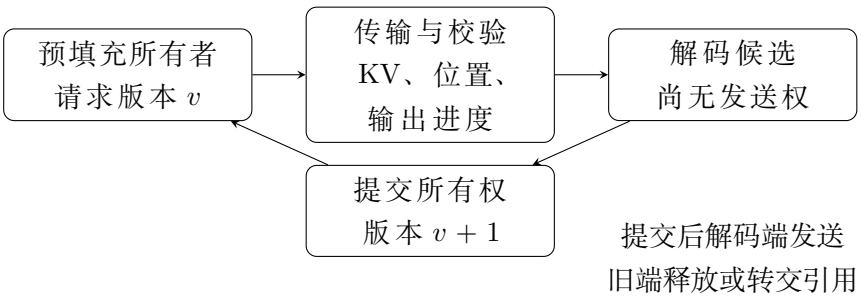


图 32.3: P/D 交接采用准备、校验和所有权提交. 数据已到达不等于目标节点已经拥有输出权.

可以通过请求版本或租约令牌保证同一时刻只有一个合法输出所有者. 接收端准备完成后再提交迁移; 提交前失败仍由旧所有者处理, 提交后旧端不得继续发送. 跨故障域时 “提交是否成功” 的不确定性需要查询权威状态, 不能靠双方同时重试解决.

P/D 两端可以使用不同并行布局, 但必须有兼容的逻辑缓存表示和明确重分布路径. 混合不同模型修订、适配器或位置配置不属于普通缓存搬迁. 安全边界、网络加密与授权也要随状态转移保持, 不能因为数据位于内部网络就忽略租户身份.

32.6　扩缩容及启动提前量

32.6.1　分阶段容量模型

副本容量取决于输入、输出长度分布及阶段组合. 将输入词元与输出词元相加, 再除以一个吞吐常数, 只能用于混合比例固定时的粗估; 更稳妥的方法分别考虑预填充工作、解码工作和缓存占用, 再映射到阶段池或共置实例.

在假定某个固定负载下单副本容量为 q、目标系数 u 时, 粗略就绪副本需求为

$$R_{\mathrm{need}} = \left\lceil \frac{D}{uq} \right\rceil . \tag{32.4}$$

这只是部署资源估计, 批调度与尾延迟模型由第三十一章《批处理调度》展开. 尚在加载、预热或排空的副本不能算作完整可用容量.

32.6.2　服务容量测算

设一个固定长度混合下, 单副本容量假设为 400 请求/分钟, 规划利用系数 0.75, 当前 6 个就绪副本提供 1800 请求/分钟. 需求由 1800 突增到 2400, 粗略需要 8 个副本; 新增副本从创建到就绪需要 90 秒.

这 90 秒内, 即使扩容指令立即发出, 仍有每分钟 600 个请求的容量缺口, 累计约 900 个请求. 如果排队预算只允许容纳 300 个请求, 则剩余 600 个必须通过提前

预热、流量分担或明确拒绝处理. 扩容能解决未来容量，不能消除启动期间已发生的缺口.

一般地，启动窗口中的累计正容量缺口可写为

$$C_{\mathrm{gap}} = \int_{t}^{t+T_b} [D(\tau) - Q(\tau)]_+ \, \mathrm{d}\tau, \tag{32.5}$$

其中 $[x]_+ = \max(x, 0)$. 在无丢弃的流体队列中，期末积压增量满足 $B(t+T_b) - B(t) \leqslant C_{\mathrm{gap}}$；只有整个窗口需求不低于容量等条件下，正缺口才等于积压增长. 窗口内的富余容量可以消化先前积压，因此不能把正部积分一般地当成期末增量. 上述 900 请求算例具有恒定正缺口，故满足该条件. 这个模型忽略离散请求与批次耦合，不直接给出尾延迟达标概率.

32.6.3 控制环的滞后及抖动

自动扩缩容包括观测窗口、聚合、决策、资源配置和启动. 过长观测窗口会延迟响应，过短窗口可能追随噪声. 新增容量到达前重复发出扩容，又可能造成过量资源；控制器需把正在启动的容量计入预测，而不是计入当前供给.

缩容应比扩容更谨慎，使用稳定窗口和排空机制；同时保留必要热容量以覆盖突发与故障. 若资源紧张，扩容请求本身可能等待调度，因此启动提前量应包含基础设施供给时间，而不只是模型加载时间.

32.7 模型路由及服务降级

32.7.1 模型选择及副本选择

模型路由（Model Routing）依据请求特征、质量要求和资源状态，为一次请求选择执行模型. 商业服务可以在统一入口后配置多个模型池，把适合的请求交给成本较低的模型，将较强模型的容量留给更需要它的任务. 用户所说的服务“降智”可能指其中的跨模型降级，但回答质量下降本身不能证明模型被替换.

模型路由应与同一模型内部的副本选择分开. 前者可能改变任务能力，后者主要改变执行位置；同名模型跨供应商运行，也需要核对实际修订、量化和模板. 另一个维度是计算预算：同一模型减少推理词元、候选数量或工具调用，也可能改变成功率. 因而执行决策应同时固定模型身份、供应商、运行配置和预算，而不能只记录一个产品名称.

这里的“较强”与“较弱”相对于具体任务和评估分布成立. 较小模型可能在特定领域优于较大模型，价格与参数量都不能直接替代质量测量. IPR 研究采用质量估计和可调容忍度进行模型选择 [175]；它提供了公开的路由方法实例，其论文实验不能证明任意请求都保持原模型质量，也不能用来推断其他服务存在未披露替换.

32.7.2　带约束的路由决策

记请求为 x，当前服务状态为 s，候选模型为 m，配置预算为 b. 先根据用户授权、数据驻留、模态、上下文长度、结构化输出和工具协议筛选合法集合 $\mathcal{A}(x)$. 在此基础上，一种设计目标是

$$(m^*, b^*) = \operatorname*{arg\,min}_{(m,b)\in\mathcal{A}(x)} \widehat{C}(m, b \mid x, s), \quad \text{满足} \quad \widehat{Q}(m, b \mid x) \geqslant q_{\min}, \quad \widehat{T}(m, b \mid x, s) \leqslant \Delta. \tag{32.6}$$

$\widehat{C}$ 是预计总成本，$\widehat{Q}$ 是同一评价口径下的预计质量，$\widehat{T}$ 包含路由、排队、生成和必要验证的预计时延，Δ 是剩余截止预算. 该式是本节的工程建模，不是对某个产品算法的复现. 预测约束只能作为决策依据，不能构成逐请求质量或期限保证；尾延迟要求还需使用相应分位数或超时风险估计.

质量估计器可以利用任务类型、完整上下文特征以及离线标注结果. 训练数据应覆盖各候选模型在同类请求上的表现，并按来源或任务划分训练、验证和测试集合. 只根据问题长度判断难度容易漏掉短问题中的复杂依赖；只把小模型的自信陈述当作可靠性，也无法识别自信错误. 模型、模板或流量分布变化后，应重新验证估计器与阈值.

合法集合为空，或预测没有候选满足门槛时，应按契约排队、拒绝或显式请求调整服务等级. 容量压力不能自动扩大数据访问权限或跨越用户限定的模型范围.

32.7.3　预先选择、级联及故障回退

请求前路由在生成之前选择模型，只承担一次选中模型的生成成本，但选择时看不到真实答案. **级联调用**（Model Cascade）先让低成本模型生成候选，再根据验证结果决定是否升级；检查可以包括可执行测试、结构约束、证据核验或经校准的质量评分. 检查通过仅表示满足已检查条件，不能把格式正确当作事实正确.

设小模型生成成本为 C_s，验证成本为 C_v，升级比例为 p，升级后大模型调用的平均成本为 C_h. 在固定请求分布及忽略其他开销的条件下，级联平均成本为

$$\mathbb{E}[C_{\text{cascade}}] = C_s + C_v + pC_h. \tag{32.7}$$

若直接调用大模型的平均成本为 C_0，级联节省成本的条件为

$$p < \frac{C_0 - C_s - C_v}{C_h}, \qquad C_h > 0. \tag{32.8}$$

例如假设 $C_s = 1, C_v = 0.5, C_h = C_0 = 10$，升级比例为 0.3，平均成本为 4.5 个单位，相对基线降低 55%. 这是构造算例；升级时携带候选或额外证据会改变 C_h，而升级请求通常还需承担两次串行生成和验证的时延. 级联中的小模型候选在验收前应保留在服务内部，避免先流出错误内容再静默替换.

服务降级（Service Degradation）是在资源不足或故障时按预定契约降低服务等级．过载降级依据队列、容量或期限风险触发；故障回退依据超时、限流或不可用等执行结果触发．备用模型可能同等或更强，回退不必然意味着能力下降．应限制尝试次数与总预算，并确认旧尝试状态，防止重复计算或重试流量继续压垮备用池．跨模型尝试需要重新序列化请求并建立匹配的新缓存，不能复用另一模型的 KV 状态．

32.7.4 容量约束下的模型路由

考虑提供固定模型与自动模型两种模式的服务．固定模式将用户指定修订作为约束；自动模式允许在事先公开的候选集合内选择．以下流程是一种设计案例，阈值应从实际负载与质量评估确定．

算法 32.2 容量受限时的模型选择

输入为请求、服务契约、路由快照、质量门槛和剩余期限；输出为可追溯响应或明确失败．

1. 解析模式、授权候选和能力要求，筛选合法模型池；记录策略版本和原始请求身份．
2. 检查大模型队列与预计完成时间．固定模式只在允许的同模型部署间选择；容量不足时有界排队或报告不可用．
3. 自动模式下，若大模型拥堵，评估较低成本候选的质量和剩余容量；同时满足质量门槛、协议要求和时间预算才分流．
4. 没有合格替代时按契约排队或失败；选择级联时把首轮、验证和升级全部计入期限，并保留升级容量预算．
5. 为每次尝试固定模型修订与配置．已经对外发送文本或执行工具写操作后，不静默拼接另一模型的结果；采用明确中断或具有副作用保护的恢复流程．
6. 记录实际执行身份、降级原因、用量和终止结果．大模型恢复后，经过稳定窗口逐步恢复分配，避免阈值附近反复切换．

不变量是路由始终位于授权候选集合内，每次执行具有唯一身份，失败与降级对调用方可辨识．

分流减轻的是某个资源池的压力，并不创造总容量．备用池也需要接纳控制；若两个模型共享同一紧张设备，分流收益取决于实际计算与驻留开销．高峰期还应分别统计正常路由与过载降级的任务分布，判断质量变化来自策略、输入变化还是执行故障．

32.7.5　质量门槛及服务透明性

路由评估应在相同题集、上下文、评分规则和预算口径下，与固定模型基线比较. 除平均质量、成本和尾延迟，还要按语言、任务难度、上下文长度与工具类型报告结果，并记录小模型覆盖率、升级比例和错误放行情况. 错误放行可定义为“小模型候选被验收但最终判错”，其比例必须明确以放行请求还是全部请求为分母. 高风险任务单独设门槛，不能由大量简单任务的正确答案抵消.

只观察已经路由到小模型的请求，无法直接知道同一请求在大模型上的结果. 可以在合规抽样集上补做配对评估，但须计入额外成本并隔离工具副作用. 第九章《模型评估》的抽样、不确定性和分层比较方法适用于这里；根据测试结果反复调路由阈值后，该测试集就需要作为开发数据处理.

服务契约应说明固定身份还是自动选择、可用候选、降级条件、计费口径和退出降级的方式. 调用记录至少关联请求身份、尝试身份、请求模型、实际模型修订、供应商、策略版本、执行预算及降级原因；敏感上下文按最小必要原则保存. 模型自报身份、单次回答文风与速度不足以证明实际执行模型，代理返回的模型字段也只是声明. 较强证据来自可关联的网关、执行端和上游记录，并以访问控制及防篡改措施支持审计.

模型路由的收益应在授权与质量约束下计算，实际交付的模型和计算预算应当可追溯.

32.8　发布及回滚

32.8.1　候选修订的暴露方式

金丝雀发布（Canary Release）让候选接收有限真实流量，观察质量与系统指标. **影子流量**（Shadow Traffic）复制请求到候选路径但不返回其结果，适合比较资源与离线质量；它不能完全代表真实用户体验，也会增加计算和数据处理成本. 包含副作用的工具请求不能直接在影子路径重复执行.

蓝绿部署（Blue–Green Deployment）保留两套环境，通过路由切换晋级或回退；它需要额外驻留容量. 滚动替换则逐步更新副本，节省部分双份资源但延长新旧共存时间. 几种方式可以组合，选择依据是资源预算、回滚速度与协议兼容性.

32.8.2　质量及系统联合判断

候选可能更快却更容易错误，也可能质量提高但尾延迟超标. 发布标准应分别约束任务质量、故障率、停止行为、延迟及资源；不能把所有维度压成一个分数后让速度抵消严重错误.

灰度样本的语言、租户和长度可能偏离整体流量，观察窗口应记录这些切片及样

本量．新旧版本的随机生成比较还需固定评估设计，不能仅选择看起来较好的单次输出．指标异常时应能够追溯实际修订和配置，避免把同时发生的流量变化误归因于模型．

32.8.3 回滚恢复契约

回滚不只是恢复权重，还包括模板、运行配置、路由与兼容依赖．稳定修订若已被删除或冷却，路由回退也不会立刻恢复容量．发布计划应保留足够可用旧容量及可拉取制品．

新请求可以切回稳定修订，在途请求应按原修订完成或明确终止．若会话外部状态或工具协议已发生不兼容写入，模型回滚无法逆转这些变化，需要兼容读取或独立迁移策略．缓存也按修订隔离，不能在回滚时混用候选状态．

算法 32.3 候选服务修订的晋级

输入为候选与稳定修订、评估标准和流量预算；输出为晋级结果或恢复稳定状态．

1. 固定候选制品与配置，完成兼容性准备，建立可追溯评估基线和稳定容量保留策略．
2. 加载候选并预热，在隔离路径验证真实请求生命周期、错误和资源释放．
3. 按允许的数据范围进行影子或小流量比较，记录模型、长度、任务与租户切片．
4. 仅在质量和服务指标同时满足预定条件时逐步扩大；任一关键条件失败即停止新增候选流量．
5. 回退时原子更新路由，新请求固定稳定修订，候选排空或按期限终止；保留失败证据与完整版本组合．

整个过程不修改已发布制品，也不在生成中途静默切换模型身份．

32.9 遥测及关联

32.9.1 三种信号的用途

可观测性（Observability）通过外部信号推断系统状态．指标描述聚合趋势，日志记录状态变化，分布式追踪联系请求跨组件的时间路径；OpenTelemetry 将这些作为不同信号组织 [176]．

指标适合回答错误率和队列是否异常，追踪适合定位时间花在哪一跳，日志适合解释一次取消或迁移为何发生．请求身份、追踪身份、修订和工作进程身份应能够关

联，但不必全部作为指标标签．遥测服务故障时，推理路径也应保持有界缓冲，不能无限积压日志耗尽内存．

32.9.2　时间测量边界

设 t_a 为请求进入约定入口，t_f 为首词元对观察者可见，t_l 为末词元可见，输出词元数 $N_o > 1$．则

$$\mathrm{TTFT} = t_f - t_a, \qquad \mathrm{TPOT} = \frac{t_l - t_f}{N_o - 1}, \qquad T_{\mathrm{e2e}} = \mathrm{TTFT} + (N_o - 1)\mathrm{TPOT}. \tag{32.9}$$

这里首词元延迟已经包括该边界内的排队；不能另加排队项．单词元输出没有相邻间隔，TPOT 不应人为记为零．客户端指标包含网络与刷新，工作进程指标则可只描述设备阶段；二者应分别命名．

分位数不能跨副本直接平均得到全局分位数，应合并适当分布信息后计算．小样本 P99 很不稳定，报告需要窗口与样本量．只统计成功请求也会产生选择偏差，超时、取消和失败应独立计入服务结果．

32.9.3　标签基数的乘法增长

标签基数（Label Cardinality）描述指标维度的不同取值数．若模型、区域、状态分别有 m, r, s 个取值，理论组合上限为 mrs；再加入每请求唯一身份，时序数量可能随请求总数增长．

例如 20 个模型、3 个区域、6 种状态形成最多 360 种组合，加入 100 万个请求身份后理论组合空间大幅扩大．不是所有组合都会出现，但唯一身份仍会使存储与查询成本失控．请求和用户信息更适合进入受控日志或追踪，指标保留有限集合标签．

提示、输出与缓存摘要也可能暴露敏感内容或形成可关联标识．通常记录计数、长度、错误类别与受控引用即可；调试原文需要单独的访问、采样和保留规则，不把普通遥测变成无界训练数据副本．

32.10　故障、背压及重试放大

32.10.1　资源不足传播

背压（Backpressure）使下游容量约束反馈到上游，防止无界接纳．网关预算、有限队列、执行容量和流式发送缓冲是不同边界．慢客户端可能使发送缓冲增长，即使设备仍能生成；继续无界计算会同时占用缓存和网络内存．

过载拒绝需要稳定错误语义及有限重试提示．降低输出预算、切换模型或关闭某些功能属于行为变化，应向调用者明确，而不是静默改变服务承诺．扩容尚未完成时，保护已接纳请求往往需要主动限制新流量．

32.10.2 重试放大效应

假设每次尝试独立失败概率为 p，最多允许 k 次总尝试，则期望尝试数为

$$A_{\text{retry}} = 1 + p + p^2 + \cdots + p^{k-1} = \frac{1 - p^k}{1 - p} \quad (p \neq 1). \tag{32.10}$$

例如 $p = 0.5, k = 3$，平均尝试次数 1.75，原始 100 请求/秒会形成 175 次尝试/秒. 若故障来自过载，失败概率并不独立，额外尝试反而可能继续推高 p，形成正反馈.

若网关和客户端各自允许三次尝试，最坏可形成九次下游尝试. 重试预算应跨层协调，并受同一个截止期限控制. 退避和随机抖动减少同步冲击，但不能创造容量；不可恢复配置错误、权限错误或已部分输出请求不应按普通暂态失败重试.

32.10.3 故障恢复及状态重建

工作进程失联可能丢失设备缓存. 若尚未输出，可以在合法条件下重新计算；已经输出时，重放需要保存精确前缀、采样状态和事件进度，仍受数值与执行兼容性约束. 无法满足恢复协议时，应明确报告中断，而不是拼接另一条生成轨迹.

故障演练应观察检测、隔离、取消、容量转移和最终资源回收，而不只看进程重启成功. 整个故障域失效后，入口应按剩余容量限流，不能把全部原流量瞬间压给幸存副本. 恢复时间目标需要结合制品获取、预热和路由收敛，并预先定义可接受的数据或状态丢失边界.

服务正确性是状态一致性

制品身份决定计算含义，请求状态决定输出与资源所有权，副本状态决定可用容量. 故障与发布都在改变这些状态. 只有三者的迁移规则一致，模型计算才能持续构成可靠服务.

32.11 推理服务基准测试

制品、请求和容量对象需要分别关联版本、执行状态与资源责任. 构造记录只用于说明预算关系，不能代替线上压测. 工作进程内部的缓存、批处理和算子优化属于可选择的引擎能力，单一内核优化并不等同于完成部署.

32.12 推理引擎体系

32.12.1 引擎在服务系统中的位置

vLLM 与 SGLang 将模型加载、缓存管理、调度和设备执行组织成推理运行时，并提供服务接口 [177, 178]. TensorRT-LLM 围绕 NVIDIA 平台提供推理优化及运行

组件 [179]. 这些引擎位于工作进程及其调度路径中；完整服务还需要入口认证、流量控制、版本路由、监控和故障处理.

一条聊天请求通常经过协议解析、模板渲染与分词、调度接纳、预填充、逐步解码和流式返回. 词元流决定模型实际接收的条件，调度器决定何时执行，缓存管理决定中间状态如何保存. 应用层看到相似的 HTTP 接口，并不能说明三个环节使用了相同语义.

32.12.2　vLLM 缓存调度

vLLM 的发展与 PagedAttention 工作有关，其分页思想将请求的逻辑序列与缓存物理块解耦 [167]. 实际引擎还需要协调批处理、模型执行、采样和缓存生命周期. 应把算法论文中的机制与所用软件版本的实现分开核查；支持的模型、算子、量化和服务参数以相应版本为准 [180].

例如多个请求共享提示前缀时，复用已计算的缓存可以减少重复预填充；但解码仍要读取有效历史状态，缓存还占用内存. 若改变模型、适配器或输入位置语义，原有缓存未必有效. 前缀复用收益需要在真实共享比例和请求到达条件下测量，不能由缓存命中率单独推断端到端加速.

32.12.3　SGLang 结构化执行

SGLang 的代表机制 RadixAttention 以基数树组织可复用的前缀缓存，使多次模型调用之间的共享前缀能够参与缓存匹配与回收 [178]. 其运行时还涉及调度、并行和受约束输出等能力，实际组合需要核对版本、模型与后端支持. 存在相同自然语言片段，不代表模板渲染后的词元前缀也相同.

比较 SGLang 与 vLLM 时，应固定权重、分词器、输入词元、生成限制和请求分布，再分别比较冷缓存与可复用缓存的条件. 共享前缀占比不同、预热方式不同或默认生成配置不同，都可能改变结果. 受约束输出还需独立检验格式通过率与内容正确性，格式合法并不保证答案满足任务.

32.12.4　平台约束及选型依据

TensorRT-LLM 的部署选择需结合 GPU 架构、精度、模型实现和具体执行后端 [179]. 应核查所选路径的构建或加载成本、动态形状范围与版本绑定，而不把一种执行路径的要求推广到全部后端. 模型更新频率和制品准备时间也会影响总成本.

选择引擎时，先确认模型及其模态、量化、上下文和适配器需求得到支持，再建立质量一致的负载对照. 报告首词元延迟、逐词元延迟、端到端延迟、有效输出词元吞吐、峰值显存和失败比例，并保留尾延迟. 吞吐最优配置可能突破交互延迟约束；最终选择应依据目标服务等级下的有效容量与运维成本.

第三十三章　推理计算优化

缓存减少已经完成的计算，算子优化减少完成一次计算所需的数据移动与调度，推测解码则尝试在一次目标模型调用中确认多个输出．三者作用于不同成本项．判断一种优化是否成立，首先要确定它保留了什么：同一数学算子、同一概率分布，还是仅保留某项任务质量的近似水平．

第三十章《增量解码》解释历史状态的复用．本章先推导在线归一化与分块注意力，再讨论融合、编译和形状，最后证明精确推测解码的分布保持性质．低精度误差在第三十四章《模型优化》展开，请求级调度在第三十一章《批处理调度》展开．

33.1　计算、搬运及启动成本

本章主要符号见表 33.1．

表 33.1: 推理计算优化的主要符号．

符号	含义
T_q, T_k, d_h, d_v	查询数、键数、键头维数和值头维数．
Q, K, V	单头矩阵，形状分别为 $T_q \times d_h, T_k \times d_h, T_k \times d_v$．
z_j, v_j	某查询对第 j 个键的分数及对应值向量．
m, ℓ, u	已处理键集合的最大分数、缩放指数和、缩放值加权和．
F, Q_b, R, b	运算量、搬运字节数、有效计算速率和有效带宽．
p, q	给定前缀下的目标采样分布与草稿采样分布．
k, α	单轮草稿候选数及特定假设下的条件接受概率．

对一个执行片段，计算时间至少为 $\frac{F}{R}$，数据传输时间至少为 $\frac{Q_b}{b}$．两者可部分重叠，因此 $\max(\frac{F}{R}, \frac{Q_b}{b})$ 是简化下界，而不是全部延迟．主机调度、设备启动、同步和布局变换还会增加成本．小矩阵可能受启动时延限制，大矩阵可能受计算或带宽限制，不能只比较浮点运算数．

若某优化仅加速原总时间中的比例 f，该部分加速 s 倍，其他部分不变，则总加速比为

$$S = \frac{1}{(1-f) + \frac{f}{s}}. \tag{33.1}$$

例如某算子占 40%，即使加速 4 倍，总加速也只有 $\dfrac{1}{0.6+0.1}\approx 1.43$. 加入新的布局转换或同步后还会更低. 算子收益必须沿完整路径核算.

33.2　在线 Softmax 的充分统计量

33.2.1　分块归一化合并

单查询注意力输出为

$$o=\frac{\sum_j e^{z_j}v_j}{\sum_j e^{z_j}},\qquad z_j=\frac{q^{\top}k_j}{\sqrt{d_h}}+M_j. \tag{33.2}$$

M_j 表示允许的加性偏置或掩码. 将键分为两组后，各组独立 Softmax 的分母不同，直接相加或等权平均两组输出通常错误；必须知道每组在全局分母中所占的质量.

对非空有效键集合 A，保存

$$m_A=\max_{j\in A} z_j,\quad \ell_A=\sum_{j\in A} e^{z_j-m_A},\quad u_A=\sum_{j\in A} e^{z_j-m_A}v_j. \tag{33.3}$$

m_A,ℓ_A 是标量，$u_A\in\mathbb{R}^{d_v}$. 在线归一化通过更新最大值与归一化和，避免先遍历完整向量求最大值再单独计算分母 [181].

33.2.2　适配器合并定理

对不相交有效集合 A,B，令 $m=\max(m_A,m_B)$，则

$$\ell=e^{m_A-m}\ell_A+e^{m_B-m}\ell_B, \tag{33.4}$$

$$u=e^{m_A-m}u_A+e^{m_B-m}u_B. \tag{33.5}$$

因为 $e^{m_A-m}e^{z_j-m_A}=e^{z_j-m}$，第一行正好等于全体键的缩放指数和，第二行同理. 因此合并结果仍满足式 (33.3)，最终输出为 $o=\dfrac{u}{\ell}$.

只要每次合并保留这三个统计量，就可以按任意分块顺序处理同一键集合. 在精确算术下，合并对应集合并集，因而结果与分块方式无关；浮点舍入会受顺序影响. **不物化完整概率矩阵并不意味着省略某些键，也不意味着近似 Softmax.**

空块不能直接通过 $m=-\infty$ 与另一个空块相减来处理. 实现应使用有效标志：全掩码块不贡献统计量；首个非空块直接初始化. 若整个查询行没有合法键，数学分母为零，需要按模型接口明确定义输出或拒绝输入，不能让未定义值传播.

例 25.1

设分数为 $(\log 2, \log 1, \log 6)$，对应标量值为 $(1, 4, 5)$，前两个键组成块 A，第三个键组成块 B．有

$$m_A = \log 2, \qquad \ell_A = \frac{3}{2}, \qquad u_A = 3,$$
$$m_B = \log 6, \qquad \ell_B = 1, \qquad u_B = 5.$$

共同最大值为 $\log 6$，块 A 的缩放系数为 $\frac{1}{3}$，因此

$$\ell = (\frac{1}{3})(\frac{3}{2}) + 1 = \frac{3}{2}, \qquad u = (\frac{1}{3})3 + 5 = 6, \qquad o = 4.$$

完整计算为 $\dfrac{2 \times 1 + 1 \times 4 + 6 \times 5}{2 + 1 + 6} = 4$．两个块的局部输出分别为 2 和 5；直接相加得到 7，等权平均得到 3.5，均不等于正确结果，因为两块的全局指数质量分别为 3 和 6.

算法 33.1　单查询的分块精确注意力

输入：查询、按块读取的键值、可见性规则；**输出：**注意力值向量.

1. 初始化为“尚无有效键”，不对空集合最大值作算术运算.
2. 读取一块键值，计算分数与掩码．若无合法位置，跳到下一块.
3. 计算本块 (m_b, ℓ_b, u_b)；若是首个有效块，直接保存，否则按式 (33.5) 合并.
4. 处理完所有块后，若存在有效键，返回 $\frac{u}{\ell}$；否则执行预先定义的全掩码行处理.

33.3　FlashAttention 及存储层次

显式实现通常先写出 $T_q \times T_k$ 分数，再读出求 Softmax，最后读取概率矩阵计算与 V 的乘积．若 $T_q = T_k = 8192$，单头分数矩阵含 67108864 个元素；即使每元素仅 2 字节，也占 128 MiB，32 头则仅这一类矩阵已达 4 GiB，尚未计批量和其他状态.

FlashAttention 将查询块、键值块及归一化状态安排在片上存储中，利用分块计算减少中间矩阵与高带宽显存之间的往返 [143]．核心收益在于数据移动，不是将稠密注意力的 $O(T_q T_k d_h)$ 算术阶改成线性．因果掩码可以跳过某些完全不可见块，但仍保留对应可见集合内的完整注意力.

33.3.1　块大小、并行度及反向重算

较大的块可以复用更多数据，却占用更多寄存器和片上存储，降低同时驻留的执行单元数；过小的块则增加读取和循环开销．因此块大小由头维、精度、设备资源及并行映射共同决定，不能只追求最大块.

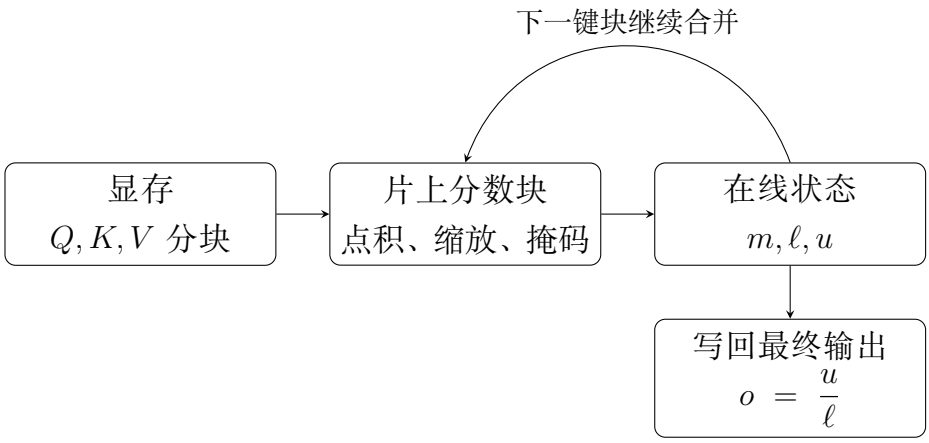


图 33.1: 分块注意力的数据移动．完整分数矩阵不必写回显存，但查询仍处理全部合法键.

训练时无需保存全部注意力概率，也可以在反向中由保存的行归一化信息和重算分数恢复局部概率．令 $P = \mathrm{softmax}(S)$、$O = PV$，上游梯度为 G_O，则

$$G_V = P^{\mathsf{T}} G_O, \qquad G_P = G_O V^{\mathsf{T}}, \tag{33.6}$$

$$(G_S)_{ij} = P_{ij}\left[(G_P)_{ij} - \sum_k P_{ik}(G_P)_{ik}\right], \tag{33.7}$$

$$G_Q = \frac{G_S K}{\sqrt{d_h}}, \qquad G_K = \frac{G_S^{\mathsf{T}} Q}{\sqrt{d_h}}. \tag{33.8}$$

这些关系解释为什么反向可逐块重新计算所需概率. 实际实现还须处理掩码、Dropout 及浮点累计；不能在重算时更换随机掩码.

33.3.2　接口、后端及正确性条件

缩放点积注意力接口规定输入输出语义，具体后端由设备、形状、精度和支持条件选择．调用统一接口不等于确认使用了某个指定内核．采用优化路径时，掩码极性、非方形因果对齐、头分组、缩放因子和张量步幅均须保持一致.

例如缓存解码的查询只覆盖最新位置，而键包含整个历史．如果将一个 $1 \times T_k$ 因果掩码误按查询局部索引从零对齐，可能只允许读取第一个键．正确可见集合取决于查询的逻辑位置，不能只由矩阵左上角推断．此类错误输出可能仍有限，因此“没有 NaN”不是正确性证据.

33.4　算子融合及执行图

算子融合（Operator Fusion）把相邻运算放在同一执行内核中，减少中间张量写回和启动．以 $y = \phi(xW + b)$ 为例，若矩阵乘法后写回 $Z = xW$，偏置与激活再分别

读写，额外流量与 Z 的元素数成正比．将偏置、激活作为矩阵乘法的尾部计算，可避免这些完整中间结果的显存往返．

融合也有边界：中间结果被多个分支使用时，重新计算可能比保存更贵；融合后寄存器压力过大可能使数据溢出到较慢存储；跨设备通信或跨线程全局归约常需要显式同步．数学表达能合并，不保证单个大内核一定更快．

图编译（Graph Compilation）将一段运算图转为特化执行程序；**执行图回放**（Execution Graph Replay）复用已记录的提交依赖，主要减少重复调度．两者并非同义：回放不必重新选择算子算法，编译也不保证所有动态控制流都能被捕获．

当主机读取设备张量的值以决定分支时，可能引入同步并切断可捕获区域．应把数据依赖与元数据依赖分开分析．不能为了避免图断裂而把用户要求的停止条件或可见性逻辑删去．

33.4.1　形状特化的摊销模型

设一次编译成本为 C，未编译每次耗时 t_e，编译后耗时 $t_c < t_e$，调用 n 次，则净收益条件为

$$C + nt_c < nt_e \quad \Longleftrightarrow \quad n > \frac{C}{t_e - t_c}. \tag{33.9}$$

若 $C = 3$ 秒，每次由 2 毫秒降到 1.5 毫秒，需要超过 6000 次调用才能取得净时间收益．对每个形状分别编译时，应逐桶计算 C_i, n_i，冷门桶可能始终无法摊平．

形状分桶（Shape Bucketing）把实际形状映射到有限执行形状．设长度随机变量为 T，映射到桶上界 $b(T)$，填充位置的期望为 $\mathbb{E}[b(T) - T]$．对稠密注意力，额外计算更接近 $\mathbb{E}[b(T)^2 - T^2]$，不能只按平均长度差估计．桶过多增加编译和缓存，桶过少增加无效计算与预留内存．

33.5　推测解码的精确性

33.5.1　接受质量及残差质量

推测解码（Speculative Decoding）由较便宜的草稿过程提出候选，再由目标模型验证 [182]．下面讨论保持目标采样分布的精确形式，不将贪心匹配规则混入随机采样证明．

采样分布条件

p 与 q 定义在同一离散词元空间，均为归一化概率．候选确实按 q 采样，计算的概率包括各自实际使用的温度、截断和约束．目标 p 是希望保留的分布；q 不必与 p 使用相同采样参数．所有概率对应同一个已确认前缀．

对候选 $x \sim q$，当 $q(x) > 0$ 时以

$$a(x) = \min\{1, \frac{p(x)}{q(x)}\} \tag{33.10}$$

接受．$q(x) = 0$ 的词元不会被提议，无需对其计算该比值．接受并输出 x 的无条件质量为 $q(x)a(x) = \min(p(x), q(x))$．总接受率为

$$A = \sum_x \min(p(x), q(x)) = 1 - \frac{1}{2}\sum_x |p(x) - q(x)|. \tag{33.11}$$

第二个等号由 $\min(a, b) = \dfrac{a + b - |a - b|}{2}$ 得到．因此接受率等于 1 减去两个分布的总变差距离．

若拒绝，令 $Z = 1 - A$，从残差分布

$$r(x) = \frac{[p(x) - q(x)]_+}{Z} \tag{33.12}$$

采样，其中 $[u]_+ = \max(u, 0)$．由于两分布和均为 1，正差之和恰为 Z，残差归一化成立．最终输出概率为

$$\Pr(\text{output} = x) = \min(p(x), q(x)) + Zr(x) = p(x). \tag{33.13}$$

若 $Z = 0$，拒绝事件概率为零，不应执行除以零的残差构造．目标有质量而草稿为零的位置，可通过拒绝后的残差获得，因此草稿不必覆盖目标的完整支持集．

例 25.2

设 $p = (0.5, 0.3, 0.2)$、$q = (0.2, 0.5, 0.3)$．接受概率为 $(1, 0.6, \frac{2}{3})$，接受路径质量为 $(0.2, 0.3, 0.2)$，总接受率 0.7．拒绝概率 0.3，正残差为 $(0.3, 0, 0)$，因此拒绝后必定输出第一个词元．合并得到 $(0.5, 0.3, 0.2)$，正好是目标分布．

若错误地在拒绝后直接从 p 重采样，最终质量为 $(0.2, 0.3, 0.2) + 0.3p = (0.35, 0.39, 0.26)$，已改变目标．较高接受率不能补救错误的拒绝分支．

33.6　多词元验证及缓存事务

草稿依次提出 k 个候选 $x_1, \ldots, x_k$．目标模型利用因果前向，一次计算这些候选对应前缀下的条件分布，以及全部接受之后的下一分布．验证必须按候选顺序进行：仅在前面候选全部接受时，当前目标条件才对应已确认前缀．

第一次拒绝发生在 j 时，保留 $x_{<j}$，按该位置残差采样修正词元，并丢弃 $x_{>j}$．如果全部接受，则可从目标在完整草稿前缀后的分布再采样一个词元．逐位置应用式 (33.13) 并对已确认前缀归纳，可得最终序列仍服从目标自回归分布．

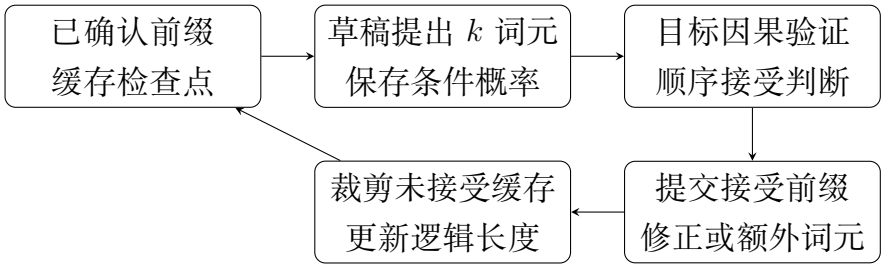


图 33.2: 推测解码的提交边界．草稿产生的状态是暂存状态，拒绝后的后缀不能继续参与注意力．

生成的修正词元尚未必被目标模型前向处理. 因此实现需区分“已输出序列长度”和“已物化 KV 的长度”，下一次调用从未物化的后缀接续．草稿模型也要恢复到相同已确认前缀．简单删除输出文本却保留被拒绝词元的 KV，会破坏下一步条件概率.

终止词元一旦接受或由修正分支生成，应立即结束请求，不再附加奖励词元．长度预算也应裁剪本轮可提交量．随机数流不同可以导致同一种子生成不同具体样本；分布保持不等于逐样本输出相同．[1]

算法 33.2　保持目标分布的推测解码

输入：目标 p、草稿 q、已确认前缀、候选预算 k、停止规则；**输出：**目标分布下的新词元.

1. 保存草稿与目标的已确认缓存边界．草稿自回归采样不超过 k 个候选，并保存或能够精确重建每步完整的条件提议分布 $q_i(\cdot)$．仅保存抽中词元的概率不足以构造拒绝后的残差分布；等价实现必须提供精确残差采样所需的分布表示.
2. 目标因果验证候选，取得每个候选前缀下的实际目标概率及全部接受后的分布.
3. 按顺序比较独立均匀随机数与接受概率；接受则提交该候选，遇停止标记立即结束.
4. 第一次拒绝时，按式 (33.12) 采样修正词元，舍弃后续候选及其缓存，结束本轮.
5. 若候选全部接受且尚未达到停止或长度条件，从额外目标分布采样一个词元.
6. 同步两套逻辑前缀，标记尚未物化的 KV 后缀；继续下一轮或返回完成结果.

[1]确定性贪心解码可采用与目标最大概率词元逐项比较的专门规则，但那是在复现贪心路径，不是证明随机目标分布保持.

33.7　推测收益及正确性边界

设一轮中每个候选在之前均已接受的条件下，接受概率都为常数 α，暂不考虑提前终止．至少提交 $i+1$ 个词元需要前 i 个候选全部接受，因此一轮提交量 N 的期望为

$$\mathbb{E}[N]=\sum_{i=0}^{k}\Pr(N\geqslant i+1)=\sum_{i=0}^{k}\alpha^{i}=\frac{1-\alpha^{k+1}}{1-\alpha},\tag{33.14}$$

$\alpha=1$ 时取极限 $k+1$．条件接受率不恒定时，应使用各级生存概率，不能把所有候选的总体平均接受率直接代入幂次．

设草稿每步代价 c_d，目标验证一轮代价 $c_v(k)$，缓存和采样管理代价 c_m，普通目标单步代价 c_t．近似加速比为

$$S\approx\frac{c_t\mathbb{E}[N]}{kc_d+c_v(k)+c_m}.\tag{33.15}$$

取 $k=3,\alpha=0.8$，期望提交量为 2.952．若 $c_t=10$ 毫秒、$c_d=1$ 毫秒、$c_v=12$ 毫秒、$c_m=1$ 毫秒，则近似加速比为 $\dfrac{29.52}{16}=1.845$．若验证代价增到 30 毫秒，则降到 $\dfrac{29.52}{34}<1$．这些是给定成本的构造算例，不是任何设备的实测．

滑动窗口注意力（Sliding-Window Attention）只保留规定范围内的可见键．对原本全注意力的模型，删除窗口外质量通常改变输出，即使分块计算本身完全精确．若被删除的原注意力概率质量为 δ，所有值向量范数不超过 M，保留部分重新归一化后输出 o'，则当 $\delta<1$ 时

$$\|o-o'\|\leqslant 2\delta M.\tag{33.16}$$

证明是将原输出写成 $(1-\delta)o'+\delta o_{\mathrm{drop}}$，再用三角不等式．该上界需要知道被删除的质量，不能由“距离较远”自动推断 δ 很小．原生窗口模型按照其既定掩码执行，则是在保留该模型自身的目标．

最终正确性需要区分算子数值容差、缓存逻辑一致、采样分布与任务质量．随机解码不能只比较一次文本是否相同；应结合分布证明与小词表概率算例．性能记录则区分首次编译、稳态执行、首词元、后续词元以及排队边界，避免把局部内核收益误认为端到端收益．

第三十四章　模型优化

模型优化围绕质量、容量和执行成本组织四类方法：量化调整数值表示，剪枝选择保留的结构，低秩分解用矩阵因子形成近似，蒸馏把教师行为转移给学生．每种方法都需要说明近似对象、误差目标、训练或校准条件，以及目标硬件上的执行方式．

本章先建立量化映射与低精度计算的数值基础，再推导剪枝、低秩分解和蒸馏，最后讨论组合优化与同条件比较．知识蒸馏只有在学生结构或表示更省资源时才产生压缩收益；参数高效微调中的 LoRA 学习低秩更新，低秩压缩则以因子替换原矩阵．

34.1　压缩对象及误差预算

局部近似的边界

分析线性层时采用列向量约定 $y = Wx + b$，其中 $W \in \mathbb{R}^{m\times n}$．偏置若未说明则保持不变．剪枝敏感度以可微损失的局部展开为依据，二阶结论不自动适用于大幅参数变动．低秩最优性针对明确的矩阵范数或校准输入分布．蒸馏教师在学生更新期间冻结，师生输出类别及监督位置已对齐．

符号	含义与形状
$W, \widehat{W}, E$	原矩阵、压缩后矩阵及误差 $E = \widehat{W} - W$，形状 $m \times n$.
x, X, C_x	输入 $x \in \mathbb{R}^n$；校准列矩阵 $X \in \mathbb{R}^{n\times N}$；二阶矩 $C_x = \mathbb{E}[xx^{\mathsf{T}}]$.
M, k, s	二值剪枝掩码、保留权重数量、单权重存储字节数.
w, g, H	展平参数向量、损失梯度和 Hessian，形状分别为 P、P、$P \times P$.
r, σ_i	近似秩与降序排列的奇异值.
A, B	低秩因子，$A \in \mathbb{R}^{m\times r}$，$B \in \mathbb{R}^{r\times n}$.
z_T, z_S, C, τ	教师和学生的 C 维 logits、类别数与正温度.
$p_T^\tau, p_S^\tau, \lambda$	温度分布与硬软目标混合系数，$0 \leqslant \lambda \leqslant 1$.
H_T, H_S, P_h	展平有效位置后的师生表示与对齐投影，形状为 $N_t \times d_T$、$N_t \times d_S$、$d_S \times d_T$.

模型压缩（Model Compression）可以减少权重容量、计算次数、数据搬运或特定结构规模．这些目标不等价．**知识蒸馏**（Knowledge Distillation，KD）首先是一种监督转移方法，只有学生的实际结构或表示更省资源时才形成压缩．混合专家架构的稀疏激活也不等于压缩，因为它可能同时增大总参数量．

比较方案时，应固定原模型、任务、输入长度分布、批量、输出质量要求与目标设备．局部误差、任务指标和系统指标分别测量：矩阵 Frobenius 误差描述数值近似，任务正确率描述行为，峰值内存和延迟描述执行成本．不能让其中一个指标替代其他两类．

34.2　量化

34.2.1　量化对象及基本符号

位宽对象

仅权重量化（Weight-Only Quantization）降低权重存储位宽，而输入激活可以保持浮点．记号 W4A16 通常表示四比特权重与十六比特激活，但仍未说明码本、分组、乘法指令、累加精度及输出类型．W8A8 进一步降低激活位宽，**键值缓存量化**（Key-Value Cache Quantization）则压缩解码历史中的键与值．这三种对象具有不同生命周期和误差传播方式．

权重通常可以离线统计并一次编码；激活随请求和位置变化；缓存随生成增长并被反复读取．只压缩权重不会按同一比例降低长上下文缓存，激活位宽也不决定累加器位宽．描述一个低精度模型时，应分别给出存储、乘法输入、累加和输出的表示．

表 34.1: 主要符号．矩阵乘法采用列向量输入约定．

符号	含义
$x, q, \widehat{x}$	原始实数、整数码值、恢复实数
s, z	正尺度与整数零点
$q_{\min}, q_{\max}$	有限整数编码范围
ℓ, u	实际可恢复范围的下端与上端
W, X, Y	权重 $d_o \times d_i$、输入 $d_i \times n$、输出 $d_o \times n$
E, C, H	权重误差、输入二阶矩、局部重建 Hessian
D	正对角通道缩放矩阵，$d_i \times d_i$
g, b	每组元素数与码值位宽

量化编码及恢复

仿射量化（Affine Quantization）采用

$$q = \operatorname{clip}\Big(\operatorname{round}(\frac{x}{s}) + z, q_{\min}, q_{\max}\Big), \qquad \widehat{x} = s(q - z), \quad s > 0. \tag{34.1}$$

其中**量化尺度**（Quantization Scale）s 决定相邻可恢复值的间距，**零点**（Zero Point）z 是实数零对应的整数码．要求 z 是编码范围内的整数，就可以精确恢复零；这对于

填充、稀疏值和某些算子的零语义很重要. **反量化**（Dequantization）返回一个近似实数，而不是恢复原始数值. 整数推理的此类表示与算术关系见 Jacob 等人的研究[183].

对希望覆盖的区间 $[a,b]$，先扩展为 $a'=\min(a,0)$、$b'=\max(b,0)$. 若 $a'<b'$，一种常见初始化是

$$s=\frac{b'-a'}{q_{\max}-q_{\min}},\qquad z=\operatorname{clip}\left(\operatorname{round}(q_{\min}-\frac{a'}{s}),q_{\min},q_{\max}\right). \tag{34.2}$$

由于 z 被取整，真正的恢复端点是 $\ell=s(q_{\min}-z)$、$u=s(q_{\max}-z)$，不一定恰好为 a',b'. 端点调整和舍入规则应共同写入协议，不能一边用理想区间证明误差，一边用另一组实际端点执行.

全零张量需要显式约定，例如取任意固定正 s、合法 z，全部编码为 z. 不应令 $s=0$ 后继续计算 $\frac{x}{s}$. 非零常量在扩展区间包含零后不再是零宽区间；若采用其他常量特例，也必须保证编码与恢复一致.

对称量化及码值范围

对称量化（Symmetric Quantization）在有符号编码下令 $z=0$. 例如选择有效码值范围 $[-127,127]$，令 $c=\max|x|>0$，可取 $s=\frac{c}{127}$. 八比特有符号存储本身还可表示 -128，但该对称方案有意不使用它. 若使用 $[-128,127]$，可恢复范围不再关于零完全对称，尺度约定也应重新说明. [1]

最近舍入（Round to Nearest，RTN）在等距离点需要平局规则，例如取偶数. 向零截断会产生不同偏差，不能套用最近舍入的所有结论. 除非特别说明，本章采用最近舍入，数值例题避免依赖未声明的平局选择.

34.2.2 舍入、截断及饱和误差

> **均匀量化的误差边界**
>
> $s>0$、$z\in\mathbb{Z}$ 固定，码值是连续整数区间，最近舍入后执行范围截断. 计算暂视为精确实数运算，忽略尺度本身的浮点舍入.

两类误差的分解

当 $x\in[\ell,u]$ 时，其最近格点距离至多为半个步长，因此

$$|\widehat{x}-x|\leqslant\frac{s}{2.} \tag{34.3}$$

[1] “INT8”只描述存储类型，不能唯一确定使用多少有效码字、如何处理零点和采用何种舍入. 不同制品即使都保存为八位整数，也可能不兼容.

令 $x_c = \mathrm{clip}(x, \ell, u)$，则总误差可分解为

$$\widehat{x} - x = (\widehat{x} - x_c) + (x_c - x). \tag{34.4}$$

第一项是范围内格点误差，第二项是**饱和误差**（Saturation Error）. 若 $x > u$，恢复值固定为 u，误差绝对值为 $x - u$；若 $x < \ell$，则为 $\ell - x$. 超出范围的误差不受 $\frac{s}{2}$ 全局约束. **裁剪**（Clipping）缩小范围会减小多数数值的步长，也会增加尾部饱和损失.

在误差均匀分布于 $[-\frac{s}{2}, \frac{s}{2}]$ 的理想模型下，均方误差为

$$\mathbb{E}[e^2] = \frac{1}{s}\int_{-\frac{s}{2}}^{\frac{s}{2}} e^2 \,\mathrm{d}e = \frac{s^2}{12}. \tag{34.5}$$

但真实权重可能聚集于格点附近，激活可能含大量零，误差也可能与输入相关. 这一噪声模型适合估计量级，不能当作所有张量的实测误差公式.

给定连续密度 $p(x)$，实际均方误差可写为

$$\mathbb{E}[(\widehat{X} - X)^2] = \int_{-\infty}^{\ell} (x - \ell)^2 p(x)\,\mathrm{d}x + \int_{u}^{\infty} (x - u)^2 p(x)\,\mathrm{d}x \tag{34.6}$$

$$+ \int_{\ell}^{u} (Q_s(x) - x)^2 p(x)\,\mathrm{d}x, \tag{34.7}$$

其中 Q_s 为范围内最近格点映射. 前两项是尾部损失，最后一项是舍入损失. 这给出了选择裁剪范围的统计目标，而不是一味追求无溢出或最小尺度.

例 26.1

取三比特无符号码 $q \in \{0, \dots, 7\}$，$s = 0.5, z = 2$，实际恢复范围为 $[-1, 2.5]$. 对输入 $(-1.2, -0.6, 0, 0.7, 2.8)$，编码、恢复与误差见表 34.2. 均方误差为

表 34.2: 给定尺度与零点的手工编码结果，误差定义为 $\widehat{x} - x$.

x	$\mathrm{round}(\frac{x}{s}) + z$	q	$\widehat{x}$	误差
-1.2	0	0	-1	0.2
-0.6	1	1	-0.5	0.1
0	2	2	0	0
0.7	3	3	0.5	-0.2
2.8	8	7	2.5	-0.3

$$\frac{0.04 + 0.01 + 0 + 0.04 + 0.09}{5} = 0.036.$$

最后一个值的误差超过 $\frac{s}{2} = 0.25$，因为发生饱和. 第一个值也在恢复范围之外，但舍入后码值尚未越界；因此检测饱和应依据实数可恢复范围与误差语义，不能仅计数最终整数裁剪操作是否改变码值.

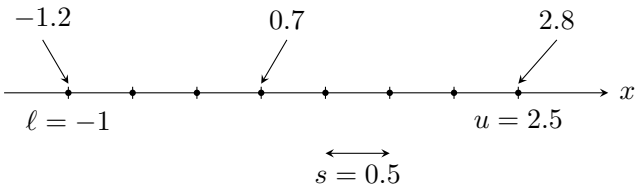


图 34.1: 范围内映射到最近格点，范围外映射到端点．端点外的误差随输入距离增长.

34.2.3 粒度及校准分布

尺度共享范围

逐张量量化（Per-Tensor Quantization）为整个张量使用同一尺度．**逐通道量化**（Per-Channel Quantization）允许每个通道独立选择尺度．对 $W \in \mathbb{R}^{d_o \times d_i}$，若每个输出通道对应一行，则逐输出通道尺度为 s_i；改变矩阵存储方向时，不能仅按数组的“第一维”推定通道含义.

分组量化（Group-Wise Quantization）在每个明确分组内共享尺度和必要零点．线性层常按每行的输入方向分组，大小为 g；最后一组可能不足 g，仍需元数据及布局约定．粒度越细，越容易适应局部动态范围，但尺度数量、读取成本和内核约束也随之增加.

例如

$$W = \begin{bmatrix} 0.1 & 0.2 & -0.1 \\ 10 & 20 & -10 \end{bmatrix}. \tag{34.8}$$

使用有效三比特对称码 $[-3, 3]$，逐张量尺度 $\dfrac{20}{3}$ 使第一行全部恢复为零．第二行的 10 恰在平局点，若采用取偶数则恢复为 $\dfrac{40}{3}$；这个结果依赖明确舍入规则．逐行尺度分别为 $\dfrac{0.2}{3}$ 和 $\dfrac{20}{3}$，第一行恢复为 $(\dfrac{2}{15}, 0.2, -\dfrac{2}{15})$，其平方误差均值为 $\dfrac{2(\frac{1}{30})^2}{3} = \dfrac{1}{1350}$，而逐张量第一行误差均值为 0.02．第二行保持原量化结果．这说明异尺度通道共享步长可能损害小值通道，不证明逐通道量化总能提高任务质量.

若原始 P 个元素每个 16 位，量化码每个 b 位，每组有 h 字节元数据且忽略填充，则总存储近似为

$$B_q = P\frac{b}{8} + \left\lceil \frac{P}{g} \right\rceil h. \tag{34.9}$$

四比特码、$g = 64, h = 4$ 时，每元素约 0.5625 字节，相对两字节基线压缩约 3.56 倍，并非恰好四倍．真实矩阵逐行分组时，应逐行求组数；对齐、索引和未量化层还会增加成本.

校准分布估计

量化校准（Quantization Calibration）用代表性输入估计激活范围、误差或通道重要性. 最小最大值策略尽量覆盖观测极值，百分位策略允许一定尾部裁剪，直方图或重建误差策略显式权衡式 (34.7). 这些策略使用不同目标，不存在脱离数据分布的统一最优范围.

校准应覆盖预期语言、领域、模板、长度和生成阶段. 只使用短输入估计长上下文缓存范围，或只观察预填充却假设逐词解码完全相同，可能遗漏实际分布. 最终测试数据应与范围、位宽和算法选择隔离，避免把评估集变成调参集.

静态量化（Static Quantization）在部署前固定激活尺度；**动态量化**（Dynamic Quantization）在运行时按张量、词元或组更新尺度. 动态策略减少对固定范围的依赖，却需要归约、缩放与元数据处理. 它也不能保证不存在异常值；只要一个组内极端值与大量普通值共存，动态最大值尺度仍会造成精度损失.

输出误差传播

令 $E = \widehat{W} - W$，则单输入误差为 Ex，满足

$$\|Ex\|_2 \leqslant \|E\|_2 \|x\|_2. \tag{34.10}$$

若输入二阶矩 $C = \mathbb{E}[xx^{\mathsf{T}}]$ 存在，则

$$\mathbb{E}\|Ex\|_2^2 = \operatorname{tr}(ECE^{\mathsf{T}}). \tag{34.11}$$

因此同样的权重均方误差可能产生不同输出误差：高频或高幅值输入方向赋予对应列更大权重. 下游非线性和残差还会重新传播误差，局部重建并不等同于端到端任务最优. 这个区别构成激活感知方法与二阶补偿方法的共同出发点.

34.2.4　GPTQ 的二阶误差补偿

局部重建目标

生成式模型训练后量化（GPTQ）利用层输入构建局部重建目标，并以二阶信息补偿量化误差 [184]. 固定校准输入 $X \in \mathbb{R}^{d_i \times n}$，希望在离散码本约束下减小 $\|(W - \widehat{W})X\|_F^2$. 每一输出行可以分别讨论.

把某行权重转为列向量 $w \in \mathbb{R}^{d_i}$，令更新为 δ. 重建损失增量为

$$\mathcal{E}(\delta) = \|\delta^{\mathsf{T}} X\|_2^2 = \tfrac{1}{2}\delta^{\mathsf{T}} H \delta, \qquad H = 2XX^{\mathsf{T}}. \tag{34.12}$$

这是固定线性层重建的精确二次形式，不是整个语言模型任务损失的精确 Hessian.

单坐标量化补偿

对当前尚可调整的坐标集合，H 正定且码本已固定．将坐标 i 固定为恢复值 $\widehat{w}_i$，其余坐标暂可连续调整．若校准矩阵秩不足，可加入正阻尼 λI，此时优化的已是正则化重建目标．

约束二次问题的解

记 $d=\widehat{w}_i-w_i$，e_i 为第 i 个标准基向量．求解

$$\min_{\delta}\ \tfrac{1}{2}\delta^{\mathsf{T}}H\delta,\qquad e_i^{\mathsf{T}}\delta=d. \tag{34.13}$$

拉格朗日函数为 $\frac{1}{2}\delta^{\mathsf{T}}H\delta+\mu(e_i^{\mathsf{T}}\delta-d)$. 令梯度为零, 有 $H\delta+\mu e_i=0$, 所以 $\delta=-\mu H^{-1}e_i$. 代回约束得到

$$\delta^{*}=\frac{d}{(H^{-1})_{ii}}H^{-1}e_i,\qquad \mathcal{E}(\delta^{*})=\frac{d^2}{2(H^{-1})_{ii}}. \tag{34.14}$$

若用量化残差 $w_i-\widehat{w}_i$ 表示，则更新前有负号．算法中的符号差异常来自残差方向，不应机械照抄．非对角项决定其他坐标如何补偿；若输入不相关且 H 对角，则没有跨坐标补偿．

固定该坐标后，剩余自由集合的逆 Hessian 可由当前逆矩阵 $K=H^{-1}$ 消去相应行列：

$$K_{\text{new}}=K_{-i,-i}-\frac{K_{-i,i}K_{i,-i}}{K_{ii}}. \tag{34.15}$$

逐坐标直接执行代价较高．GPTQ 采用一致的处理顺序、分块延迟更新及 Cholesky 形式组织这些计算，以提高大矩阵处理效率 [184]．这些组织方式不能把逐层贪心的离散选择变成全局最优保证.

例 26.2

设

$$H=\begin{bmatrix}2&1\\1&2\end{bmatrix},\qquad H^{-1}=\frac{1}{3}\begin{bmatrix}2&-1\\-1&2\end{bmatrix},\qquad w=\begin{bmatrix}0.3\\0.6\end{bmatrix}. \tag{34.16}$$

采用间隔 0.5 的格点，先把第一坐标量化为 0.5，$d=0.2$．式 (34.14) 给出 $\delta=(0.2,-0.1)^{\mathsf{T}}$，故第二坐标变为 0.5，恰好已在格点上．补偿误差为 $\frac{1}{2}\delta^{\mathsf{T}}H\delta=0.03$；若只改变第一坐标，误差为 0.04．该例显示相关输入允许利用另一坐标抵消输出扰动，不意味着每一步补偿后都无需再量化其他坐标.

算法 34.1　二阶补偿量化的概念流程

输入：层权重、校准输入、量化格点、坐标顺序与阻尼．**输出：**量化码值及元数据．**状态：**当前连续权重、自由坐标集合、曲率分解.

1. 根据 X 构建重建曲率，处理不可观测方向并加入所需阻尼，得到稳定分解.
2. 按固定顺序选择自由坐标，对其当前补偿后的值作格点选择.

3. 根据式 (34.14) 将残差传播到其余自由坐标，已固定坐标不再改变；更新自由子问题.

4. 所有坐标处理后停止，保存格点、分组、处理顺序及布局. 实际高效实现使用分块与分解复用，避免重复显式求逆.

不变量：已固定权重始终位于指定格点；校准曲率只约束该重建问题，不能替代独立任务评价.

34.2.5　AWQ 及 SmoothQuant 的通道缩放

通道缩放恒等式

对任意正对角矩阵 $D = \mathrm{diag}(d_1, \ldots, d_{d_i})$，有

$$WX = (WD)(D^{-1}X). \tag{34.17}$$

第 j 列权重乘以 d_j，对应输入通道除以 d_j，未量化的函数不变. 但量化不是线性算子，因此对右侧两部分量化后的误差一般不同于直接量化左侧. 通道缩放的作用是重新分配表示难度，不能凭恒等式声称量化后仍完全等价.

激活感知权重量化（Activation-Aware Weight Quantization，AWQ）重点关注仅权重量化中的显著输入通道 [185]. 用 $\mathcal{Q}$ 表示量化后恢复的权重，其校准目标可以写为

$$\min_{D} \ \|\mathcal{Q}(WD)D^{-1}X - WX\|_F^2. \tag{34.18}$$

若某一权重的组步长从 Δ 变为 Δ'，缩放后换回原坐标的最近格点误差界为 $\dfrac{\Delta'}{2d_j}$. 只有当 $\dfrac{\Delta'}{d_j}$ 相对原步长更小，相关方向才获得更细分辨率. 过大缩放可能提高整个组的最大值，损害其他通道；不能把所有通道无限放大.

AWQ 使用激活统计构造缩放候选并进行有限搜索，而不是根据权重幅值单独决定重要性. 这里“保护显著通道”不意味着最终必须把这些通道保留为高精度；可通过尺度变换仍然输出统一低比特权重. 候选范围、裁剪和校准数据影响结果，也不能从方法名称推定其不会对校准分布过拟合.

激活离群值迁移

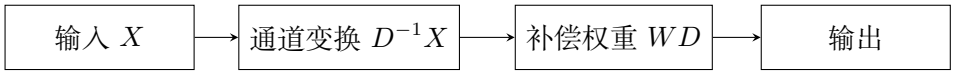
平滑量化（SmoothQuant）利用同一恒等式缓解激活离群值，使权重与激活都更适合低精度计算 [186]. 令 $a_j = \max\limits_{t} |X_{jt}|$、$b_j = \max\limits_{i} |W_{ij}|$，对非零通道可取

$$d_j = \frac{a_j^{\alpha}}{b_j^{1-\alpha}}, \qquad 0 \leqslant \alpha \leqslant 1. \tag{34.19}$$

变换后的输入最大幅值为 $\frac{a_j}{d_j} = (a_j b_j)^{1-\alpha}$，权重列最大幅值为 $b_j d_j = (a_j b_j)^{\alpha}$. 在 $\alpha = \frac{1}{2}$ 时，两者在该通道达到相同最大幅值. 零通道、极小值以及尺度上下界需要显式处理，避免除零或异常放大.

例如 $a = (100, 1)$、$b = (0.01, 1)$，$\alpha = \frac{1}{2}$ 时 $d = (100, 1)$，两个变换后张量的对应通道最大幅值均为 $(1, 1)$. 这个构造例子展示数值范围如何迁移，不说明任意网络都能如此平衡. 选择 α 仍需考虑两边实际量化粒度与误差.

AWQ 以权重量化误差及激活重要性为中心，SmoothQuant 则平衡权重和激活的量化难度. 它们可共用代数视角，但目标与校准统计不同. 缩放若吸收到前一归一化或线性层，必须保持所有分支一致；同一输入被多个投影共享时，不能只改其中一个消费者. 跨越非线性随意移动缩放通常不保持函数不变.



量化前：函数恒等
量化后：误差取决于两侧的格点、粒度和尺度

图 34.2: 通道尺度变换保持原函数，却改变低精度表示的误差分配.

34.2.6 PTQ、QAT 及低精度训练

训练后量化（Post-Training Quantization，PTQ）从已训练权重出发，利用统计校准、搜索或局部重建产生量化模型，通常不进行完整任务训练. 简单最大值权重量化可以不用激活数据，GPTQ、AWQ 和 SmoothQuant 则使用输入统计；所以 “PTQ 不需要数据” 并非一般定义.

量化感知训练（Quantization-Aware Training，QAT）在训练前向中加入**伪量化**（Fake Quantization），即 $\widehat{x} = D(Q(x))$，让优化过程适应部署格点 [183]. 浮点主参数可以保留，前向使用恢复后的近似值. 因最近舍入几乎处处导数为零、跳变处不可导，常使用**直通估计器**（Straight-Through Estimator，STE），例如把反向近似为

$$\frac{\partial \widehat{x}}{\partial x} \approx \mathbf{1}_{\ell \leqslant x \leqslant u}. \tag{34.20}$$

这是人为选择的优化代理，不是真实量化函数的导数；尺度与裁剪参数的学习也需要相应规则. 前向模拟的舍入、饱和和粒度若与部署不一致，训练恢复的质量可能无法转移到实际内核.

QAT 不要求训练器本身用目标整数指令完成全部运算. **低精度训练**（Low-Precision Training）则进一步让某些前向、反向矩阵乘法或状态采用低精度执行，需要管理累加、主权重和更新尺度. QLoRA 的冻结量化基座加可训练适配器，是另一

种参数与计算边界；它不等同于对全部量化码值做梯度更新，详见第二十一章《参数高效微调》.

34.2.7　FP8 及非均匀表示

浮点格点随指数变化

八位浮点（8-bit Floating Point，FP8）不是一种均匀整数网格. 对普通规格化二进制浮点数，可以写成 $(-1)^s 2^{e-\beta}(1+\frac{f}{2^m})$，其中 s 为符号位、e 为指数域、β 为偏置、m 为尾数字段位数. 一个指数区间内间隔为 $2^{e-\beta-m}$，随数量级变化.

在远离溢出和下溢、采用最近舍入的规格化区域，单位舍入误差量级为 $2^{-(m+1)}$，描述的是相对误差尺度；不能套用单一全张量绝对误差界. 次正规数、零附近以及饱和区域需要分别分析.

Micikevicius 等人提出的 FP8 编码包括 E4M3 与 E5M2[187]. 二者均有一个符号位，分别分配四位指数三位尾数、五位指数两位尾数. 该论文中 E4M3 最大有限值为 448、最小正规数为 2^{-6}；E5M2 最大有限值为 57344、最小正规数为 2^{-14}. E4M3 通过减少特殊值编码扩展范围，不表示无穷大；E5M2 使用不同的特殊值规则. 其他编码变体不能仅凭名称共享这些端点.

E4M3 在相似数值区间拥有更细尾数，E5M2 提供更大动态范围. 具体训练不应机械地把全部张量转换为同一格式；梯度、权重、激活与归约具有不同数值需求. [2]

缩放、累加及更新

FP8 张量仍常需要外部尺度. 例如令 $\widehat{x} = s\ \mathrm{cast}_{\mathrm{FP8}}(\frac{x}{s})$，尺度把当前范围映射到有限编码区间. 由当前张量估计尺度需要额外归约；用历史最大值延迟更新尺度会引入状态，并可能来不及覆盖分布突变. 静态、动态和分块尺度必须绑定实际执行协议.

低精度乘法不意味着低精度累加. 长内积可能在低位累加中丢失小贡献或溢出，因而通常需要更宽累加器、分段归约或其他受支持的数值策略. 若优化器更新量小于权重所在格点间距，直接在粗格点中更新可能完全消失；高精度主参数或合适舍入方式可以缓解，但也增加状态成本. 是否支持某种 FP8 乘法、累加和布局，由具体硬件与内核决定，不能由文件类型推出.

非均匀四比特码本 NF4 则面向另一种分布假设，第二十一章《参数高效微调》已经说明其在 QLoRA 中的职责. 对非均匀码本，局部最近舍入误差由相邻码字间隔控制，不能把统一整数步长 $\frac{s}{2}$ 直接套到所有区间.

[2] “尾数位”在此指不包含规格化隐含前导一的尾数字段. 将隐含位计入有效精度后再比较，才不会把 E4M3 误认为只有三位总有效数字.

34.2.8 整数计算及反量化融合

整数点积中的零点修正

设一个输出为 $y=\sum_{j=1}^{d_i} w_j x_j$，权重与激活使用尺度 s_w, s_x 和零点 z_w, z_x. 则恢复后的点积为

$$\widehat{y} = s_w s_x \sum_j (q_{w,j}-z_w)(q_{x,j}-z_x) \tag{34.21}$$

$$= s_w s_x \left[\sum_j q_{w,j}q_{x,j} - z_x\sum_j q_{w,j} - z_w\sum_j q_{x,j} + d_i z_w z_x\right]. \tag{34.22}$$

对称权重 $z_w=0$ 可消除其中部分修正，但激活零点仍可能存在. 偏置可在兼容尺度下加入累加器，输出若还要保持整数表示，则需要**再量化**（Requantization）：按输出尺度重新舍入并裁剪.

例如 $q_w=(2,-1)$、$z_w=0, s_w=0.5$，$q_x=(5,1)$、$z_x=2, s_x=0.25$，恢复值为 $w=(1,-0.5)$、$x=(0.75,-0.25)$. 整数修正和为 $2(5-2)+(-1)(1-2)=7$，乘 0.125 得到 0.875，与恢复实数点积一致. 若省略激活零点修正，整数点积为 9，将错误得到 1.125.

累加器必须覆盖和的动态范围. 即使输入均为八比特，d_i 项乘积之和也未必适合八位或十六位累加. 分组尺度若随 j 变化，尺度不能直接提出整个求和号外，需要逐组累加和缩放，这正是粒度影响内核设计的原因之一.

三种运行路径

量化文件可以在加载时完整恢复成高精度权重，此时减少的是磁盘与传输体积，驻留权重可能没有减少. 也可以在每次前向先恢复一个完整权重张量，再调用浮点矩阵乘法；此时额外读写高精度临时数组可能抵消带宽节省. 第三种方式是**融合反量化**（Fused Dequantization）：内核读取压缩块，在寄存器或片上工作区中恢复并立即参与计算，避免完整中间张量往返显存.

融合路径仍可能使用浮点乘法，而原生低精度矩阵乘法则直接消费硬件支持的低精度操作数. 二者都可能有收益，但收益机制不同. 比特打包、尺度加载、布局转置、尾块处理和线程组织共同决定性能，算法名称本身不保证某条路径被执行.

带宽、算力及端到端收益

若某层执行 F 次浮点等价操作、传输 B 字节，计算吞吐上界为 C、带宽为 $\mathcal{B}$，一个简化下界是

$$t \geqslant \max\left(\frac{F}{C}, \frac{B}{\mathcal{B}}\right). \tag{34.23}$$

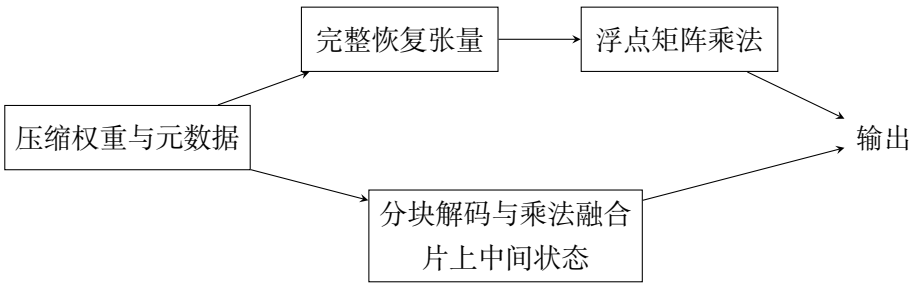


图 34.3: 完整恢复与融合恢复的数据路径不同．必须确认实际执行路径，才能解释运行时收益.

量化主要减小 B 时，带宽受限的解码可能受益；高批量计算受限场景则更依赖可用乘法指令与内核效率．反量化、归约和调度增加的成本还未包含在这个下界里.

若原系统时间有比例 f 可以加速为 a 倍，其他部分不变，则总体加速比为 $\dfrac{1}{[(1-f)+\frac{f}{a}]}$．例如只有 60% 时间受益，即使该部分快四倍，整体也仅约 1.82 倍. 词元化、调度、缓存、通信和采样都可能成为剩余瓶颈．因此文件压缩率、层内加速和请求时延改善应分别报告.

34.2.9　键值误差及发布边界

缓存误差传播

设一个查询 q 的分数为 $z_j = \dfrac{q^{\mathsf{T}} k_j}{\sqrt{d_k}}$，量化键误差为 $e_{k,j}$，则分数误差为 $\delta z_j = \dfrac{q^{\mathsf{T}} e_{k,j}}{\sqrt{d_k}}$. 一阶近似下，注意力概率变化为

$$\delta p \approx [\operatorname{diag}(p) - pp^{\mathsf{T}}]\delta z. \tag{34.24}$$

若值误差矩阵为 E_V、原输出 $o = V^{\mathsf{T}} p$，则一阶输出误差为 $\delta o \approx V^{\mathsf{T}} \delta p + E_V^{\mathsf{T}} p$，另有二阶交叉项．键误差改变权重分配，值误差改变被聚合内容，二者不能只用一个平均缓存误差完全概括.

缓存范围还涉及不同层、头、位置与新旧词元的分布．保留近期高精度尾部或分块校准属于明确设计选择，不是低位缓存的必要定义．发布评估应包含长上下文、不同生成长度和实际缓存格式；短序列困惑度不足以验证全部缓存行为.

制品及证据

算法 34.2　量化制品的形成

输入：固定模型、校准集、目标位宽与内核约束、质量预算．**输出：**量化制品及版本清单．**状态：**校准统计、候选尺度、局部与整模型误差记录.

1. 先明确目标后端可消费的布局、粒度、零点和累加约定，固定高精度参考与

输入协议.

2. 采集校准统计，依据所选方法计算尺度、通道变换或二阶补偿，保存异常值与零通道处理规则.

3. 按目标布局编码权重和元数据，使用与部署一致的恢复语义评估局部及整体质量；候选不满足预算时调整指定配置，而非隐藏回退.

4. 固定制品身份，在目标硬件上确认实际内核与负载表现. 只有质量与执行证据同时成立时，才形成可发布版本；保存原始基线以支持回滚.

不变量：编码与消费协议一致；每个性能结论绑定硬件、形状、批量、长度与精度条件. 该流程是发布要求，本章没有执行这些测量.

清单应记录模型与词表身份、量化对象、码本、尺度粒度、零点、舍入、裁剪、校准版本、布局、后端和必要通道重排. 剪枝或适配器合并改变权重后，原尺度与校准结论未必继续成立，需要形成新的制品. 对不支持的内核发生高精度回退时，应直接报告实际路径.

首词元时延（Time to First Token，TTFT）、**每输出词元时间**（Time per Output Token，TPOT）、吞吐和峰值内存分别描述不同负载特征. 测量需要预热、设备同步和固定输入分布；驻留权重、缓存与临时工作区也应分开计量. 数学误差界、给定数值算例和目标设备实测构成不同证据，不能互相替代.

低精度计算的完整契约

量化方案由可表示值、映射规则、统计分布、张量布局和执行内核共同定义. 位宽只是其中一项；同样的四比特文件可以对应不同误差和完全不同的计算路径.

34.3 剪枝

34.3.1 剪枝及敏感度

幅值剪枝的最优性范围

剪枝（Pruning）通过删除权重或结构减少模型自由度. **非结构化剪枝**（Unstructured Pruning）使用逐元素掩码 $M \in \{0,1\}^{m\times n}$，得到 $\widehat{W} = M \odot W$. 若仅要求保留 k 个原有权重，且保留的数值不允许调整，则

$$\min_{M:\sum M_{ij}=k} \|W - M \odot W\|_F^2 = \min_{M} \sum_{M_{ij}=0} W_{ij}^2. \tag{34.25}$$

因此保留绝对值最大的 k 个元素，使被删除的平方和最小. 这是**幅值剪枝**（Magnitude Pruning）在特定权重重建目标下的依据；阈值处出现相同幅值时，应固定并列选择规则以得到可复现掩码.

最小权重误差不等于最小任务误差. 若一个较小权重连接的输入幅度很大，或者该方向对输出损失极为敏感，删除它可能比删除较大权重更危险. 不同层还可以通过尺度重参数化维持近似相同函数，直接跨层比较权重幅值因此可能受到表示尺度影响.

一阶和二阶近似

设损失为 $\mathcal{L}(w)$，参数变化为 Δw. 二阶展开为

$$\mathcal{L}(w+\Delta w)-\mathcal{L}(w)\approx g^{\mathsf{T}}\Delta w+\frac{1}{2}\Delta w^{\mathsf{T}}H\Delta w, \tag{34.26}$$

其中 $g=\nabla\mathcal{L}(w)$，**海森矩阵**（Hessian Matrix）$H=\nabla^2\mathcal{L}(w)$ 描述局部二阶变化. 只删除第 q 个权重时 $\Delta w=-w_q e_q$，故

$$\Delta\mathcal{L}_q\approx -g_q w_q+\frac{1}{2}H_{qq}w_q^2. \tag{34.27}$$

一阶量反映当前位置沿删除方向的变化，绝对值 $|g_q w_q|$ 可作为影响幅度的启发式；它不保留变化方向. 接近驻点时 $g\approx 0$，若采用对角曲率近似，可用 $\dfrac{H_{qq}w_q^2}{2}$ 比较局部损失增量. 经典 Optimal Brain Damage 使用二阶敏感度研究剪枝. [188]

例如 $w=(0.1,1)$、$g=0$、$H=\mathrm{diag}(1000,1)$. 删除第一个权重的二阶代价为 $\dfrac{1000\cdot 0.1^2}{2}=5$，删除第二个为 $\dfrac{1}{2}$. 幅值排序会优先删除第一个，曲率排序却得出相反结论. 这个例子说明敏感度需要同时考虑参数幅度与局部损失几何.

多个参数一起删除时，交叉项 $H_{ij}\Delta w_i\Delta w_j$ 不再消失. 因此逐个计算的对角分数之和一般不等于联合删除代价. 神经网络真实 Hessian 也未必正定，局部近似还可能在大幅剪枝时失效；常见工程方法使用正半定重建曲率、阻尼或分块近似，以控制计算和数值问题.

允许其他权重补偿

若删除一个权重时允许其他参数调整，可以求解

$$\min_{\Delta w}\frac{1}{2}\Delta w^{\mathsf{T}}H\Delta w,\qquad e_q^{\mathsf{T}}\Delta w=-w_q. \tag{34.28}$$

假设 $g=0$ 且 H 正定. 拉格朗日函数为

$$\mathcal{F}(\Delta w,\mu)=\frac{1}{2}\Delta w^{\mathsf{T}}H\Delta w+\mu(e_q^{\mathsf{T}}\Delta w+w_q). \tag{34.29}$$

驻点条件 $H\Delta w+\mu e_q=0$ 给出 $\Delta w=-\mu H^{-1}e_q$. 代入约束，得到

$$\mu=\frac{w_q}{(H^{-1})_{qq}},\qquad \Delta w^*=-\frac{w_q}{(H^{-1})_{qq}}H^{-1}e_q,\qquad \Delta\mathcal{L}^*=\frac{w_q^2}{2(H^{-1})_{qq}}. \tag{34.30}$$

这表达了利用参数耦合补偿删除误差的思想，属于 Optimal Brain Surgeon 的核心二阶框架. [189] 它并未说明在大模型中可以廉价形成完整逆 Hessian；$P \times P$ 的存储和求解成本通常不可接受.

取 $H = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$，$w = (1,2)^\top$，删除第一个权重. $H^{-1} = \frac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$，因而

$$\Delta w^* = (-1, 0.5)^\top, \qquad w' = (0, 2.5)^\top, \qquad \Delta\mathcal{L}^* = \frac{1}{2(\frac{2}{3})} = 0.75. \tag{34.31}$$

若只置零而不补偿，$\Delta w = (-1,0)^\top$，代价为 1. 补偿利用第二个权重分担变化，但这一结论只对设定的局部二次模型成立.

对单个线性层，可在校准输入上改用重建目标

$$\mathcal{L}_{\text{rec}}(\widehat{W}) = \frac{1}{N}\|(\widehat{W} - W)X\|_F^2. \tag{34.32}$$

对任一输出行的参数，曲率为 $\frac{2XX^\top}{N}$. 它可通过输入统计构造，但可能奇异，常需阻尼或稳定分解. SparseGPT 将可扩展的局部重建与补偿用于大模型剪枝，说明校准激活和执行约束可以共同进入压缩设计. [190] 校准目标仍不是全任务损失，分布外输入可能受到不同影响.

结构化剪枝的联动关系

结构化剪枝（Structured Pruning）删除完整通道、注意力头或层，使张量尺寸真正改变. 对前馈网络

$$f(x) = W_d\phi(W_u x + b_u) + b_d, \tag{34.33}$$

其中 $W_u \in \mathbb{R}^{h\times d}$、$W_d \in \mathbb{R}^{d\times h}$. 保留中间索引集合 S，令选择矩阵 $P_S \in \{0,1\}^{k\times h}$，则压缩结构为

$$\widehat{f}(x) = W_d P_S^\top \phi(P_S W_u x + P_S b_u) + b_d. \tag{34.34}$$

因此上投影删除对应行及偏置，下投影删除对应列，外部 d 维接口保持不变. 门控前馈结构有两条上投影路径，两者都应保留相同中间索引，并同步缩减下投影；只调整一侧无法保持逐元素乘法的语义.

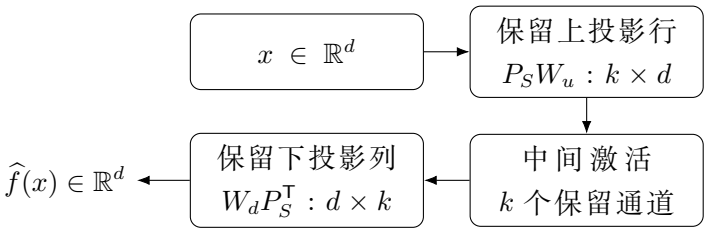


图 34.4: 前馈中间通道剪枝必须同步改变上下投影，保持外部残差接口.

取 $W_u = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$、$W_d = \begin{bmatrix} 1 & 0 & 0.1 \\ 0 & 1 & 0.1 \end{bmatrix}$，偏置为零，$\phi$ 为 ReLU．输入 $x = (1, 2)^\mathsf{T}$ 时中间激活为 $(1, 2, 3)^\mathsf{T}$，输出为 $(1.3, 2.3)^\mathsf{T}$．删除第三通道后，输出为 $(1, 2)^\mathsf{T}$，误差范数为 $\sqrt{0.3^2 + 0.3^2} \approx 0.4243$．权重数从 $3 \cdot 2 + 2 \cdot 3 = 12$ 降为 8，减少三分之一，但是否值得仍取决于任务与输入分布．

注意力头删除同样需要调整查询、键、值和输出投影中对应的布局，并检查分组查询的共享关系．删除整层会改变残差路径的深度；删除残差主宽度则还涉及归一化、嵌入与共享权重．结构化压缩必须输出一致的模型配置、参数形状与序列化格式，不能只修改某个参数文件．

34.3.2　稀疏性何时产生执行收益

稀疏率（Sparsity）为零元素比例，但稀疏矩阵仍需要存储非零值和位置．设非零数量为 k，数值每个 s 字节，索引每个 s_i 字节．以按行压缩的一种简化格式为例，存储约为

$$M_{\text{sparse}} \approx k(s + s_i) + (m + 1)s_i, \tag{34.35}$$

而稠密存储为 mns．忽略行指针时，只有 $\dfrac{k}{mn} < \dfrac{s}{s + s_i}$ 才节省空间．例如两字节数值配四字节索引，需要非零比例低于三分之一，才能抵消逐元素索引开销．实际块稀疏或位图格式可改变这个阈值，因此不能把它当作所有稀疏格式的统一门槛．

半结构化稀疏（Semi-structured Sparsity）限制每个小组内的非零数量，例如常见的 2:4 模式在规定分组中保留两个非零值．固定模式降低索引和调度复杂度，却约束了哪些权重能够保留．同样 50% 的零元素，任意分布与满足 2:4 布局的分布，对硬件可能完全不同．专用库对数据类型、矩阵维度、布局和稀疏格式有具体要求，cuSPARSELt 就明确面向受支持的结构化稀疏矩阵运算．[191]

稠密内核通常仍按完整矩阵形状计算零元素．使用稀疏内核后，索引读取、不规则访存、负载不均与较低计算密度也可能抵消运算减少．块结构越规则，越容易形成可用执行路径，但其误差预算也更受约束．剪枝前就应确定后端可消费的形式，避免训练完任意掩码后才发现只能还原成稠密矩阵运行．

剪枝后的恢复训练必须明确掩码是否固定．如果置零后仍对完整参数自由更新，零权重可能重新生长；如果只对保留结构训练，应确保优化器状态与结构同步．恢复训练能够调整保留参数，却不能保证任何压缩率下都恢复原质量．

34.4　低秩分解

34.4.1　低秩分解及最优近似

SVD 及谱尾误差

奇异值分解（Singular Value Decomposition，SVD）写成

$$W = U\Sigma V^{\mathsf{T}} = \sum_{i=1}^{q} \sigma_i u_i v_i^{\mathsf{T}}, \qquad q = \min(m,n), \quad \sigma_1 \geqslant \cdots \geqslant \sigma_q \geqslant 0. \tag{34.36}$$

左右奇异向量分别正交，奇异值描述对应输入方向到输出方向的放大程度．截断为秩 r 的矩阵

$$W_r = \sum_{i=1}^{r} \sigma_i u_i v_i^{\mathsf{T}} \tag{34.37}$$

舍弃较小奇异方向．由秩一分量在 Frobenius 内积下正交，

$$\|W - W_r\|_F^2 = \sum_{i=r+1}^{q} \sigma_i^2, \qquad \|W - W_r\|_2 = \sigma_{r+1}. \tag{34.38}$$

前式给出总平方误差，后式给出最大方向放大误差；$r = q$ 时尾项为零．截断 SVD 的最佳低秩近似性质由 Eckart–Young 定理刻画．[192]

下面说明 Frobenius 最优性．对任意秩不超过 r 的候选 Z，令 P 为其列空间上的正交投影，则 $(I-P)Z = 0$．投影分解使

$$\|W - Z\|_F^2 \geqslant \|(I-P)(W-Z)\|_F^2 = \|(I-P)W\|_F^2. \tag{34.39}$$

另一方面，

$$\|(I-P)W\|_F^2 = \operatorname{tr}(WW^{\mathsf{T}}) - \operatorname{tr}(PWW^{\mathsf{T}}). \tag{34.40}$$

在 WW^{T} 的正交特征基中，令 $a_i = u_i^{\mathsf{T}} P u_i$，则 $0 \leqslant a_i \leqslant 1$ 且 $\sum_i a_i = \operatorname{rank}(P) \leqslant r$，补零特征值后有

$$\operatorname{tr}(PWW^{\mathsf{T}}) = \sum_{i} \sigma_i^2 a_i \leqslant \sum_{i=1}^{r} \sigma_i^2. \tag{34.41}$$

最后一个不等式来自把至多 r 单位权重分配给降序的最大系数．因此任意 Z 的误差至少为谱尾平方和，而 W_r 达到这一界，最优性得证．[3]

[3]若截断处奇异值相同，最优子空间可能不唯一；这不改变最小误差值．矩阵的这一结论也不能直接推广为任意高阶张量分解都具有相同闭式最优解．

参数量及计算阈值

低秩分解（Low-rank Factorization）以两个较窄矩阵替换原矩阵. 可将奇异值分配到两因子，例如

$$A = U_r \Sigma_r^{\frac{1}{2}}, \qquad B = \Sigma_r^{\frac{1}{2}} V_r^{\mathsf{T}}, \qquad W_r = AB. \tag{34.42}$$

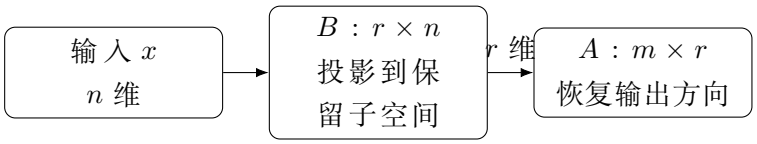
于是 $y \approx A(Bx) + b$. 两个线性层之间不能插入新的非线性，否则不再表示同一个低秩矩阵. 因子分配也非唯一，AD 与 $D^{-1}B$ 对任意可逆 D 给出相同乘积，但它们的数值范围和后续量化误差可能不同.

原权重数为 mn，因子权重数为 $r(m+n)$，因此严格减少参数需要

$$r < \frac{mn}{m+n}. \tag{34.43}$$

对方阵 $m = n = d$，阈值为 $r < \dfrac{d}{2}$，不是简单的 $r < d$. 若偏置只保留一次，前后偏置数量相同；若额外引入中间偏置，则应计入并说明它如何保持或改变函数.

对 N 个输入，原主要操作数约为 $2Nmn$，两因子约为 $2Nr(m+n)$. 这个计数忽略内核启动、中间张量和形状效率. 真实实现还可能需要额外写入和读取 $N \times r$ 的中间结果，因此参数与操作数阈值只是必要分析，不能直接推出相同比例加速.



两层之间无新增非线性；偏置在输出端保持

图 34.5: 低秩因子替换原矩阵，实际执行两次投影.

例 26.1

令 $W = \mathrm{diag}(4, 2, 1, 0)$. 它的平方 Frobenius 范数为 $16 + 4 + 1 = 21$. 秩 1 近似为 $W_1 = \mathrm{diag}(4, 0, 0, 0)$，误差平方为 5，相对误差为 $\sqrt{\dfrac{5}{21}} \approx 0.4880$. 可取

$$A = (2, 0, 0, 0)^{\mathsf{T}}, \qquad B = (2, 0, 0, 0), \tag{34.44}$$

以 8 个稠密存储参数替代 16 个参数. 输入 $x = (1, 1, 1, 1)^{\mathsf{T}}$ 时，原输出为 $(4, 2, 1, 0)^{\mathsf{T}}$，近似输出为 $(4, 0, 0, 0)^{\mathsf{T}}$，实际误差范数为 $\sqrt{5}$.

秩 2 近似保留 4 和 2，误差范数降为 1，但因子参数为 $2(4+4) = 16$，已经没有参数节省；秩 3 完全恢复原矩阵，因子参数却为 24. 例题表明秩越大误差越小，并不意味着每个秩都属于压缩方案. 实际模型若奇异值谱下降缓慢，低秩参数节省与质量预算之间可能根本没有共同可行区间.

34.4.2 输入加权的低秩目标

权重范数把各输入方向视为同等重要. 对真实输入，令 $E=\widehat{W}-W$，则

$$\mathbb{E}\|Ex\|_2^2=\mathbb{E}\,\mathrm{tr}(Exx^{\mathsf{T}}E^{\mathsf{T}}) \tag{34.45}$$

$$=\mathrm{tr}(EC_xE^{\mathsf{T}})=\|EC_x^{\frac{1}{2}}\|_F^2. \tag{34.46}$$

这里 $C_x=\mathbb{E}[xx^{\mathsf{T}}]$ 是二阶矩，不一定等于协方差；若输入均值非零，仍需保留均值贡献. 称为**激活加权低秩近似**（Activation-weighted Low-rank Approximation），是因为常见校准输入就是前一层激活.

当 C_x 正定时，设 $Z=WC_x^{\frac{1}{2}}$、$D=\widehat{W}C_x^{\frac{1}{2}}$. 右乘可逆矩阵不改变秩，故

$$\min_{\mathrm{rank}(\widehat{W})\leqslant r}\|(W-\widehat{W})C_x^{\frac{1}{2}}\|_F^2=\min_{\mathrm{rank}(D)\leqslant r}\|Z-D\|_F^2. \tag{34.47}$$

对 Z 做截断 SVD 得 Z_r，再变换回去，得到

$$\widehat{W}=Z_rC_x^{-\frac{1}{2}}. \tag{34.48}$$

这才是该输入分布与固定线性输出误差下的最优低秩解；一般不等于直接截断 W.

取 $W=\mathrm{diag}(3,2)$、$C_x=\mathrm{diag}(1,9)$. 普通 SVD 秩 1 近似保留第一方向，输出误差为 $2^2\cdot 9=36$. 加权矩阵 $Z=\mathrm{diag}(3,6)$ 则保留第二方向，回变换得到 $\widehat{W}=\mathrm{diag}(0,2)$，输出误差为 $3^2\cdot 1=9$. 后者的权重 Frobenius 误差平方为 9，比前者 4 更大，但在设定输入分布下输出误差更小. 这是目标不同导致的合理差异. 该二维例子用于解释加权，秩 1 因子不满足严格参数节省，不能作为压缩率证据.

有限校准集可估计 $C_x\approx\dfrac{XX^{\mathsf{T}}}{N}$. 当样本不足或激活相关时，该矩阵可能奇异；未被样本覆盖的方向没有得到约束. 可在有效支持子空间求解，或采用 $C_x+\eta I$ 的阻尼形式，但后者相当于给未观测方向加入额外惩罚，改变了原目标. 过小特征值的逆平方根还会放大数值误差，不能直接无条件求逆.

局部输出误差小也不保证最终任务误差小. 后续层对不同输出方向具有不同敏感度，生成中的分布变化还会改变后续输入. 恢复训练可以围绕压缩后激活重新调整，但必须使用与最终测试隔离的数据.

34.5 知识蒸馏

34.5.1 知识蒸馏及温度

硬标签和软标签

硬标签（Hard Label）用单个类别或目标词元表示监督，**软标签**（Soft Label）给出完整概率分布. 教师分布中不同错误类别的概率差异，可能表达相似性和不确定性；

也可能包含教师偏差. 蒸馏并不是宣称教师概率等于真实条件分布，而是让学生拟合一个具有额外结构的监督信号. 温度软化与硬软目标组合由经典知识蒸馏工作系统阐述. [193]

设教师和学生 logits 分别为 $z_T, z_S \in \mathbb{R}^C$，温度 $\tau > 0$，则

$$p_{T,j}^{\tau} = \frac{e^{\frac{z_{T,j}}{\tau}}}{\sum_k e^{\frac{z_{T,k}}{\tau}}}, \qquad p_{S,j}^{\tau} = \frac{e^{\frac{z_{S,j}}{\tau}}}{\sum_k e^{\frac{z_{S,k}}{\tau}}}. \tag{34.49}$$

增大温度通常使分布更平缓，减小温度使其更集中. 类别排序不因共同正温度缩放改变，但类别间的概率差和优化权重会改变. 蒸馏使用的训练温度不必等于部署采样温度，二者应分别记录.

若真实类别为 y，常见混合目标为

$$\mathcal{L} = (1-\lambda)[-\log p_{S,y}^{1}] + \lambda \tau^2 D_{\mathrm{KL}}(p_T^{\tau} \| p_S^{\tau}). \tag{34.50}$$

这里硬标签项使用温度 1，软标签项的教师与学生使用同一温度. 教师熵相对于学生参数是常数，因而 KL 与教师为目标的交叉熵具有相同学生梯度，但数值不同. 反转 KL 的方向会改变目标，不能因调用接口接受两组概率就随意交换.

温度平方因子的推导

教师冻结时，记 $p_j = p_{T,j}^{\tau}$、$q_j = p_{S,j}^{\tau}$，则

$$D = \sum_j p_j(\log p_j - \log q_j), \qquad \log q_j = \frac{z_{S,j}}{\tau} - \log \sum_k e^{\frac{z_{S,k}}{\tau}}. \tag{34.51}$$

对学生 logit $z_{S,l}$ 求导，

$$\frac{\partial \log q_j}{\partial z_{S,l}} = \frac{1}{\tau}(\mathbf{1}[j=l] - q_l). \tag{34.52}$$

代入 KL，利用 $\sum_j p_j = 1$，得到

$$\frac{\partial D}{\partial z_{S,l}} = -\frac{1}{\tau}\sum_j p_j(\mathbf{1}[j=l] - q_l) = \frac{q_l - p_l}{\tau}, \tag{34.53}$$

$$\frac{\partial(\tau^2 D)}{\partial z_{S,l}} = \tau(q_l - p_l). \tag{34.54}$$

一个 $\frac{1}{\tau}$ 来自 logits 缩放；另一个近似尺度需要进一步分析概率差，不能从链式法则中直接凭空得到.

softmax 对公共平移不变，因此分别中心化师生 logits：$\widetilde{z}_j = z_j - C^{-1}\sum_k z_k$. 在 τ 相对所有中心化 logit 幅度足够大时，

$$e^{\frac{\widetilde{z}_j}{\tau}} = 1 + \frac{\widetilde{z}_j}{\tau} + O(\tau^{-2}), \qquad p_j^{\tau} = \frac{1}{C} + \frac{\widetilde{z}_j}{C\tau} + O(\tau^{-2}). \tag{34.55}$$

故

$$q_j - p_j = \frac{\widetilde{z}_{S,j} - \widetilde{z}_{T,j}}{C\tau} + O(\tau^{-2}), \tag{34.56}$$

未缩放 KL 梯度主项为中心化 logit 差除以 $C\tau^2$. 乘以 τ^2 后,

$$\frac{\partial(\tau^2 D)}{\partial z_{S,j}} = \frac{\widetilde{z}_{S,j} - \widetilde{z}_{T,j}}{C} + O(\tau^{-1}). \tag{34.57}$$

因此温度平方因子补偿的是高温近均匀区域的主要梯度尺度. 它不保证所有温度、所有样本的梯度范数完全相等, 也不证明温度越高越好. 高温下不同类别近乎均匀, 小概率信息与噪声都可能被放大到更显著的监督位置.

高温极限还给出 $\tau^2 D \approx \dfrac{\|\widetilde{z}_S - \widetilde{z}_T\|_2^2}{2C}$. 这里必须中心化: 给教师全部 logits 加一个常数不改变分布, 不应改变蒸馏目标, 而未经中心化的平方误差会惩罚这个无意义平移.

蒸馏损失梯度

例 26.2

考虑两个类别, $\tau = 2$, 教师 logits 为 $(2\log 4, 0)$, 学生 logits 为 $(0, 0)$, 硬标签是第一类, $\lambda = \dfrac{1}{2}$. 温度化教师分布为 $(0.8, 0.2)$, 学生为 $(0.5, 0.5)$. 软目标 KL 为

$$D = 0.8\log(\frac{0.8}{0.5}) + 0.2\log(\frac{0.2}{0.5}) \tag{34.58}$$

$$= 0.8\log 1.6 + 0.2\log 0.4 \approx 0.192745, \tag{34.59}$$

乘以 $\tau^2 = 4$ 得 0.770979. 硬标签损失为 $-\log 0.5 \approx 0.693147$, 最终混合损失约为

$$\mathcal{L} = \tfrac{1}{2}(0.693147 + 0.770979) \approx 0.732063. \tag{34.60}$$

软目标对学生 logits 的梯度为

$$2[(0.5, 0.5) - (0.8, 0.2)] = (-0.6, 0.6), \tag{34.61}$$

硬标签梯度为 $(-0.5, 0.5)$, 混合后为 $(-0.55, 0.55)$. 梯度下降提高第一类 logit 并降低第二类 logit. 若忘记温度平方, 软项梯度变成 $(-0.15, 0.15)$, 即使混合系数未变, 硬软监督的相对强度也已改变.

教师若给出错误分布, 学生仍会沿教师方向更新. 混入硬标签可以提供另一信号, 但效果受标签质量和权重影响, 不能作为所有教师错误的自动修复. 比较蒸馏收益必须保留同结构学生仅使用硬标签训练的基线, 而不只与更大教师比较.

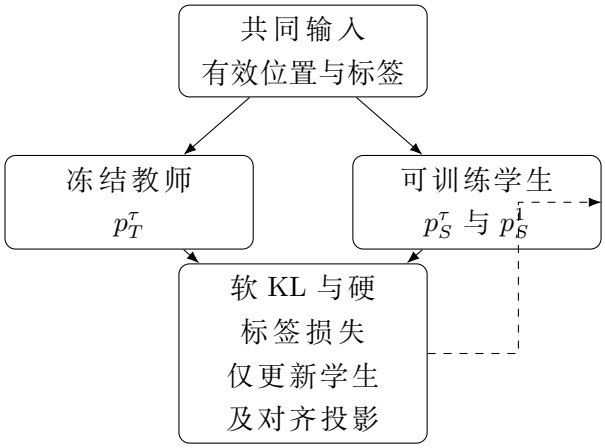


图 34.6: 输出分布蒸馏的数据与梯度边界．教师输出为固定目标，硬标签项与软标签项使用各自声明的温度.

34.5.2　语言模型的蒸馏契约

词元分布及数据位置

词元级蒸馏（Token-level Distillation）在给定前缀上对齐师生下一词元分布．若有效掩码为 m_{bt}，总有效位置数 $N_t = \sum_{b,t} m_{bt}$，可写为

$$\mathcal{L}_{\text{token}} = \frac{\tau^2}{N_t} \sum_{b,t} m_{bt} D_{\mathrm{KL}}(p_T^\tau(\cdot \mid h_{bt}) \| p_S^\tau(\cdot \mid h_{bt})). \tag{34.62}$$

师生必须使用相同可见前缀和对应目标位置．填充、提示监督区域、因果移位和样本装箱边界若不同，分布列相同也不能得到正确监督.

词表对齐更是必要条件．两个分词器的第 100 列可能代表不同文本单位，不能逐列计算 KL．即使通过映射合并相近词元，也必须定义概率质量如何守恒、不同切分怎样对应位置．无法建立可靠映射时，可以选择文本级序列监督，代价是放弃完整词表分布信息.

完整 logits 缓存大小约为 $N_t V s$ 字节，V 为词表大小、s 为每元素字节数．取最可能的若干类别可以降低存储，却会丢失尾部质量；截断后重新归一化得到的是另一个目标．缓存必须绑定样本标识、教师权重、分词器、模板、位置掩码和温度处理方式，避免内容正确但排列错位.

教师冻结包含参数不更新和目标生成方式稳定两个方面．训练模式下的随机失活仍可能改变教师分布；若有意使用随机教师集成，需要把这种随机性定义为目标的一部分．在线教师消耗额外算力，离线缓存消耗存储及版本管理成本，二者应按训练规模选择.

中间表示及注意力蒸馏

中间表示蒸馏（Intermediate Representation Distillation）对齐教师与学生的隐藏状态. 设 $H_S \in \mathbb{R}^{N_t \times d_S}$、$H_T \in \mathbb{R}^{N_t \times d_T}$，投影 $P_h \in \mathbb{R}^{d_S \times d_T}$，一种目标为

$$\mathcal{L}_h = \frac{1}{N_t} \| H_S P_h - H_T \|_F^2. \tag{34.63}$$

记 $E_h = H_S P_h - H_T$，则

$$\frac{\partial \mathcal{L}_h}{\partial H_S} = \frac{2}{N_t} E_h P_h^{\top}, \qquad \frac{\partial \mathcal{L}_h}{\partial P_h} = \frac{2}{N_t} H_S^{\top} E_h. \tag{34.64}$$

投影只承担训练对齐时，可以在部署移除；若它被加入学生推理路径，则应计入最终成本. 层数不同时需显式定义层映射，而不是比较同名层. 隐藏表示还存在旋转和尺度的不唯一性，直接逐元素拟合会施加比输出一致更强的约束.

注意力蒸馏（Attention Distillation）可以比较对应层或头的注意力分布，但要求查询与键位置可对齐. 教师多头与学生少头的映射、掩码与头聚合规则都会改变目标. 注意力相似并不充分保证值路径和最终输出相似；因此中间目标通常作为任务或输出目标的补充.

序列级蒸馏

序列级蒸馏（Sequence-level Distillation）以教师生成的完整文本作为学生监督，早期序列到序列研究系统讨论了这一方法. [194] 若教师序列分布为 $p_T(y \mid x)$，理想交叉熵为

$$\mathcal{L}_{\text{seq}} = -\mathbb{E}_{y \sim p_T(\cdot \mid x)} \log p_S(y \mid x). \tag{34.65}$$

使用独立教师样本可以形成蒙特卡洛估计；若只保留贪心或束搜索选出的 y^*，目标变成 $-\log p_S(y^* \mid x)$，不再是整个教师分布交叉熵的无偏估计. 它可能降低目标多样性，也可能让学生更容易学习集中的输出形式.

序列监督通过重新分词教师文本，可以允许师生使用不同词表，但内容质量、终止协议和数据分布仍需校验. 第二十六章《可验证推理训练》讨论了验证器筛选与推理轨迹蒸馏；筛选会改变教师条件分布，不能把被接受文本视为无偏教师样本.

输出分布、序列与中间表示三类蒸馏可以组合，但每增加一个目标都增加对齐条件. 教师的错误、偏见与不恰当置信度也可能被学生继承. 蒸馏数据和教师选择应与最终测试隔离，报告学生对真实标签和任务的表现，而不只报告“与教师更相似”.

34.6　组合压缩及误差传播

34.6.1　同一矩阵的组合误差

设第一次压缩得到 $W^{(1)} = W + E_1$，第二次得到 $W^{(2)} = W^{(1)} + E_2$．总误差为 $E_1 + E_2$，因此

$$\|E_1 + E_2\|_F^2 = \|E_1\|_F^2 + \|E_2\|_F^2 + 2\langle E_1, E_2\rangle_F. \tag{34.66}$$

两种压缩各自误差小，不意味着总误差平方等于它们之和；误差方向可能抵消，也可能加强．若剪枝改变了激活分布，原来的加权低秩或量化校准统计也不再精确对应后续输入．

对因子量化，原低秩近似为 AB，量化后为 $(A + \Delta A)(B + \Delta B)$，则

$$\widehat{W} - W = (AB - W) + \Delta AB + A\Delta B + \Delta A\Delta B. \tag{34.67}$$

于是谱范数上界为

$$\|\widehat{W} - W\|_2 \leqslant \|AB - W\|_2 + \|\Delta A\|_2\|B\|_2 + \|A\|_2\|\Delta B\|_2 + \|\Delta A\|_2\|\Delta B\|_2. \tag{34.68}$$

这说明因子尺度分配会影响后续误差：即使乘积相同，一个因子过大也可能放大另一个因子的扰动．

34.6.2　组合误差传播

例 26.3

对无偏置的两层网络 $f(x) = W_2\phi(W_1x)$，压缩后为 $\widehat{f}(x) = (W_2 + E_2)\phi((W_1 + E_1)x)$．若 ϕ 为 L_ϕ-Lipschitz 且 $\phi(0) = 0$，加减中间项得到

$$\|\widehat{f}(x) - f(x)\|_2 \leqslant \|E_2\|_2\|\phi((W_1 + E_1)x)\|_2 \tag{34.69}$$

$$+ \|W_2\|_2\|\phi((W_1 + E_1)x) - \phi(W_1x)\|_2 \tag{34.70}$$

$$\leqslant L_\phi[\|E_2\|_2(\|W_1\|_2 + \|E_1\|_2) + \|W_2\|_2\|E_1\|_2]\|x\|_2. \tag{34.71}$$

它揭示前层误差受到后层放大的机制，但可能很保守．残差、归一化和自回归反馈需要各自分析，不能把这个两层界直接当作整书所有模型的质量界．

用标量线性链展示组合项．原模型 $f(x) = 3(2x) = 6x$，第一步将内层权重 2 近似为 1.8，第二步将外层权重 3 近似为 2.7．两者分别相对原模型单独使用时，输出误差均为 $-0.6x$；组合后输出为 $2.7 \cdot 1.8x = 4.86x$，误差为 $-1.14x$．展开为

$$3(-0.2)x + (-0.3)2x + (-0.3)(-0.2)x = (-0.6 - 0.6 + 0.06)x. \tag{34.72}$$

两个单独的相对误差均为 10%，组合相对误差为 19%，既不是简单相加 20%，也不是平方和．真实网络中误差相关性更复杂，因此需要保留每个中间产物，才能解释后续变化．

34.6.3 组合顺序及恢复训练

压缩操作一般不可交换. 先剪枝再分解，与先分解再剪枝因子，约束的是不同集合；先改变通道再量化，会改变每通道范围；蒸馏到另一结构则重新定义了后续可压缩对象. 若目标是显著缩小架构，可先设计学生再蒸馏；若保留主体结构，可先做结构剪枝或低秩替换，再进行恢复和最终表示校准. 这些是降低重复工作的策略，而不是唯一正确顺序.

恢复训练 (Recovery Training) 使用保留参数适应压缩结构，可结合真实标签、原模型输出或中间状态. 它本身消耗数据与训练资源，也可能过拟合校准分布. 每次结构变化后都应重新确认权重共享、嵌入与输出头绑定、配置形状、稀疏掩码以及优化器状态；保存完整基线与中间版本，才能定位质量损失来源.

算法 34.3 按误差与执行约束构建压缩产物

输入：原模型、独立校准数据、恢复训练数据、质量预算与目标后端. 输出：具有明确结构、表示及来源记录的候选压缩模型.

1. 固定原模型和评估契约，列出各模块形状、共享关系与目标后端可执行的稀疏或低秩形式.
2. 在校准输入上估计通道敏感度或输入二阶矩，按预算产生剪枝掩码、保留索引或低秩因子. 记录目标函数与近似方法.
3. 同步修改关联模块及配置，重建实际推理结构；不要仅把稠密参数置零后宣称结构已缩小.
4. 按需要恢复训练或蒸馏. 教师及其输出版本冻结，目标位置对齐，恢复数据与最终测试隔离.
5. 结构稳定后按本章量化部分的方法确定量化表示和校准范围，保存每次组合前后的产物与误差证据.
6. 在目标负载上评估最终任务质量及实际执行路径，超出预算时回退到对应中间版本. 达到预定资源和质量要求后停止增加压缩操作.

不变量：每个参数形状与模型配置一致；输出接口和词表契约明确；后端实际消费所声明格式；最终测试不参与压缩阈值调优.

34.7　真实硬件收益及统一比较

方法	直接改变的对象	收益必须满足的条件
非结构化剪枝	权重支持集	稀疏格式和内核能抵消索引及调度开销.
结构化剪枝	通道、头或层形状	关联结构一致，新形状仍能有效映射硬件.
低秩分解	矩阵秩与因子结构	秩满足参数阈值，双投影成本低于原运算.
知识蒸馏	学生训练目标与可选架构	学生在质量约束下确实具有更低执行成本.
量化	数值表示与位宽	格式、布局与低精度内核匹配，转换成本受控.

若可压缩部分占原端到端时间比例 f，该部分加速 s_c 倍，其他部分不变，则理想加速比为

$$S = \frac{1}{(1-f) + \frac{f}{s_c}}. \tag{34.73}$$

例如线性计算占 60%，即便这部分加速两倍，端到端也只有 $\frac{1}{0.4+0.3} \approx 1.43$ 倍；这是时间分解的算例，并非设备测量. 实际压缩还可能改变内存流量、通信和调度，需重新测量各部分，不能把此式作为固定预测.

模型制品大小、加载后权重内存、键值缓存、工作区和峰值内存应分别记录. 减少权重未必同比减少长上下文 KV；增加中间低秩激活可能扩大工作区；结构化头删除可能影响缓存形状，但必须按具体结构重新计数. 第二十八章《算力基础设施》说明了计算与带宽的约束，压缩收益应对应到这些实际路径.

同样的压缩结构在不同批量和序列长度下可能具有不同表现. 小矩阵更容易受启动与带宽影响，大矩阵更可能受计算吞吐影响；稀疏布局打包的启动成本与稳态内核成本也应分开. 必须记录数值精度、后端版本、实际内核、输入输出长度、冷暖状态和并发，避免用一次文件缩小或单层操作数下降宣称完整服务提速.

压缩的三个完成条件

数学近似必须在明确误差目标下成立，模型结构与制品格式必须能够恢复并执行，任务质量和系统成本必须在同一负载契约下满足要求. 任何一个条件尚未建立，都不能由“参数更少”替代.

第五部分

模型架构拓展

第三十五章　双向语言建模

理解一段已给定文本时，当前位置的意义往往同时依赖左右两侧．例如，“苹果发布了新产品”与“苹果已经成熟”中的同一词语，需要后文帮助消除歧义．自回归模型为了生成必须限制未来信息，理解任务却可以利用完整观测．本章讨论如何在保留双向可见性的同时，构造有意义的自监督目标，以及如何把得到的上下文表示用于分类和抽取．

基于 Transformer 的双向编码表示（Bidirectional Encoder Representations from Transformers，BERT）以双向编码器作为共享主干，并在预训练与下游任务中配置不同输出头 [8]．它的代表性贡献是训练目标与表示结构的配合，而不只是某个特殊分类词元．本章先说明输入和信息流，再推导损坏输入上的恢复目标，最后讨论任务适配与评估边界．

假设与适用范围

输入是已知的有限词元序列，普通文本与控制词元的编号固定．填充位置不作为有效键，不参与目标统计．损坏输入由明确的随机通道产生，监督标签来自损坏前的原序列．以下 BERT 配方描述原始设计；不将其视为所有双向编码模型的统一常数．

35.1　双向编码输入

35.1.1　三类嵌入及其不同职责

对于句对 A, B，经典输入组织为

$$[\mathrm{CLS}],\ A,\ [\mathrm{SEP}],\ B,\ [\mathrm{SEP}]. \tag{35.1}$$

词元编码与特殊符号的规则见第四章《文本表示》．设批量大小为 B_0，填充后长度为 T，隐藏宽度为 d．使用 B_0 区别于句对中的第二个句段 B．三类编号分别选择词元、位置和句段嵌入：

$$R_{bt} = E[x_{bt}] + P[t] + S[s_{bt}], \qquad H^{(0)} = \mathrm{Dropout}(\mathrm{LN}(R)). \tag{35.2}$$

其中 $E \in \mathbb{R}^{V\times d}$ 为词元嵌入，$P \in \mathbb{R}^{T_{\mathrm{pos}}\times d}$ 为可学习绝对位置表，$S \in \mathbb{R}^{2\times d}$ 为经典句对的句段表．所有加数均为 d 维向量．位置上限约束实际使用的位置编号；仅把调用

参数中的最大长度改大，并不会自动产生训练过的新位置向量.

词元编号说明“出现了什么”，位置编号说明“位于何处”，句段编号说明“属于哪一段”. 句段编号不是自然语言句子数量的计数器，也不等于注意力隔离边界. 句对中的 A、B 虽具有不同句段编号，仍可通过双向自注意力互相读取. 若任务要求隔离两个片段，需要另行定义注意力允许关系.

符号	含义	形状或范围
$x, \widetilde{x}$	原词元及损坏后的输入编号	$B_0 \times T$
m_{bt}	输入有效性标记	$\{0,1\}$
$\mathcal{M}_b$	第 b 条序列被选中监督的位置集	有效非控制位置的子集
$H^{(\ell)}$	第 ℓ 层上下文表示	$B_0 \times T \times d$
U, q	MLM 词表分数及概率	$B_0 \times T \times V$
r_b	句对是否连续的二元目标	$\{0,1\}$
C	下游类别数	正整数

35.1.2 双向可见性及编码层

对于一个有效查询位置 i，双向注意力允许读取所有有效键 j. 定义加性掩码

$$M^{\mathrm{bi}}_{bij} = \begin{cases} 0, & m_{bj} = 1, \\ -\infty, & m_{bj} = 0. \end{cases} \tag{35.3}$$

它没有 $j \leqslant i$ 的约束. 用第五章《注意力机制》中的缩放点积注意力计算后，原始 BERT 的编码层采用注意力和前馈网络两条 Post-LN 残差子层：

$$A^{(\ell)} = \mathrm{LN}\big(H^{(\ell-1)} + \mathrm{Dropout}(\mathrm{MHA}(H^{(\ell-1)}, M^{\mathrm{bi}}))\big), \tag{35.4}$$

$$H^{(\ell)} = \mathrm{LN}\big(A^{(\ell)} + \mathrm{Dropout}(\mathrm{FFN}(A^{(\ell)}))\big). \tag{35.5}$$

前馈网络使用 GELU，原始配置中隐藏层通常扩展为 $4d$. 这与原始 Transformer 的 ReLU 配置、以及许多自回归模型的 Pre-LN 配置应分别陈述；各结构的归一化和梯度机制见第六章《Transformer 网络结构》.

多层双向传播使每个有效位置都可聚合整段观测. 填充查询行虽然可能得到一个非零输出，但它不应进入任务损失或池化统计. 是否把该行输出清零属于实现约定；如果将其所有键都屏蔽，则必须单独处理全屏蔽行的归一化，不能对全为负无穷的行直接求 Softmax.

35.1.3 掩码机制对照

为隔离可见性的作用，固定同一查询行的原始分数为 $(0, \log 2, \log 3)$，三个键都有效. 位于第 2 个位置的查询在双向情形下得到

$$a^{\mathrm{bi}} = \frac{1,2,3}{6} = (\frac{1}{6}, \frac{1}{3}, \frac{1}{2}). \tag{35.6}$$

因果情形屏蔽第 3 个键，并对剩余两项重新归一化，得到

$$a^{\text{causal}} = (\frac{1}{3}, \frac{2}{3}, 0). \tag{35.7}$$

若值向量在此简化为标量 $(1, 4, 10)$，两种输出分别为 6.5 和 3. 区别不仅在于未来项变为零，原来可见项的权重也因分母变化而改变. 若第 3 个位置实际上是填充，则双向模型同样得到 $(\frac{1}{3}, \frac{2}{3}, 0)$，说明“右侧”与“有效”是两个不同判断.

这个例题没有改变参数，不依赖训练，因而能把差异归因于掩码；它不证明双向模型的任务质量更高，也不把注意力权重直接解释为因果重要性.

35.2　损坏通道及掩码语言建模

35.2.1　掩码重建目标

如果一个双向编码器在每个位置读取原词元，又以原词元作为同位置监督，模型可能依赖输入查表形成近似恒等映射，而不必学习上下文. **掩码语言建模**（Masked Language Modeling，MLM）通过改变输入的一部分，再要求恢复原词元，降低这一捷径的作用.

设原序列为 x，先从允许的位置中抽取监督集合 $\mathcal{M}$，再用随机损坏通道 $c(\widetilde{x} \mid x, \mathcal{M})$ 产生观察输入. 目标写为

$$\mathcal{J}(\theta) = \mathbb{E}_{x, \mathcal{M}, \widetilde{x}} \left[-\frac{1}{|\mathcal{M}|} \sum_{t \in \mathcal{M}} \log q_\theta(x_t \mid \widetilde{x}) \right], \qquad |\mathcal{M}| > 0. \tag{35.8}$$

模型读到的是 $\widetilde{x}$，标签仍是 x_t. 它一般不是只读取“删除 $\mathcal{M}$ 后的文本”：被选中位置可能被替换为随机词元，也可能保留原值. 显式写出损坏通道才能准确描述训练分布.

35.2.2　经典损坏配方及概率含义

原始 BERT 从可候选位置中选择约 15% 作为预测位置；条件于一个位置已被选中，约 80% 替换为 MASK，10% 随机替换，10% 保留原词元 [8]. 按这些理想概率计算，全部候选位置中约 12% 进入 MASK 分支，1.5% 进入随机替换分支，1.5% 进入保留分支.

监督集合与 MASK 出现位置并不相同. 只对输入中实际等于 MASK 的位置计算损失，会漏掉随机替换和保留分支. 随机替换也可能恰好抽到原词元；上述 1.5% 是进入该分支的比例，不是保证实际改变内容的比例. [1]

[1]若从含原词元的 V 个候选中均匀抽样，随机替换分支实际改变内容的概率为 $1 - \frac{1}{V}$. 若候选集合排除特殊词元或原词元，则需按实际抽样集合重新计算.

保留与随机替换使模型不能仅靠输入是否为 MASK 判断一个位置是否参与目标，也减轻了预训练时总在特殊符号处预测而下游输入没有 MASK 的不一致. 它们并不完全消除复制捷径：保留分支确实提供原值，训练目标只是让多种损坏条件共同贡献梯度.

候选位置的抽取方式还需说明. 逐位置伯努利抽样在短序列上可能产生空集合；固定个数抽样则引入位置之间的依赖. 整词遮盖或连续片段遮盖改变条件信息和任务难度，不能认为它们只改变了数据读取速度.

35.2.3 动态损坏及评估口径

每次访问文本重新抽取损坏，会让同一原文在不同轮次产生不同条件；预先固化损坏则形成有限的训练样例集合. 两种做法在有限预算下的监督覆盖不同. 验证时可固定原文、位置和替换结果，以比较参数变化；也可在多次独立损坏上求平均，但需报告抽样方式和不确定性.

对批量监督位置总数 $N_M = \sum_b |\mathcal{M}_b| > 0$，按词元加权的损失是

$$\mathcal{L}_{\mathrm{MLM}} = -\frac{1}{N_M} \sum_b \sum_{t \in \mathcal{M}_b} \log q_\theta(x_{bt} \mid \widetilde{x}_b). \tag{35.9}$$

这不总等于式 (35.8) 中先对每条序列求均值再平均的经验版本. 序列被选中位置数不同时，两种归一化赋予文本不同权重. 只有明确目标后，才能判断梯度累积和分布式归约的分母是否正确.

算法 35.1　构造一个 MLM 训练批次

输入：原序列、有效性标记、控制词元位置、损坏规则、随机状态. **输出：**损坏输入、原值标签、监督集合及有效监督数.

1. 从有效且允许监督的位置构造候选集合；禁止用填充编号的相等比较替代位置来源判断.
2. 按给定抽样方式选取 $\mathcal{M}_b$，复制原输入得到 $\widetilde{x}_b$；先保存每个选中位置的原词元标签.
3. 对每个选中位置抽取分支：替换 MASK、从规定集合随机替换，或保留原值. 标签在三种情况下均不改变.
4. 构造只读取有效键的双向掩码，并统计 N_M. 若 $N_M = 0$，按预定策略重新采样或跳过批次，不除以零.
5. 编码损坏输入，计算选中位置的词表损失，按式 (35.9) 归一化；反向完成后更新参数.

不变量：监督标签来自损坏前的文本；参与监督的位置由集合决定，而非由损坏后的词元值决定．构造过程线性扫描批量位置，编码器计算成本另计．

35.3　词元预测损失

35.3.1　词表分布映射

设共享主干输出为 $H \in \mathbb{R}^{B_0 \times T \times d}$．MLM 头先对每个位置施加非线性变换，再映射到词表：

$$G_{bt} = \mathrm{LN}(\mathrm{GELU}(H_{bt} W_h + b_h)), \tag{35.10}$$

$$U_{bt} = G_{bt} E^{\mathsf{T}} + b_V, \qquad q_{bt} = \mathrm{softmax}(U_{bt}). \tag{35.11}$$

$W_h \in \mathbb{R}^{d \times d}$，$b_h \in \mathbb{R}^d$，$b_V \in \mathbb{R}^V$；输入词元表 E 同时作为输出投影权重．共享不省去全词表分类计算，但减少独立参数，并使查表路径与预测路径的梯度相加．其一般机制见第四章《文本表示》．

训练只需要选中位置的分数时，可先从 H 收集 N_M 行再执行输出头，将输出张量从 $B_0 T \times V$ 缩小为 $N_M \times V$．由于这里的变换逐位置独立，这不改变选中位置的数学结果；主干仍必须编码完整损坏上下文，不能只保留待预测的几个词元．

考虑三条序列各贡献一个监督位置：第一处进入 MASK 分支，第二处随机替换，第三处保留原值．为便于核对，假设目标词表仅有三类，原标签依次为第 1、2、3 类，预测分布为

$$Q = \begin{bmatrix} .6 & .3 & .1 \\ .2 & .5 & .3 \\ .1 & .2 & .7 \end{bmatrix}. \tag{35.12}$$

三个位置全部计入损失，因此

$$\mathcal{L} = -\frac{1}{3}(\log .6 + \log .5 + \log .7) = -\frac{1}{3}\log .21 \approx 0.520216. \tag{35.13}$$

根据第三章《神经网络基础》中的交叉熵求导，分数梯度为

$$\overline{U} = \frac{1}{3}(Q - I_3) = \frac{1}{3}\begin{bmatrix} -.4 & .3 & .1 \\ .2 & -.5 & .3 \\ .1 & .2 & -.3 \end{bmatrix}. \tag{35.14}$$

每行梯度和为零．若只保留 MASK 行，所得损失变成 $-\log .6 \approx 0.510826$，数值可能看似相近，却删除了另两行的全部监督，并改变归一化后的梯度尺度．损失数字接近不能证明目标等价．

输出权重共享时，预测路径产生 $\overline{E}_{\mathrm{out}} = \overline{U}^{\mathsf{T}} G$；输入路径还产生按损坏后词元编号聚合的 $\overline{E}_{\mathrm{in}}$．最终梯度为两者之和．这里输入编号可能是 MASK 或随机词元，而输出目标仍是原词元，它们参与同一参数表的不同路径．

35.4　句对目标及共享主干

35.4.1　下一句预测目标

下一句预测（Next Sentence Prediction，NSP）使用句对标签 $r \in \{0,1\}$ 区分构造的连续句段与随机负句段. 经典正例来自语料中的后续片段，负例由随机选择另一片段形成，配方约为一半正例、一半负例. 它预测的是样例构造关系，不等于判断两个句子的逻辑蕴含，也不证明模型掌握时间因果关系.

取 CLS 位置的最终向量 h_c，经池化变换 $z = \tanh(h_c W_p + b_p)$ 后，二元分数为 $a = w^{\mathsf{T}} z + b$. 以 $\sigma(a)$ 表示正例概率，损失为

$$\ell_{\mathrm{NSP}} = -r \log \sigma(a) - (1-r)\log(1-\sigma(a)), \qquad \frac{\partial \ell_{\mathrm{NSP}}}{\partial a} = \sigma(a) - r. \tag{35.15}$$

NSP 两样本损失　若正例预测概率为 .8，负例被预测为正例的概率为 .3，两例平均损失为 $-\dfrac{\log .8 + \log .7}{2} \approx 0.289909$. 训练用稳定的分数形式计算，不先取极小概率再取对数.

随机负例若与正例有明显文体或来源差异，模型可能依靠这些线索完成分类. 句对目标的难度和有效性取决于负例构造，不能由正负数量平衡推出.

35.4.2　多任务损失及梯度汇合

设句对平均损失为 $\mathcal{L}_{\mathrm{NSP}}$，联合目标可写为

$$\mathcal{L} = \mathcal{L}_{\mathrm{MLM}} + \lambda \mathcal{L}_{\mathrm{NSP}}, \qquad \lambda \geqslant 0. \tag{35.16}$$

对共享主干参数 θ_e，有

$$\nabla_{\theta_e} \mathcal{L} = \nabla_{\theta_e} \mathcal{L}_{\mathrm{MLM}} + \lambda \nabla_{\theta_e} \mathcal{L}_{\mathrm{NSP}}. \tag{35.17}$$

两任务的梯度可能协同，也可能冲突. MLM 按监督词元数归一化，NSP 按句对数归一化，即使 $\lambda = 1$，也不意味着两者对主干的影响同样大. 比较损失权重时应同时考虑归一化口径和梯度尺度.

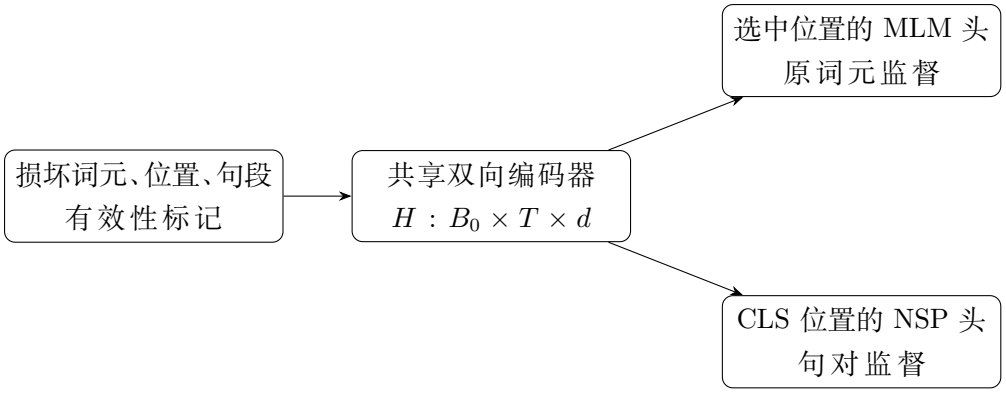


图 35.1: 共享表示与两类监督. 词元损坏不改变句对构造标签；两个头的梯度在共享主干汇合.

35.5　分类、标注及抽取任务

35.5.1　序列级分类及池化

对 C 类互斥标签，可令 $u = zW_c + b_c \in \mathbb{R}^C$，最小化序列交叉熵. 预训练头与下游标签空间通常不同，下游分类头需要新的参数和标签映射. CLS 向量是一个具有完整上下文的指定位置，其任务能力来自训练，不是该字面名称自动保证的.

还可以对有效位置取平均，得到

$$z_b = \frac{\sum_t m_{bt} H_{bt}}{\sum_t m_{bt}}. \tag{35.18}$$

若只池化普通文本，应进一步排除控制词元，并使用相应分母. 平均池化与 CLS 池化是不同的读出函数，应在任务数据上比较，而不是假定两者等价. 直接把预训练隐藏向量用于余弦检索，也不能自动获得适合语义相似度的度量空间；可能需要相应对比训练.

35.5.2　词元级标注及子词对齐

命名实体识别（Named Entity Recognition，NER）等任务对每个文本位置预测类别，分数形状为 $B_0 \times T \times C$. 原始标注若以词或字符为单位，而模型输入为子词，就必须建立对齐规则. 一个原词拆成三个子词时，可只监督首子词，也可按标注语义传播标签；两种策略的有效位置数和损失权重不同.

特殊词元与填充通常不接受实体标签. 若采用带边界语义的标签体系，不能简单复制一个词首标签到所有续接子词；解码后还需恢复合法片段并映射到原文偏移. 独立位置分类不保证全局标签序列合法，是否增加结构化解码应由任务约束决定.

35.5.3　抽取式问答的起止分布

给定问题与包含答案的文段，编码器输出每个位置的向量. 两组参数分别给出起点和终点分数：

$$a_t = w_s^{\mathsf{T}} H_t, \qquad b_t = w_e^{\mathsf{T}} H_t, \quad p_s = \mathrm{softmax}(a), \quad p_e = \mathrm{softmax}(b). \tag{35.19}$$

只在允许作为答案的文段位置上归一化. 真值为 (i^*, j^*) 时，可以使用

$$\ell = -\log p_s(i^*) - \log p_e(j^*). \tag{35.20}$$

独立起止分布可能把最高概率终点放在起点之前，因此推断时需在 $i \leqslant j$、最大长度及允许位置约束下选择片段，例如最大化 $a_i + b_j$．这是带约束的解码，而非直接分别取两个最大值．无答案样例还需要明确的拒答位置或额外分类目标．

长文段切窗时，要保存窗口到原文的偏移、答案是否完整落入窗口以及重叠窗口的合并规则．切窗只改变输入覆盖，不会自动保证每个样本仍包含正确答案；对不含答案的窗口错误复制原标签会制造训练噪声．

35.6　双向目标同自回归目标的差异

35.6.1　条件预测及联合概率

自回归目标利用链式法则定义 $p(x) = \prod_t p(x_t \mid x_{<t})$．MLM 则在损坏输入条件下恢复部分位置，并不直接给出这个有序生成分解．若同时遮盖多个位置，常见输出头分别预测各位置，并未在这些待恢复变量之间建立逐步生成条件．

例如，两处缺失内容在真实条件下只能取 (a, a) 或 (b, b)，两种组合各占一半．两个位置的边缘分布均为 $(.5, .5)$．若独立从两个边缘抽样，将以总概率 .5 产生不合法的 (a, b) 或 (b, a)．边缘预测准确不意味着联合恢复准确．

伪似然归一化反例　还可以对每个位置单独遮盖并把条件概率相乘，得到类似伪似然的分数．但它通常不在所有序列上归一化．以二元变量 x_1, x_2 为例，若两个条件均赋予“相同”为 .9、“不同”为 .1，两个同值序列的乘积各为 .81，两个异值序列各为 .01，总和为 1.64 而非一．把该乘积直接称为整句概率就是错误的．

目标与结构必须配合

双向编码器可以在给定观测中读取左右上下文；掩码目标通过损坏输入建立恢复任务．它们并不自动提供严格自回归生成协议．若采用迭代填空生成，还必须定义更新次序、采样规则、终止条件和质量评估．

35.6.2　比较模型时保持任务单位一致

MLM 损失取决于损坏率、损坏方式、位置抽样和有效监督数．它不能直接与因果语言模型的每词元困惑度比较，甚至两个不同损坏配方下的 MLM 损失也不具有自动一致的难度．比较表示能力应在相同下游任务、数据隔离、适配预算和指标条件下进行．

部署时必须同时恢复分词器、配置、主干、任务头和标签映射．仅恢复主干而让分类头随机初始化，会得到形状正确但语义无效的预测．训练模式下的随机失活、位置上限、句段编号与填充掩码均属于预测函数的组成部分，需纳入一致性检查．保存与重载的正确性证明的是制品一致，不能代替独立任务评估．

第三十六章　混合专家模型

稠密前馈层对每个词元应用同一组参数. 增大前馈宽度可以增加表达能力，但也使每个词元承担更多矩阵计算. 混合专家模型试图部分分离参数容量与单次计算：保存多组可学习变换，由输入决定本次调用其中哪些变换. 然而，减少参与计算的参数并不自动减少全部权重的存储，也不自动降低跨设备通信与执行尾部.

本章承接第三章《神经网络基础》的链式求导，从专家混合的函数形式出发，推导稀疏路由、分发与合并、容量约束以及路由器的梯度. 随后用完整数值例题说明权重归一化、专家负载和溢出处理如何改变输出，最后讨论共享专家、软专家混合与专家并行. 具体模型家族的结构差异在开源模型章节展开，集群拓扑和分布式训练将在相应章节继续细化.

36.1　条件计算及专家结构

36.1.1　专家函数的定义

混合专家模型（Mixture of Experts，MoE）包含若干专家函数 F_e，并由**路由器**（Router）或**门控网络**（Gating Network）确定各专家对当前输入的贡献. 专家通常是结构相同、参数独立的前馈网络，但“结构相同”不要求其训练后的函数相同. 名称中的专家描述条件化计算组件，不能据此将某一编号直接命名为数学、法律或编程专家. [1]

在 Transformer 中，常见做法是用 MoE 替换部分逐位置前馈子层，注意力和残差主干仍保留. 稀疏门控使一次输入只调用少量专家，是**条件计算**（Conditional Computation）的一种形式 [195]. 它并不等于为每种任务部署一个完整语言模型，也不等于多个独立模型生成答案后再投票.

设一个专家采用门控前馈形式，行向量输入 $x \in \mathbb{R}^{1\times d}$ 的输出为

$$F_e(x) = \left[\phi(xW_{e,g}) \odot (xW_{e,u})\right] W_{e,d}, \tag{36.1}$$

其中 $W_{e,g}, W_{e,u} \in \mathbb{R}^{d\times d_f}$，$W_{e,d} \in \mathbb{R}^{d_f\times d}$，$\phi$ 是固定的激活函数. 每个专家都输出宽度

[1] 专家编号是模型内部的参数索引. 对专家参数及路由输出通道作一致置换，可以表示同一个函数，因此编号本身不具有稳定的外部语义. 专家是否形成可解释分工，需要独立的行为与干预证据.

d，以便加权合并并与残差相加．普通两层前馈专家也可使用相同路由机制；专家内部结构与路由结构是两个可分别改变的维度．

36.1.2 统一符号及适用假设

将当前路由组内的有效词元表示展平为 $X \in \mathbb{R}^{N\times d}$．这里 N 是实际参与路由的位置数，可以来自多个序列，但不把填充位置算入其中．路由组可能只覆盖一个本地微批次，也可能覆盖一个专家并行组；统计量必须明确采用哪一种范围．

符号	含义	形状或范围
N, d, d_f	有效词元数、模型宽度、专家中间宽度	正整数
E_r, k	路由专家数、每词元所选专家数	$1 \leqslant k \leqslant E_r$
W_r, b_r	路由器权重和偏置	$d \times E_r$，E_r
G, P	路由分数与全专家概率	$N \times E_r$
S_t	词元 t 选中的专家集合	大小为 k
a_{te}	稀疏混合权重	非负标量
n_e, C_e	专家申请负载与容量	非负整数
D_e	词元到专家槽位的选择矩阵	$C_e \times N$
H_e, Y	专家输出与合并结果	$C_e \times d$，$N \times d$
$f_e, \bar{p}_e$	硬分配比例与平均路由概率	无量纲
$P_{\text{total}}, P_{\text{active}}$	总参数量、每词元激活参数量	参数个数

本章用 E_r 表示专家数量，避免与嵌入矩阵混用．P 在路由部分表示专家概率矩阵，带下标的 P_{total} 等表示参数计数．

基本稀疏路由模型

路由组含 $N > 0$ 个有效词元；各专家输入输出宽度相同；每个词元先选择 k 个不同专家；分数有限且并列选择规则明确．先讨论没有容量溢出的情况，再显式加入容量与拒绝规则．对梯度进行局部推导时，假定选择集合和接收集合保持不变．

36.1.3 稠密混合及稀疏混合

路由器先计算

$$G = XW_r + \mathbf{1}_N b_r^{\mathsf{T}}, \qquad p_{te} = \frac{\exp(g_{te})}{\sum_{j=1}^{E_r} \exp(g_{tj})}. \tag{36.2}$$

Softmax 沿专家轴归一化．**稠密专家混合**（Dense Mixture of Experts）对每个词元执行所有专家，再计算

$$y_t = \sum_{e=1}^{E_r} p_{te} F_e(x_t). \tag{36.3}$$

概率可以接近零，但除非执行过程真正跳过相应专家，其算术开销仍然存在. “权重很小”与“没有执行”不是同一个条件.

稀疏专家混合（Sparse Mixture of Experts）只执行选中集合 S_t 中的专家：

$$S_t = \mathrm{TopK}(g_{t,:}, k), \qquad y_t = \sum_{e \in S_t} a_{te} F_e(x_t). \tag{36.4}$$

Top-k 表示取最高的 k 个分数对应的不同编号. 由于 Softmax 对同一行保持分数次序，在没有额外变换时，按分数或概率选择得到同一集合. 但概率如何用于合并，仍有不同约定.

36.2　路由权重及局部梯度

36.2.1　两种归一化约定

一种常见形式在选中专家内重新归一化：

$$a_{te} = \begin{cases} \dfrac{\exp(g_{te})}{\sum_{j \in S_t} \exp(g_{tj})}, & e \in S_t, \\ 0, & e \notin S_t. \end{cases} \tag{36.5}$$

此时每个词元保留的混合权重和为一. 也可以先对全部专家归一化，再直接保留选中的概率，即 $a_{te} = p_{te}\mathbf{1}[e \in S_t]$. 后者保留权重之和一般小于一；输出幅度因而也受到路由置信度影响. Switch 的单专家路由保留了被选专家的全专家门控概率 [196]，不能把它改写成选中集合内重新归一化后仍声称函数相同.

两种形式在 $k = 1$ 时差异尤其明显. 选中集合内归一化会得到唯一权重为一，在固定选择集合的邻域内，任务损失不能再通过这个恒为一的权重更新路由分数. 保留全专家概率则仍有连续梯度路径. 路由器是否能从任务损失学习，不能只看代码里是否出现 Softmax，还要看后续归一化是否消去了其作用.

并列分数处需要一个明确的编号选择规则；选择边界附近，微小浮点扰动也可能改变专家集合. 确定性并列规则可以使重复执行更可解释，但不消除硬选择本身的非光滑性.

36.2.2　专家及路由器梯度

暂时固定一个词元的选中集合，并省略词元下标. 令专家输出 $z_e = F_e(x)$，混合结果 $y = \sum_e a_e z_e$，上游梯度为列向量 $h = \dfrac{\partial \mathcal{L}}{\partial y}$. 定义标量敏感度 $r_e = h^{\mathsf{T}} z_e^{\mathsf{T}}$，则

$$\frac{\partial \mathcal{L}}{\partial a_e} = r_e, \qquad \frac{\partial \mathcal{L}}{\partial z_e} = a_e h^{\mathsf{T}}. \tag{36.6}$$

专家参数的梯度由其局部网络反向传播获得，并乘以该词元的混合权重．未选专家不接收这个词元经专家执行路径产生的梯度，但可能从同批其他词元、正则项或其他共享路径获得贡献．

对式 (36.5)，选中集合内的 Softmax 雅可比为 $a_e(\mathbf{1}[e=j]-a_j)$，所以

$$\frac{\partial \mathcal{L}}{\partial g_j} = \sum_{e\in S} r_e a_e(\mathbf{1}[e=j]-a_j) \tag{36.7}$$

$$= a_j\left(r_j - \sum_{e\in S} a_e r_e\right), \qquad j\in S. \tag{36.8}$$

未选分数的这一局部梯度为零．式中括号比较某个专家的输出敏感度与当前混合平均值；因此路由器可以学习改变已选专家之间的连续权重．

若保留全专家概率，令 $r_e=0$ 对所有未选专家成立，则同样的 Softmax 反向得到

$$\frac{\partial \mathcal{L}}{\partial g_j} = p_j\left(r_j - \sum_{e\in S} p_e r_e\right), \qquad j=1,\dots,E_r. \tag{36.9}$$

未选分数也可因全局分母收到梯度，但这不意味着未选专家已经执行或获得同样的参数梯度．必须区分路由器参数与专家参数．

最后，若所有词元的分数梯度构成 G_G，路由器线性层满足

$$\frac{\partial \mathcal{L}}{\partial W_r} = X^{\mathsf{T}} G_G, \qquad \frac{\partial \mathcal{L}}{\partial b_r} = G_G^{\mathsf{T}}\mathbf{1}_N. \tag{36.10}$$

输入梯度还应加上所有已选专家的回传与路由器的回传．漏掉任一路径，都不再对应完整混合函数的导数．

36.2.3　硬选择的导数边界

Top-k 的编号集合在分数排序不变的小邻域中保持常数，在排序交换处发生跳变．通常的反向传播沿已选择路径求导，不会自动告诉模型“如果换成另一个专家，损失会怎样”．加噪声、辅助目标或其他选择机制可以改变探索和学习行为，但每种方法都需说明自己的前向与反向定义．

如果所有词元过早集中到少数专家，这些专家得到更多任务训练机会，其他专家则缺少有效更新，可能形成自我强化的负载差异．负载均衡机制因此不仅是运行调度，也影响训练过程中不同参数获得监督的分布．不过，强制每个专家处理完全相同数量、完全相同类型的输入也不是模型质量的充分条件．

36.3　词元路由

36.3.1　词元轴的专家批次变换

MoE 的输出仍按原词元顺序排列，但专家计算希望把送往同一专家的词元放在同一个批次中. **分发**（Dispatch）将原词元重排或复制到专家局部缓冲，**合并**（Combine）则将处理后的结果按原位置加权累加.

专家 e 有 C_e 个槽位，定义选择矩阵 $D_e \in \{0,1\}^{C_e \times N}$. 如果原词元 t 被接收到槽位 c，令 $(D_e)_{ct} = 1$；一个槽位至多接收一个词元. 于是

$$X_e = D_e X, \qquad H_e = F_e(X_e), \tag{36.11}$$

其中 F_e 对每个有效槽位按行计算. 再定义合并矩阵 $A_e \in \mathbb{R}^{N \times C_e}$，将对应位置设为接收后的权重，得到

$$Y = \sum_{e=1}^{E_r} A_e H_e. \tag{36.12}$$

没有拒绝且采用原权重时，$(A_e)_{tc} = a_{te}(D_e)_{ct}$. 这一写法用于说明映射关系，不要求运行时物化巨大稀疏矩阵；索引列表、前缀和与分组缓冲可以表达同样的关系.

图 36.1 概括了分组与回收路径. 一个词元选择两个专家时，分发阶段逻辑上产生两份输入引用，合并时必须将两份结果相加. 只保留最后返回结果会丢失一条专家路径. 空槽位即使因偏置产生非零专家输出，其合并权重也必须为零；否则填充缓冲会污染真实词元.

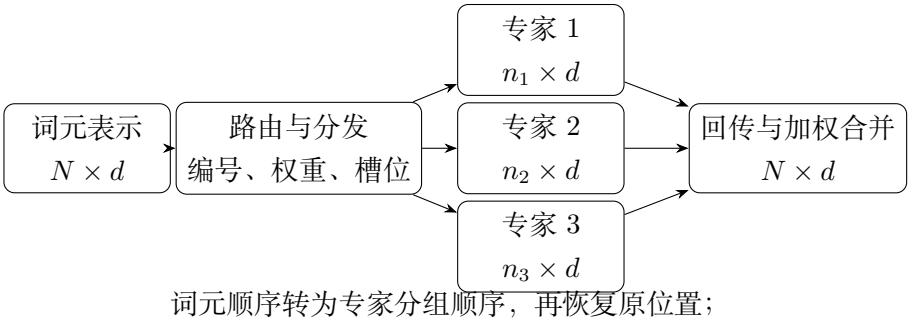


图 36.1: 稀疏 MoE 的数据流. 专家批次大小由路由决定；选择多个专家意味着多条返回路径.

36.3.2　路由容量及溢出

令申请负载 $n_e = \sum_t \mathbf{1}[e \in S_t]$，无拒绝时有 $\sum_e n_e = Nk$，平均负载为 $\dfrac{Nk}{E_r}$. 一种容量规则是

$$C = \left\lceil \gamma \frac{Nk}{E_r} \right\rceil, \tag{36.13}$$

其中 $\gamma > 0$ 为**容量因子**（Capacity Factor），所有专家暂取相同容量. Switch 中的单专家形式对应 $k = 1$[196]；其他系统可能按局部分组、不同专家宽度或不同对齐规则定义容量，不能只比较容量因子的数字.

即使总槽位 $E_r C \geqslant Nk$，局部专家仍可能溢出. 例如一个专家收到大量申请时，其他专家的空槽位不能自动代替它执行同一个函数. 容量通常与静态缓冲大小、通信消息形状和最坏负载相关，提高容量可以减少拒绝，却可能增加填充计算和临时空间.

超过容量后的处理必须成为模型定义的一部分. 拒绝某条专家路径，可以使该路径贡献为零；重新路由会调用另一组参数；等待或变长分组可以保留全部路径，但付出动态调度和尾延迟代价. 被称为“不丢词元”的实现通常是保留全部路由计算，而非移除了负载和通信约束.

若一个词元的全部专家路径被拒绝，MoE 子层贡献可能为零，残差主干仍可保留该位置. 这里的“丢词元”常指跳过该层的专家处理，不是从序列中删除编号，也不是将其监督标签自动移除. [2]

36.3.3 接收规则可能引入批次依赖

容量竞争发生在路由组内. 若采用先到先得，后部位置可能更容易被拒绝；若按门控权重排序，则低权重路径更容易失去计算. 于是同一词元在不同批次组成中，可能因为其他词元的申请而获得不同输出. 这与单纯逐词元前馈函数的性质不同.

对于严格自回归训练，还应警惕一种间接依赖：若先观察整段序列的路由权重，再按全局排名决定早期词元能否获得容量，未来词元可能影响早期位置的接收结果. 避免这一问题需要容量方案和接收规则与因果定义相容，而不能仅在注意力层放置因果掩码. 实际设计应区分前向内容读取、容量竞争以及训练辅助统计三类关系.

36.4 负载均衡及路由稳定性

36.4.1 硬负载及软概率描述不同对象

定义容量截断前的分配比例和平均概率：

$$f_e = \frac{1}{Nk} \sum_{t=1}^{N} \mathbf{1}[e \in S_t], \qquad \bar{p}_e = \frac{1}{N} \sum_{t=1}^{N} p_{te}. \tag{36.14}$$

二者各自之和为一. f_e 衡量实际申请的离散次数，$\bar{p}_e$ 衡量路由器的连续概率质量. 一个专家经常以第二名被选中，可能有较大的申请次数，却不拥有最大的平均概率，因此两种统计不能互换.

[2] 不同论文与运行时对 token dropping 的统计口径可能不同：拒绝的专家分配数、至少失去一条路径的词元数、以及失去全部路径的词元数，应分别报告.

若只统计容量截断后的负载，被饱和容量裁平的结果可能看起来很均衡，却掩盖大量溢出．应同时保留申请数、接收数、拒绝数和执行耗时．若专家宽度不同，次数均衡也不等于算术量均衡．

36.4.2　辅助均衡目标及其梯度范围

Switch 使用硬分配比例与平均概率乘积构造辅助目标．按照式 (36.14) 的 top-k 归一化，本章讨论其一种相应扩展：

$$\mathcal{L}_{\mathrm{bal}} = E_r \sum_{e=1}^{E_r} f_e \bar{p}_e, \qquad \mathcal{L} = \mathcal{L}_{\mathrm{task}} + \lambda \mathcal{L}_{\mathrm{bal}}. \tag{36.15}$$

其中 $\lambda \geqslant 0$ 控制任务目标与均衡目标的相对权重，top-1 时恢复相应统计定义．不同实现对 k、序列和设备的归一化可能不同，系数不能脱离公式移植．

在反向时通常将硬计数 f_e 视为当前已确定的常数，而通过 $\bar{p}_e$ 对路由器求导．由 Softmax 雅可比，

$$\frac{\partial \mathcal{L}_{\mathrm{bal}}}{\partial g_{tj}} = \frac{E_r}{N} \sum_{e} f_e p_{te} (\mathbf{1}[e = j] - p_{tj}) \tag{36.16}$$

$$= \frac{E_r}{N} p_{tj} \left(f_j - \sum_{e} f_e p_{te} \right). \tag{36.17}$$

因此它可以为未被选择的分数提供连续训练信号，但没有对离散计数进行通常意义下的求导．它不是对实际最大执行时间的精确梯度，也不能保证每一步更新后负载立刻均匀．

当 $f_e = \bar{p}_e = \dfrac{1}{E_r}$ 时，该辅助项为 1，形成一个便于比较的均匀参考值．但它不是均衡偏差的范数，也不能只根据某个标量接近 1 就认定系统均衡．特别是硬分配与软概率的关系、局部分组及有限样本都会影响这个乘积；应联合观察各专家分布和最大负载．

36.4.3　统计范围及任务权重

如果各设备本地批量大小不同，先计算各自平均概率再无权平均，会改变全局统计．应先合并概率和与有效词元数，再形成所需范围内的均值．另一方面，全局均衡可能掩盖单台设备上的局部热点；设计目标必须说明是专家层级、设备层级还是通信组层级．

均衡压力过大时，路由可能优先满足平均分配而牺牲任务有效性；压力过小时，少数专家可能成为瓶颈并获得不成比例的更新．除了辅助损失，也可以通过容量分配、专家复制或路由偏置控制负载，但这些机制改变的对象不同：有的改变训练目标，有的改变离散选择，有的仅改变相同专家副本的执行位置．

路由分数的数值范围同样影响稳定性. 稳定 Softmax 通过减去行最大值避免指数溢出，但不能防止分数排序因低精度舍入而变化. 路由计算与专家矩阵乘法可以采用不同精度；是否合适应结合选择差异、梯度和运行成本判断，而不是要求全部运算使用同一种数据类型.

取 $N = d = 4$，$E_r = 3$，每词元选择 $k = 2$ 个专家，使用选中集合内重新归一化. 令输入 $X = I_4$，路由器无偏置，并设

$$W_r = \begin{bmatrix} \log 0.7 & \log 0.2 & \log 0.1 \\ \log 0.6 & \log 0.3 & \log 0.1 \\ \log 0.55 & \log 0.35 & \log 0.1 \\ \log 0.1 & \log 0.2 & \log 0.7 \end{bmatrix}. \tag{36.18}$$

因为每行概率之和为一，所以 $G = XW_r = W_r$，Softmax 后恰好得到矩阵中的对数自变量. 为使全部专家结果可手算，本例用三个真实线性映射 $F_1(x) = x$、$F_2(x) = 2x$、$F_3(x) = 3x$；这些映射仅用于构造算例，不声称来自训练或具备专家分工.

词元	全专家概率	选中集合	选中后权重
1	$(0.7, 0.2, 0.1)$	$\{1, 2\}$	$(\frac{7}{9}, \frac{2}{9})$
2	$(0.6, 0.3, 0.1)$	$\{1, 2\}$	$(\frac{2}{3}, \frac{1}{3})$
3	$(0.55, 0.35, 0.1)$	$\{1, 2\}$	$(\frac{11}{18}, \frac{7}{18})$
4	$(0.1, 0.2, 0.7)$	$\{2, 3\}$	$(\frac{2}{9}, \frac{7}{9})$

最后一行按专家编号列权重，而不是按分数从高到低排列. 保存路由记录时必须同时保存编号与权重，不能假定两种排序始终一致.

没有容量拒绝时，四个输出分别为

$$y_1 = (\frac{7}{9})x_1 + (\frac{2}{9})(2x_1) = (\frac{11}{9})x_1, \tag{36.19}$$

$$y_2 = (\frac{2}{3})x_2 + (\frac{1}{3})(2x_2) = (\frac{4}{3})x_2, \tag{36.20}$$

$$y_3 = (\frac{11}{18})x_3 + (\frac{7}{18})(2x_3) = (\frac{25}{18})x_3, \tag{36.21}$$

$$y_4 = (\frac{2}{9})(2x_4) + (\frac{7}{9})(3x_4) = (\frac{25}{9})x_4. \tag{36.22}$$

由于 $X = I_4$，Y 就是以这些系数为对角元素的矩阵. 路由权重和为一，但输出系数可以大于一，因为专家输出本身并不被限制在单位范围.

专家申请数为 $(n_1, n_2, n_3) = (3, 4, 1)$，总计 8 条路径. 取容量因子 $\gamma = 1$，由式 (36.13) 得到每专家容量 $C = 3$. 虽然总槽位数为 9，大于申请总数 8，专家 2 仍会溢出.

设按词元顺序接收，专家 2 接收词元 1、2、3，拒绝词元 4．若拒绝路径直接置零且不重新归一化，词元 4 输出成为

$$y_4^{\mathrm{drop}} = (\frac{7}{9})(3x_4) = (\frac{7}{3})x_4. \tag{36.23}$$

若改为对剩余接收路径重新归一化，唯一专家 3 的权重变为一，输出则为 $3x_4$．两个结果均不同于原来的 $(\frac{25}{9})x_4$，而且彼此也不相等．溢出策略因此是明确的函数差异，不能只作为不影响结果的内存优化．

本例拒绝 1 条路径，路径拒绝率为 $\frac{1}{8}$；有 1 个词元失去部分路径，词元受影响比例为 $\frac{1}{4}$；没有词元失去全部路径，完全跳过专家层的比例为零．这三个数字分别回答不同问题．

截断前的归一化分配比例为

$$f = (\frac{3}{8}, \frac{1}{2}, \frac{1}{8}), \qquad \bar{p} = (0.4875, 0.2625, 0.25). \tag{36.24}$$

代入式 (36.15) 得到

$$\mathcal{L}_{\mathrm{bal}} = 3\left[(\frac{3}{8})(0.4875) + (\frac{1}{2})(0.2625) + (\frac{1}{8})(0.25)\right] = 1.0359375. \tag{36.25}$$

专家 2 的申请次数最多，却不是平均概率最高的专家，因为它在多个词元上作为第二选择进入集合．这解释了为什么仅检查平均门控概率不足以诊断实际执行负载．

最大申请负载相对平均负载为 $\frac{4}{\frac{8}{3}} = 1.5$．若只观察截断后的接收数 $(3, 3, 1)$，这个热点会显得减弱，但那是拒绝计算造成的结果．质量分析和性能分析都必须保留截断前后的统计．

算法 36.1　带显式容量规则的稀疏专家前向

输入： $X \in \mathbb{R}^{N\times d}$，路由器 W_r, b_r，E_r 个专家，选择数 k，容量 C；采用选中内归一化、按词元顺序接收、拒绝路径置零且不二次归一化．
输出： $Y \in \mathbb{R}^{N\times d}$，申请数、接收数、拒绝数和路由记录．**状态：** 每专家槽位计数 $c_e = 0$，输出 $Y = 0$，以及接收记录 (t, e, c, a_{te})．

1. 若 $N = 0$，返回空输出和零计数；否则检查 $1 \leqslant k \leqslant E_r$、$C \geqslant 1$，计算路由分数及全专家概率．非有限分数作为输入失败处理，不进入排序．

2. 对词元 $t = 1, \ldots, N$，按分数降序选取不同的 k 个专家；并列时按专家编号升序．计算选中集合内权重，记录所有申请．

3. 对该词元每条选中路径，若 $c_e < C$，将 x_t 写入专家 e 的下一个槽位，并记录原词元编号与权重；增加 c_e．否则记录拒绝，不产生专家输入．

4. 每个专家仅对其接收的 c_e 个有效槽位计算 F_e. 专家位于远端时，按同一槽位记录分发输入并回收输出.

5. 对每条接收记录执行 $Y_{t,:} \leftarrow Y_{t,:} + a_{te}(H_e)_{c,:}$. 全部接收记录合并完毕即停止.

不变量：每个槽位至多对应一个输入；每条接收记录仅合并一次；每专家接收数不超过容量；申请数等于接收数加拒绝数. 若无拒绝，每词元合并权重和为一；有拒绝时本算法不再要求这一行和为一.
路由线性投影计算量为 $O(NdE_r)$；直接完整排序的上界为 $O(NE_r \log E_r)$，选择算法可避免完整排序. 路由记录为 $O(Nk)$，专家计算按实际接收数计. 分发缓冲、网络代价和各专家矩阵大小需另行计入.

算法 36.1 完整定义了一种容量受限函数. 修改排序、二次归一化或拒绝规则时，应先修改这个函数定义，再讨论底层实现是否保持相同结果. 它并不要求高性能运行时逐词元串行执行，批量排序和索引操作可以在保持接收语义的前提下并行化.

36.5　总参数、激活参数及执行成本

36.5.1　参数容量及词元计算

设非路由部分参数为 P_0，每个路由专家有 P_e 个参数，路由器有 P_r 个参数，另有始终执行的共享专家参数 P_s. 在各路由专家同尺寸、没有拒绝的单层计数下，

$$P_{\text{total}} = P_0 + P_r + P_s + E_r P_e, \qquad P_{\text{active}} = P_0 + P_r + P_s + kP_e. \tag{36.26}$$

对于多层模型应逐层累加，不能假定每层都是同样的 MoE. 每词元激活比例 $\dfrac{k}{E_r}$ 仅对应可路由专家部分，不等于整个模型的计算比例.

门控前馈专家忽略偏置时约有 $P_e = 3dd_f$ 个参数. 若取 $d = 1024$、$d_f = 4096$、$E_r = 8$、$k = 2$，则单专家参数为 12 582 912，全部路由专家为 100 663 296，每词元激活的路由专家参数为 25 165 824. 这些是给定矩阵形状的算术计数，不是某个发布模型的测量结果.

以每权重两字节存储，全部路由专家权重为 201 326 592 字节，即 192 MiB；每词元只执行其中两组专家，不能据此把常驻权重预算改成 48 MiB. 实际内存还包括路由器、共享层、缓存、激活和工作空间.

激活参数不等于存储容量

稀疏性减少每词元经过的专家路径；总权重与优化器状态仍由完整参数集合及其放置方式决定. 专家分片或卸载可以降低单设备常驻量，但需计入通信、加载与峰值缓冲，不能用激活参数直接替代权重预算.

36.5.2　批次专家覆盖率

单个词元选择少量专家，不表示整个批次只读取少量权重．若多个词元选择不同专家，批次专家集合是 $\bigcup_t S_t$，其大小可以接近 E_r．增大批量可能改善每个专家的矩阵尺寸，也可能增加本步触及的权重范围，两种效应需要同时分析．

在乘法与加法分别计一次浮点操作、专家主要由矩阵乘法组成的近似下，执行 N 个词元的专家前向成本约为 $2NkP_e$．共享部分、路由排序、激活函数、通信与填充不包含在这个式子中；训练的反向成本还取决于保存和重计算的中间状态．因此不能将稠密模型的 $6PD$ 估算原样代入总 MoE 参数而忽略稀疏性，也不能只代入激活参数后宣布得到了精确时间．

36.5.3　负载不均及小矩阵效率

若每个专家处理一个局部矩阵乘法，理想均衡时每专家批量约为 $\frac{Nk}{E_r}$．专家数增加而总词元数不变，会减小这一批量．矩阵变小可能降低设备利用率，路由和内核启动开销占比则升高．更多专家不是无代价地增加容量．

各设备必须等待相关专家结果返回，执行尾部受最慢专家或最慢设备影响．即使平均专家耗时很低，单个热点也可能决定整个层的完成时间．应关注最大负载、最大接收延迟和跨设备流量，而不只是所有专家平均的利用率．

36.6　共享专家、细粒度专家及软混合

36.6.1　共享路径及细粒度组合

共享专家（Shared Expert）始终处理每个词元，路由专家则条件执行．一种输出形式为

$$y_t = F_{\text{shared}}(x_t) + \sum_{e \in S_t} a_{te} F_e(x_t). \tag{36.27}$$

共享项没有被路由权重缩小，其参数与算术开销都应每次计入．让公共变换与条件变换并存，是结构选择；不能仅凭该公式证明某条路径已经学到公共知识．

细粒度专家（Fine-Grained Expert）将较宽专家改为更多较窄专家，同时增加被选专家数．例如将中间宽度除以 m、专家数乘以 m、选择数也乘以 m，在忽略路由和其他开销时，专家总参数与每词元专家矩阵计算可以近似保持不变．这增加了组合自由度，却使路由分数、分组和通信的组织更加复杂．DeepSeekMoE 将细粒度划分与共享专家作为两项结构设计 [197]；具体型号仍应按精确配置和权重核对．

专家划分不是将一个已训练稠密矩阵任意切块后就自动得到等价 MoE．激活函数、门控组合以及条件选择会改变函数，需要相应训练或经过论证的转换过程．

36.6.2 软专家混合改变专家输入

软专家混合（Soft Mixture of Experts，Soft MoE）并不只是给所有专家保留一个很小的概率．其一种构造先把词元连续混合成有限个专家槽位，再由专家处理槽位，最后将槽位输出混合回词元 [198].

设总槽位数 $S = E_r p$，每专家有 p 个槽位，槽位打分矩阵为 $Z = X\Phi \in \mathbb{R}^{N\times S}$. 分发权重对每个槽位沿词元轴归一化，合并权重对每个词元沿槽位轴归一化：

$$D_{ts} = \frac{e^{Z_{ts}}}{\sum_{u=1}^{N} e^{Z_{us}}}, \qquad C_{ts} = \frac{e^{Z_{ts}}}{\sum_{v=1}^{S} e^{Z_{tv}}}. \tag{36.28}$$

于是槽位输入 $\widetilde{X} = D^{\mathsf{T}}X \in \mathbb{R}^{S\times d}$，各槽位按归属专家计算 $\widetilde{Y}_s = F_{e(s)}(\widetilde{X}_s)$，最后 $Y = C\widetilde{Y}$．这里的 D 是连续混合权重，区别于前文硬选择矩阵 D_e.

每个专家处理的是若干词元的加权组合，不是对每个原始词元都独立执行一次专家．因此它不同于前文的稠密混合．两次归一化也不能交换：分发列和为一，合并行和为一，各自保证不同的混合对象.

这类全局词元混合若直接用于自回归解码，会把未来位置混入早期位置的专家输入．它不能不经修改就替换因果语言模型中的逐位置 MoE；需要另行定义因果槽位构造并分析代价．论文中的结构与实验条件不能自动外推为任意解码器的兼容性证明.

36.6.3 路由方式的约束差异

方式	专家处理对象	离散选择	主要约束
稠密混合	每个词元	无需截断	每词元执行全部专家
词元选择的稀疏混合	选中词元	每词元 top-k	专家负载不固定
专家选择的稀疏混合	专家挑选的词元	每专家挑选输入	每词元路径数可变
软专家混合	连续混合的槽位	不必使用硬 top-k	槽位数与混合可见性

专家选择表示按专家分数列挑选有限词元，能够直接约束每专家的接收量，但一个词元可能得到多个专家，也可能没有专家处理．若需要固定每词元覆盖，还要增加额外约束．比较方法时，应分别说明固定的是词元路径数、专家容量还是槽位数量，而不能只用“稀疏”一词代替这些不同定义.

36.7 专家并行的通信及训练推理差异

36.7.1 参数放置及词元移动

专家并行（Expert Parallelism，EP）把不同专家参数放在不同设备．路由完成后，词元必须送到相应设备；专家输出再送回原来的词元归属设备．**全互换通信**（All-to-All）可以表达这种多发送端到多接收端的数据交换，但并不意味着每对设备每步都拥有相同流量.

设远端执行的接收路径比例为 ρ，每激活元素占 s_a 字节，输入输出宽度均为 d. 忽略元数据和协议开销，前向分发与返回的通信量级为

$$V_{\mathrm{comm}} \approx 2\rho N k d s_a. \tag{36.29}$$

该式按每条远端路径的单向有效载荷计数，并将发送输入与返回输出各计一次；不同统计工具若同时累计发送和接收端会得到不同口径. 张量并行、专家复制、压缩传输与容量拒绝都会改变实际数值.

理想传输时间至少受有效带宽与启动延迟限制. 若共有 m 轮不可忽略的启动，延迟为 ℓ、有效带宽为 b_{eff}，可以用

$$t_{\mathrm{comm}} \gtrsim m\ell + \frac{V_{\mathrm{comm}}}{b_{\mathrm{eff}}} \tag{36.30}$$

理解瓶颈；拥塞、同步和拓扑争用还会增加时间. 计算与通信只有在依赖允许时才能重叠，不能把所有理论算力与所有网络带宽同时按满负荷利用.

36.7.2　专家并行及数据并行

数据并行复制同一模型参数并处理不同数据，专家并行则让一次前向调用不同设备上的不同专家参数. 专家参数只在拥有同一专家副本的数据并行组内同步梯度，路由器与共享参数又可能采用不同复制范围. 组定义错误会造成不同参数被错误求和，或应保持一致的副本没有同步.

同一设备可容纳多个专家，单个专家也可进一步进行张量切分. 专家编号、物理设备编号和张量分片编号不是一个索引. 模型配置记录逻辑专家结构，运行时放置表记录执行位置，检查点必须能从分片还原这些关系.

36.7.3　训练及推理张量形状

训练往往同时处理较多词元，专家批次相对较大，但需保存反向状态并执行反向通信. 各专家梯度还可能十分稀疏或不均匀，优化器状态则不能仅按当前激活量估计. 激活重计算需要重现路由选择、随机噪声与容量接收结果，否则反向不再对应原前向路径.

预填充可以并行处理一段输入，而逐词元解码每个请求每步通常只新增一个位置. 并发较小时，各专家收到的矩阵可能很小，路由、权重读取和通信启动的比例增大. 增加请求合批可能提高专家利用率，却引入等待时间及更复杂的容量竞争.

训练中接受的路径拒绝比例不应直接移植到服务端. 一个被拒绝的关键词元可能改变后续生成，且输出依赖批次组成会使复现和回归更困难. 推理可以采用变长无拒绝执行、扩大容量或其他明确方案，但每项变化都需要评估其函数、质量和延迟影响.

36.8 可执行结构还原

模型名称中出现 MoE 或激活参数数字，只能提供结构线索. 需要按每层核对路由专家数、共享专家数、每词元选择数、中间宽度、门控归一化、路由激活、容量策略及哪些层使用 MoE. 再将这些配置与路由矩阵、专家参数张量和前向定义相互对应.

例如路由投影在本章约定中为 $d \times E_r$；某些库按输出乘输入布局保存为 $E_r \times d$，这属于转置约定，不应仅凭形状方向不同就断言错误. 量化参数可能按组打包，存储元素数也不能直接替代原始参数计数. 对于共享专家，既要确认参数存在，也要确认前向确实每次调用，而不是仅凭张量名称推断.

路由热力图说明某次输入如何被分配，不证明专家专业化；权重可加载说明运行时接受了张量，不证明合并规则与原模型一致；总参数和激活参数相符，也不证明设备拓扑能达到预期吞吐. 完整判断需要把结构事实、数学语义、质量证据与系统测量分开记录.

第三十七章　长上下文建模

更长的输入可以包含完整文档、多轮交互、跨文件代码和分散证据，但把更多词元放入模型，并不等于模型能够可靠使用这些信息．长度增长同时改变位置分布、候选注意力集合、计算规模和缓存容量；即使程序能够完成前向，也可能无法恢复位于中间的关键事实，或者在大量干扰信息下选择错误证据．

本章研究模型怎样从已训练的长度扩展到更长序列，以及怎样判断这种扩展是否有效．第七章《位置表示》已给出位置表示的基础，本章只保留分析扩展所需的相位关系；密集注意力内核的分块计算将在后续推理优化部分展开．这里的主线是表示分布、信息可达性、训练数据与实际长距离能力之间的联系．

37.1　上下文长度的四种含义

37.1.1　训练长度、接口上限及可用能力

上下文窗口（Context Window）指一次模型条件化所能容纳的词元范围，但至少存在四种不同口径．训练长度描述模型更新时看到的序列分布；标称窗口描述制品声明的支持范围；运行长度描述当前设备和并发下能够执行的实际规模；有效长度描述在特定任务与质量要求下能够利用的信息范围．

例如，配置允许输入十万词元，只能说明接口或位置机制没有在更短处拒绝．若模型主要在短序列上训练，远距离交互仍可能分布外；若缓存不足，运行时甚至无法容纳标称窗口；若长文本包含大量相似干扰，程序成功也不保证任务质量．

窗口通常需要同时容纳提示、模板控制词元和生成内容．设最大总长度为 $T_{\max}$，实际提示长度为 T_p，计划生成上限为 T_g，则需满足

$$T_p + T_g \leqslant T_{\max}. \tag{37.1}$$

只报告输入长度而不说明输出预算，可能掩盖生成过程中越界的风险．不同分词器对同一原文产生不同长度，跨模型比较还应保留原始字节数或字符数，而不能默认一个词元包含固定信息量．[1]

[1]产品材料中的“32k”可能表示 32000 或 32768 个词元，应以模型配置与接口约定为准．本章的计算例均写出整数词元数；容量单位 KiB 和 GiB 分别采用 2^{10} 与 2^{30} 字节．

37.1.2 符号及共同假设

符号	含义
T_0, T_1	原始参考窗口与扩展目标窗口的词元数，均大于一.
i, j, Δ	位置索引及相对位置差 $\Delta = j - i$，从零起编号.
d_r, ω_k	RoPE 旋转维数及第 k 个二维子空间的角频率.
s	位置压缩或长度扩展因子，$s \geqslant 1$.
N, d, h_{kv}, d_h	网络层数、隐藏维数、键值头数和单头维数.
w, g	包含当前位置的局部窗口大小；额外全局位置数量.
B, b_{kv}	同时驻留的序列数；单个键值元素占用的字节数.
$Q(T, \rho, \eta)$	给定长度、证据位置和干扰条件下的任务质量.

长度分析的前提

本章默认自回归 Transformer、固定分词器与明确因果掩码；旋转维数为偶数. 位置扩展公式首先在固定查询与键向量下比较相位，整网状态变化另行讨论. 复杂度分析区分预填充与单步解码，容量公式不计临时空间、分配碎片和量化元数据，不能直接作为设备准入上限.

37.1.3 扩展涉及四个不同层次

位置映射决定远处词元怎样进入已学习的表示系统；注意力结构决定哪些位置之间可以直接交换信息；训练数据决定模型是否需要学习跨越这些距离的任务；执行系统决定计算与状态是否能在资源预算内完成. 四者相互影响，却不能互相替代.

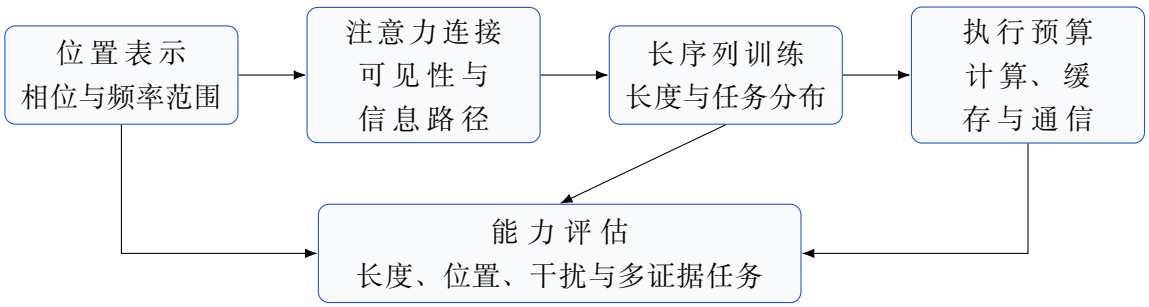


图 37.1: 长上下文扩展的四项设计及共同评价边界. 实现其中一项，不足以证明完整长距离能力.

图 37.1 中的执行优化可以保持数学函数不变，而位置或连接方式的改变通常会改变模型函数. 比较方法时应首先判断变更属于哪一层，再选择相应的训练和证据要求.

37.2　位置外推的分布偏移

37.2.1　位置外推

旋转位置编码（Rotary Position Embedding，RoPE）使第 k 个二维子空间中的查询与键通过相对角度 $\Delta\omega_k$ 影响点积．固定内容向量时，该子空间的贡献可写为

$$a_k \cos(\Delta\omega_k) + b_k \sin(\Delta\omega_k), \tag{37.2}$$

其中 a_k, b_k 由对应查询和键的分量决定．基础推导见第七章《位置表示》．位置扩展改变的是这些角度的分布，而不仅是保存一个更大的位置整数．

若训练窗口为 T_0，训练中通常只出现绝对值不超过 $T_0 - 1$ 的相对距离．直接输入 $T_1 > T_0$ 的序列，将产生未见过的相对距离和多频率联合相位组合．三角函数在所有实数位置均有定义，并不表示学习到的注意力选择函数对这些组合已经可靠．

这里也不能把单个正弦分量的周期性误读为整体表示每隔一个固定短周期必然重复．多组频率联合使用时，完全重合与近似混淆是不同问题；有限精度、频率尺度及训练分布都会影响可区分性．

37.2.2　长序列还会改变竞争集合

即使某条证据与查询之间的分数不变，增加候选键也会改变归一化分母．设一个相关键分数为 a，其余 $T-1$ 个干扰键分数为零，则相关键获得的权重为

$$p_{\mathrm{rel}} = \frac{e^a}{e^a + T - 1}. \tag{37.3}$$

当 $a = \log 100$ 时，$T = 101$ 对应 $p_{\mathrm{rel}} = 0.5$；$T = 901$ 对应 $p_{\mathrm{rel}} = 0.1$．这是一项控制变量的数学构造，并非真实模型的注意力测量．

若要求权重至少为 $\alpha \in (0,1)$，整理式 (37.3) 可得

$$a \geqslant \log(T-1) + \log \frac{\alpha}{1-\alpha}. \tag{37.4}$$

候选集合扩大可能要求更强的分数区分能力．实际干扰键的分数不相同，多个相关键也可能共同承载信息，但这个例子足以说明：解决位置外推并没有自动解决长序列中的选择与聚合问题．

37.3　位置插值及相位压缩

37.3.1　线性插值的定义

位置插值（Position Interpolation，PI）将新窗口的位置缩放到较短的参考范围，是扩展 RoPE 模型的一种方法 [199]．为明确离散端点，本章定义

$$s = \frac{T_1 - 1}{T_0 - 1}, \qquad f(i) = \frac{i}{s}, \qquad 0 \leqslant i \leqslant T_1 - 1. \tag{37.5}$$

于是 $f(0)=0$、$f(T_1-1)=T_0-1$. 位置输入可以是实数，旋转并不要求它仍为整数. 某些实现采用 $s=\dfrac{T_1}{T_0}$ 的长度比约定；二者在长窗口下接近，但端点不同，应保持配置和推导一致.

插值后的相对角度为

$$[f(j)-f(i)]\omega_k=\frac{\Delta\omega_k}{s},\tag{37.6}$$

等价于将所有角频率统一除以 s. 这保留位置顺序，却缩小相邻位置的角度差. 输入长度没有缩短：模型仍然处理 T_1 个词元，词元数、注意力候选数和缓存长度均不会因位置坐标被压缩而自动减少.

37.3.2　位置插值误差

设原位置范围为 0 到 4096，即 $T_0=4097$；目标位置范围为 0 到 32768，即 $T_1=32769$. 式 (37.5) 给出 $s=8$，映射为

$$0\mapsto 0,\quad 4096\mapsto 512,\quad 16384\mapsto 2048,\quad 32768\mapsto 4096.\tag{37.7}$$

取某二维频带 $\omega=\dfrac{1}{1024}$，两个位置的距离为 $\Delta=8192$. 直接外推时相对角度为八弧度；插值后角度为一弧度. 若固定查询与键使式 (37.2) 中的 $a=1,b=0$，这项点积贡献由 $\cos 8\approx -0.1455$ 变为 $\cos 1\approx 0.5403$.

原范围内的距离同样改变. 例如 $\Delta=1024$ 对应角度从一变为 $\dfrac{1}{8}$. 因此位置插值并非只修补超出训练窗口的部分，而会同时改动短距离表示. 扩展后需要关注原有短文本任务是否退化，不能只评估新增长度.

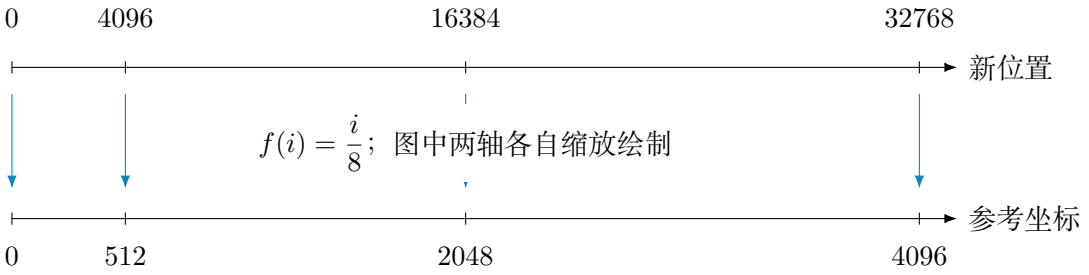


图 37.2: 位置插值的坐标对应. 图中箭头表示坐标映射，不表示删除或合并词元.

图 37.2 刻意保留全部对应关系. 与分词压缩不同，位置插值不改变输入字符串和词元序列，只改变模型对位置的使用方式.

37.3.3　相位扰动的局部上界

对二维旋转 $R(\phi)$，两个角度之间的算子范数差为

$$\|R(\phi')-R(\phi)\|_2=2\left|\sin\frac{\phi'-\phi}{2}\right|\leqslant|\phi'-\phi|.\tag{37.8}$$

固定 q,k 后，柯西—施瓦茨不等式给出

$$|q^{\mathsf{T}}[R(\phi')-R(\phi)]k| \leqslant \|q\|_2\|k\|_2\,|\phi'-\phi|. \tag{37.9}$$

对统一插值，$|\phi'-\phi|=|\Delta\omega_k|(1-\frac{1}{s})$．较大距离或较高频率可能对应较大的局部扰动．这个上界说明哪些量影响变化，但它不比较“哪种扩展一定更好”，也不约束整个深层网络的最终误差，因为后续查询和键本身也会因早先层的变化而改变．

37.4　频率调整及短长距离折中

37.4.1　统一缩放及频带差异

标准几何频率可写为

$$\omega_k = b^{-\frac{2k}{d_r}}, \qquad k=0,\dots,\frac{d_r}{2}-1, \tag{37.10}$$

$b>1$ 为频率基数．高频分量在相邻位置间变化较快，低频分量在长距离上变化较慢．统一插值把全部频率同时降低，扩大有效周期，也使局部位置差异变小．

一种不同选择是增大频率基数．设 $d_r>2$，取

$$b' = b\,s^{\frac{d_r}{d_r-2}}. \tag{37.11}$$

则频率比为

$$\frac{\omega'_k}{\omega_k} = s^{-\frac{2k}{d_r-2}}. \tag{37.12}$$

最高频分量 $k=0$ 不变，最低频分量 $k=\frac{d_r}{2}-1$ 缩小为 $\frac{1}{s}$，中间频带平滑变化．这是对给定基数规则的代数推导；具体模型是否使用这一形式，仍需以其位置配置为准，不能将所有所谓“频率缩放”都写成同一个公式．

例如 $d_r=8, s=8$，四个频率的比例依次为 $1,\frac{1}{2},\frac{1}{4},\frac{1}{8}$．它与统一插值的四个比例均为 $\frac{1}{8}$ 不同：前者更多保留快速变化的频带，后者统一压缩所有距离．两种方法都改变了多频率联合编码，均需检查与预训练状态的适配．

37.4.2　按频带混合的思想

给定参考位置跨度 T_0-1，频带 k 在该范围内经历的旋转周数为

$$n_k = \frac{(T_0-1)\omega_k}{2\pi}. \tag{37.13}$$

若 n_k 较大，训练中该频带已经经历多次旋转；若 n_k 很小，其长周期部分尚未充分覆盖．这为区别处理频段提供一种尺度指标．可用一般形式表达混合调整：

$$\omega'_k = \lambda_k\omega_k + (1-\lambda_k)\frac{\omega_k}{s}, \qquad 0\leqslant\lambda_k\leqslant 1. \tag{37.14}$$

其中 $\lambda_k = 1$ 保留原频率，$\lambda_k = 0$ 使用完整插值，中间值平滑过渡. 阈值与过渡函数是额外设计，不由本式自动确定.

YaRN 将分频调整与注意力尺度调整结合，用于扩展预训练模型的窗口 [200]. 本章采用式 (37.14) 说明其所属的方法思想，不把某个实现的默认阈值写成普遍常数，也不以某组训练预算证明任意模型无需适配就可扩展.

37.4.3 注意力温度及长度

设分数为 z_j，带温度的归一化为

$$a_j(\tau) = \frac{e^{\frac{z_j}{\tau}}}{\sum_r e^{\frac{z_r}{\tau}}}, \qquad \tau > 0. \tag{37.15}$$

较小温度通常使分布更集中，较大温度使其更平缓. 位置调整改变分数分布，长度又改变竞争集合，因而一些方法同时调整注意力尺度. 但是缩小温度也可能放大错误匹配，并不等于自动解决式 (37.3) 中的证据稀释.

若同时将查询和键乘以常数 c，点积会乘以 c^2；若只改变一侧，则只乘以 c. 某些制品把尺度放在旋转参数或注意力模块中，数学上必须按实际乘法位置核对. 重复应用尺度可能导致分布过尖，而遗漏尺度可能使与训练时的行为不一致.

37.4.4 动态缩放及缓存一致性

键值缓存（Key–Value Cache，KV Cache）保存先前位置的中间状态. 若位置缩放随当前序列长度动态变化，已经缓存的键可能采用旧频率，而新查询采用新频率. 此时两者的点积不再对应预期的共同位置规则.

即使能够反向旋转某层缓存键再使用新频率，也未必恢复整个模型重新计算的结果，因为该层键来自早先层表示，后者可能已经在旧频率下形成. 稳妥的协议应明确缩放何时确定、一个请求内是否固定、跨阈值时如何处理以及哪些缓存可以复用. 修改位置规则属于模型函数变更，应进入制品与缓存身份，而不是仅作为服务端无关紧要的长度参数.

位置扩展的必要边界

把位置坐标映射回旧范围，不会减少词元数，也不会自动教授跨文档推理. 频率、注意力尺度、训练数据和缓存协议必须共同一致；短任务保持与长任务改进需要分别给出证据.

37.5　滑动窗口、稀疏连接及可达性

37.5.1　因果滑动窗口的计算规模

滑动窗口注意力（Sliding-Window Attention，SWA）限制位置 i 只读取最近 w 个位置，窗口包含当前位置：

$$A_{ij} = \mathbf{1}[\max(0, i-w+1) \leqslant j \leqslant i]. \tag{37.16}$$

当 $T \geqslant w$ 时，可见位置对总数为

$$\sum_{i=0}^{T-1} \min(i+1, w) = \frac{w(w+1)}{2} + (T-w)w = Tw - \frac{w(w-1)}{2}. \tag{37.17}$$

因此读取计算随 $O(Twd)$ 增长，而不是完整密集注意力的 $O(T^2 d)$．但查询、键值及 FFN 的投影仍需处理全部词元；而且若实现先构造完整 $T \times T$ 矩阵再屏蔽，便没有兑现稀疏执行的全部资源收益．

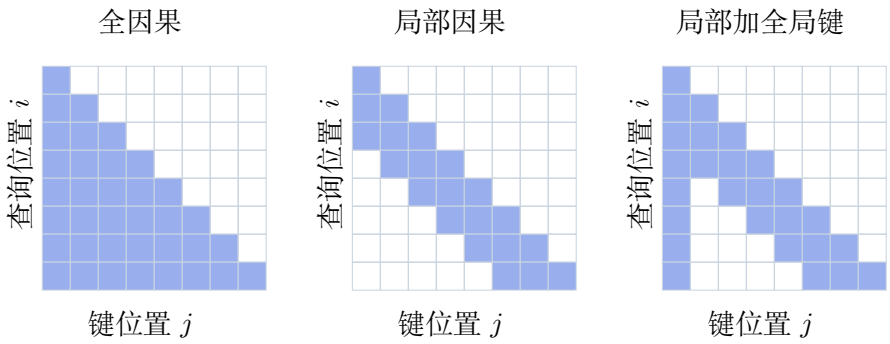


图 37.3: 单层直接可见范围：八个位置、局部窗口三．着色单元表示允许读取；第三幅额外允许读取全局键位置 0，全部连接仍满足 $j \leqslant i$．该开头位置不能汇聚未来信息．

37.5.2　跨层感受范围

窗口限制的是单层直接读取范围．忽略其他跨位置机制，若第零层位置 i 只含当前位置信息，经过 N 层窗口为 w 的因果层后，其最早可达位置为

$$\max[0, i - N(w-1)], \tag{37.18}$$

所以最多覆盖 $1 + N(w-1)$ 个连续位置．证明由归纳得到：每增加一层，当前层可读取前一层最早 $i-w+1$ 的状态，该状态又向前覆盖已有感受范围．

取 $w = 4, N = 3$，最终位置最多可达十个位置，而不是四个，也不是任意长前缀．远处信息必须通过中间表示逐层传递；存在计算图路径是信息影响输出的必要条件，不保证关键事实在有限维表示中仍被保存并正确使用．

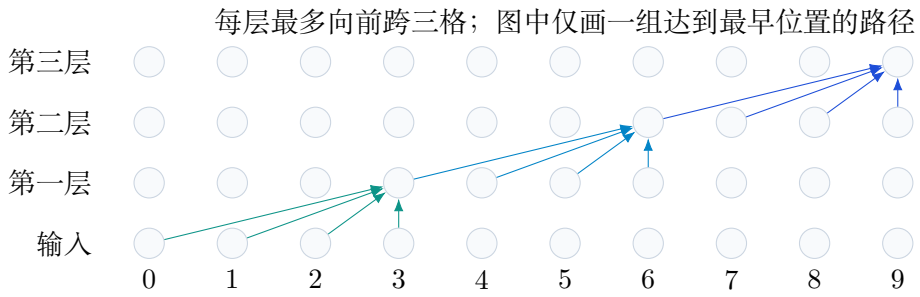


图 37.4: 局部窗口的跨层信息可达性. 三层、窗口四的位置 9 能够间接接收位置 0 的信息，但需要多次表示转换.

图 37.4 中箭头沿信息传播方向绘制. 若增加层数，图论上的可达范围扩大；这并不意味着可以通过无限增加深度而不付出优化、延迟和信息压缩代价.

37.5.3　全局位置及混合层

局部窗口可以与少量全局连接结合. Longformer 等结构研究了局部与任务相关全局注意力的组合 [201]. 在双向编码中，全局位置可先汇聚各处信息，下一层再将聚合结果传给其他位置，从而缩短信息路径. 若全局位置数为固定 g，连接规模可按 $O[T(w+g)]$ 估计.

把这种结构用于因果解码时必须重新检查方向. 序列开头的一个固定全局词元不能读取未来内容，因而不能自动成为后续全文的动态摘要. 若要使某个摘要位置收集之前的片段，必须把它放在片段之后，或者定义额外记忆更新机制；两者都属于模型结构与训练协议.

另一种混合方式在部分层使用全局密集注意力，其余层使用局部窗口. 设 N_g 个全局层、N_l 个局部层，则注意力部分规模约为

$$O(N_g T^2 d + N_l T w d). \tag{37.19}$$

只要 N_g 不为零，超长长度下仍存在二次项. 不能因为大多数层采用窗口，就宣称整网对长度严格线性.

37.5.4　稀疏连接结构

稀疏注意力（Sparse Attention）还可以采用步长连接、分块连接、内容选择或不同层的不同模式. 相同非零位置数不代表相同信息路径：只连接近邻可能难以跨越远处，少量跨块边可以缩短路径，但也可能漏掉所需内容.

若使用内容相关选择，筛选本身具有成本，而且漏选会改变模型函数. 若使用固定随机边，可达性结论也依赖随机图的条件，不能由一个稀疏率数字推断所有样本都能完成多证据推理. 选择连接方式时，应同时说明可见性、路由或选边规则、实际稀疏内核以及需要重训或适配的范围.

37.6　长序列训练的长度及任务分布

37.6.1　序列长度及依赖距离

将许多无关短文档拼成一个长序列，可以增加位置索引覆盖，却未必提供跨长距离的学习信号．如果每个目标的答案都在附近，模型仍可依靠局部相关性降低损失．有效训练样本需要包含目标任务真正依赖的远处信息，例如跨章节指代、跨文件定义与调用、多证据合并和带时序约束的长记录．

文档拼接还需要明确边界：独立文档之间是否允许注意力跨越，位置是否重置，重置后是否有文档标识，损失是否覆盖分隔词元．没有掩码限制的重置位置会让不同文档出现重复位置，同时仍可互相读取，形成与独立样本不同的目标．不能把打包方式当作与模型训练无关的存储细节．

37.6.2　按序列采样及按词元配比

设训练按序列以概率 q_ℓ 选择长度桶 ℓ，其平均有效长度为 T_ℓ．该桶在实际训练词元中的期望占比为

$$\pi_\ell = \frac{q_\ell T_\ell}{\sum_r q_r T_r}. \tag{37.20}$$

若短桶长度为 4096，长桶长度为 32768，两桶各抽一半样本，则长桶词元占比为 $\dfrac{32768}{4096+32768} = \dfrac{8}{9}$．因此“长短样本各半”并不是“长短训练词元各半”．比较扩展训练预算时，应明确采样单位、实际有效词元数和重复曝光情况．

长度课程可以逐步提高长样本比例或最大长度，以控制资源峰值和分布变化；也可从开始就混合不同长度．它们是训练策略，不具有单一必然最优顺序．短序列保持集用于观察原能力变化，长序列开发集用于选择扩展设置，最终独立测试集不应参与这些选择．

37.6.3　全局词元预算及监督密度

固定每步词元预算 D_{step} 时，序列加长通常要求减少微批大小．样本数减少会改变一个更新步包含的独立文档和任务数量．梯度累积可以恢复目标全局词元数，但不能自动复制原来的样本多样性和数值顺序．

长文问答中大量输入是条件而不是监督目标．设输入词元数为 T、监督目标数为 R，监督密度为 $\dfrac{R}{T}$．比较训练效率时，长输入和少量回答可能形成很低的监督密度；仅报告处理了多少输入词元不足以说明训练信号充足．可以混合语言建模与长距离任务，但混合权重、标签位置和泄漏边界都应明确．

算法 37.1　长上下文适配训练

输入：基础模型、目标窗口、候选位置规则、分组隔离的长短训练集、开发集与计算预算. **输出：**经过适配的模型制品及其长度配置. **状态：**当前长度课程、优化器、累计有效词元和开发评价记录.

1. 固定分词器、模板、位置频率、注意力结构及短任务基线；保存基础模型身份.
2. 按规定的长度桶和任务配比构造批次，区分序列采样率与实际词元占比；保留文档边界和证据来源.
3. 构造位置、可见性与监督集合，执行参数更新；累计真实有效输入和监督目标数量.
4. 按预定时点在长短开发集上记录质量与资源消耗，依据课程规则调整长度或采样比例；不使用测试集选择配置.
5. 达到预算、预定停止准则或发现不可接受的短任务退化时停止，按事先规定的开发准则选择制品.

不变量：同一制品中的频率与缓存解释一致；训练预算按实际处理量记录；长样本包含所声称目标需要的远距离依赖.

37.7　长度增长的计算、缓存及系统约束

37.7.1　预填充及单步解码

预填充（Prefill）一次处理已有上下文，**增量解码**（Decode）基于历史状态计算新位置. 固定模型宽度和层数时，密集预填充包含 $O(Td^2 + T^2d)$ 规模的投影和位置交互；采用键值缓存后，单步解码约为 $O(d^2 + Td)$，仍需读取历史键值.

从长度 4096 增长到 32768，长度扩大八倍，密集注意力的二次部分扩大 64 倍，线性投影部分扩大八倍. 缓存解码中与历史长度有关的读取部分扩大八倍. 整网时间不会严格等于这些比例，因为硬件利用率、批处理、内核和通信也会变化，但数学规模变化不能被忽略.

精确分块注意力可以减少中间矩阵的显存读写，仍计算同一密集函数；稀疏窗口改变可见位置集合，通常也改变模型函数. 二者可以组合使用，却不是同一种长序列技术. 训练中的重计算、序列并行及激活分片分别用额外计算或通信换取单设备容量，不保证减少集群总工作量.

37.7.2　KV 缓存容量

对等长 B 条序列，每层保留全部历史键值时，裸缓存容量为

$$M_{KV} = 2NBTh_{kv}d_h b_{kv}. \tag{37.21}$$

因子二对应键和值；b_{kv} 的单位为字节，不是位. 设 $N = 32, h_{kv} = 8, d_h = 128, b_{kv} = 2$，则单条序列每增加一个词元，缓存增长

$$2 \times 32 \times 8 \times 128 \times 2 = 131072 \text{ 字节} = 128\,\text{KiB}. \tag{37.22}$$

长度 32768 对应 4 GiB，长度 131072 对应 16 GiB；若后者同时驻留四条序列，裸 KV 就达到 64 GiB. 这些数值不含权重、临时工作区、页表和碎片，不能据此认定一张同容量设备能够执行该负载.

若所有层严格使用窗口 $w = 4096$，并且后续计算从不再读取更早状态，则每条序列理论上只需保留窗口内 KV，约为 0.5 GiB. 这一缩减成立的前提是窗口模型语义，而不是对全注意力模型任意删除旧缓存.

混合结构应逐层求和：

$$M_{KV} = 2Bh_{kv}d_h b_{kv}[N_g T + N_l \min(T, w)]. \tag{37.23}$$

同样 32 层中，八层全局、24 层窗口，在 $T = 131072, w = 4096, B = 1$ 时，裸 KV 为 $4 + 0.375 = 4.375$ GiB. 全局层仍随总长度增长，所以缓存并未完全与上下文长度脱钩.

37.7.3　缓存卸载成本

将 KV 卸载到主机或其他存储，可以降低设备驻留容量，却增加数据移动. 迁移数据量为 M、有效传输带宽为 u 时，单次传输时间下界为 $\dfrac{M}{u}$，实际还需增加同步、排队和序列化. 若解码每一步都要反复读取大批卸载状态，带宽可能成为主要瓶颈.

分组查询结构减少键值头数，缓存量随之降低，但查询输出和注意力评分工作不必按相同比例缩减. 张量并行若复制部分键值头，每设备缓存也不能一律按并行度平均相除. 缓存分页改善分配与复用，仍不会删除有效历史元素本身.

标称长窗口落地时，需要同时规定输入输出长度分布、并发、缓存数据类型和层间注意力配置. 对于不同长度请求，应保留**首词元延迟**（Time to First Token，TTFT）和**每输出词元时间**（Time per Output Token，TPOT）的统计口径；长输入的预填充等待和长历史的逐步读取是不同成本.

37.8 上下文压缩、检索及长窗口的分工

37.8.1 三种减少输入负担的方法

上下文压缩（Context Compression）可以指删除不相关原文、把原文摘要为较短文本，或把历史编码为较小的隐状态. 前两者改变输入内容，第三种改变表示及访问机制. 删除和摘要减少词元数，位置插值则不减少词元数；不能用同一个“压缩率”混称这些方法.

压缩具有信息瓶颈. 若任务可能询问任意细节，而压缩结果的可表达状态少于全部原文，则至少有不同原文映射到同一压缩结果；只凭该结果无法区分它们. 对特定任务，压缩可以保留足够的相关信息，但这种充分性依赖问题范围，不存在普遍保证任意长文所有细节无损压成固定短摘要的方案.

隐状态压缩也不能仅比较浮点元素数. 压缩器训练目标、记忆更新顺序、跨片段位置与解码访问方式共同决定语义. 对仅支持标准 KV 的预训练模型直接换成任意压缩状态，通常不是合法的即插即用操作.

37.8.2 检索范围及窗口容量

检索增强生成（Retrieval-Augmented Generation，RAG）先从外部知识集合选择相关材料，再把它们提供给生成模型. 长窗口则扩大模型一次可联合处理的材料范围. 两者可以组合：检索控制来源、时效、权限和候选集合，长窗口支持更多证据及复杂关系的联合处理.

检索可能遗漏证据，长窗口可能难以从噪声中利用证据. 扩大返回文档数量通常提高覆盖机会，同时增加干扰与成本；减少文档数量降低模型输入，却可能截断多跳问题需要的某一环. 系统设计应把证据召回与证据利用分开评价，而不能把回答错误全部归给模型窗口.

方法	主要改变	主要失效风险
扩大窗口	一次共同条件化的容量	位置分布、长距离利用与计算成本.
检索选择	外部材料的候选集合	漏召回、错误排序、权限或时效错误.
文本摘要	输入内容与长度	事实遗漏、压缩幻觉、细节不可恢复.
隐状态记忆	历史表示及访问接口	压缩瓶颈、训练适配与缓存协议.

37.8.3 跨窗口任务的结构化处理

当材料远超可用窗口，可以按问题拆解、分片读取、记录来源明确的中间结果，再汇总回答. 这是一项多阶段任务策略，而不是模型本身已经具备无限上下文. 中间结果应保留证据定位、未解决问题与冲突关系，避免多轮摘要不断丢弃约束.

多阶段处理还会累积错误，并增加调用、检索和协调成本．若最终问题依赖所有片段之间的全局关系，局部摘要可能不足；若只需少量明确事实，选择性读取可能更有效．架构选择应由任务依赖结构决定，而非固定把全部材料塞入最长窗口．

37.9　长距离能力的分离评估

37.9.1　长度、位置及干扰变量

长文本评价应至少分离总长度 T、证据相对位置 ρ、干扰强度 η、证据数量及任务类型．可将证据起点相对位置定义为

$$\rho = \frac{j_{\mathrm{start}}}{T-1}, \qquad 0 \leqslant \rho \leqslant 1, \tag{37.24}$$

但移动包含多个词元的片段时还需保证片段完整且不越界．为了比较位置作用，应尽可能固定事实内容、问题、总长度和干扰分布；否则把更难的问题恰好放在中间，会混淆因果解释．

中间信息利用退化（Lost in the Middle）研究发现，某些被测模型对相关信息位于开头或结尾的情况表现较好，而对中间位置的表现下降 [202]．这是特定模型、任务和实验条件下的发现，不是所有模型必然遵守的数学定律，更不能用一条预设 U 形曲线代替当前模型测量．

37.9.2　多依据推理

单个隐蔽字符串或密码检索能够检测基本信息回收，但不能充分评价长文理解．更完整的任务还包括多个目标、多个相似键、指代链、数值聚合、跨文档冲突与依赖多条证据的推理．RULER 等评估工作进一步扩展了长上下文任务的检查范围 [203]．

证据存在性也应作为条件：有答案样例检验利用能力，无答案样例检验模型是否在缺少依据时编造．还应设置互相矛盾的来源和具有时间版本的记录，区分找到文本、识别正确版本和完成推理三个能力．随机合成任务便于严格控制变量，真实长文任务检验语言和结构的复杂性，两者各有局限．

37.9.3　定义任务条件下的有效长度

设在固定测试分布上测得质量为 $Q(T,\rho,\eta)$，预先给定最低阈值 $q_{\min}$，并确定待考察的位置与干扰集合 $\mathcal{C}$．在离散测试长度集合 $\mathcal{T}$ 上，可定义一种保守的有效长度：

$$T_{\mathrm{eff}} = \max\left\{ t \in \mathcal{T} : \min_{\substack{T \in \mathcal{T},\ T \leqslant t \\ (\rho,\eta)\in\mathcal{C}}} Q(T,\rho,\eta) \geqslant q_{\min} \right\}. \tag{37.25}$$

该定义要求所有已测试较短长度和规定条件均达标，而不是只挑一个偶然表现较好的最长点. 若集合为空，应报告未达到阈值；有限测试也不能证明未测长度区间全部成立.

式 (37.25) 是一种明确任务条件的评价口径，不是行业统一标准. 改变任务、阈值、位置集合或统计置信要求，得到的有效长度会改变. 实际报告还应给样本量、波动范围和失败类型，避免把一个有较大不确定性的估计写成精确模型常数.

37.9.4 语言建模损失的比较边界

比较长上下文是否有用，可以在相同目标片段上改变可见前缀长度，计算

$$\Delta\mathcal{L} = \mathcal{L}_{\text{short}} - \mathcal{L}_{\text{long}}. \tag{37.26}$$

目标词元、分词器和评分位置应保持一致，否则长序列增加了更容易预测的文本，平均损失下降也可能只是样本组成变化. 正值表明该组目标在更长条件下获得较低损失，不直接证明所有远处信息均被利用.

同理，模型依靠参数记忆答对一个事实，不能证明它从长文本中读取了该事实. 可使用明确置换的实体、唯一合成键或受控反事实材料，使答案必须依赖给定上下文；同时警惕构造任务过于机械而缺少真实难度.

算法 37.2 长度与证据位置的受控评估

输入： 固定模型制品、独立任务模板、长度网格、证据位置网格、干扰等级及评分函数. **输出：** 分条件质量与成本矩阵. **状态：** 样例标识、来源记录、实际词元长度和逐条件结果.

1. 为每项任务固定问题与必要证据，保留正确答案和来源；明确是否存在答案、需要几条证据.
2. 按每个长度与位置组合插入完整证据，再用规定的干扰材料填充；分词后核对实际位置和总长度，避免静默截断.
3. 在相同生成设置下执行模型，记录回答、来源引用、失败、实际资源消耗及停止原因.
4. 依据预先定义的评分规则分别评价证据回收、答案正确性和无依据生成；在相同任务标识间进行配对比较.
5. 遍历预定网格后停止，报告全部条件与统计不确定性；若需要调参，另用开发集，不返回修改测试答案.

不变量： 比较位置时不同时更换任务难度；超时和越界不从结果中无声删除；标称窗口与达标范围分开报告.

37.10　设计取舍及失效定位

长上下文问题应从失败发生的位置诊断. 输入被截断是接口或预算问题；位置扩展后短任务退化提示表示适配问题；只有中间证据失败提示需要进一步分离位置与干扰；单事实任务通过而多证据任务失败，说明信息回收不足以解释全部能力. 相同质量下首词元时间过长或缓存增长失控，则属于执行与容量问题.

有效长上下文是联合约束

可接收的长度、可计算的长度与可可靠利用的长度不同. 一个完整扩展方案必须同时保持位置及缓存一致性，给出注意力的信息路径，在相应数据上适配，并在分离长度、位置和任务难度的评价中达到所需质量.

不存在一个只调大最大长度即可普遍解决上述问题的参数. 位置扩展、窗口稀疏化、检索、压缩和执行优化分别改变不同条件；选择组合时应追踪其具体收益及失去的信息，保留原制品与可回退配置，并把未经评价的新长度作为尚未验证的能力范围.

第三十八章　开源模型体系

前面各章分别建立了注意力、位置表示、专家路由与语言建模目标. 本章把这些机制放回完整模型，研究结构选择怎样影响训练、缓存、输入协议和部署条件. 本章以“开源”为组织线索：所列案例都能取得公开权重，但权重、代码、数据与许可的开放范围差别很大. 模型家族的名称只提供来源线索；真正参与计算的是某个具体版本的参数、配置、分词器与运行代码.

本章选用有公开技术报告或模型卡的 DeepSeek-V2/V3/R1 与 V4.1 Flash、Qwen3、Qwen2.5-VL、早期 GLM 与 GLM-4.5、Llama 4、Mistral 7B、Mixtral、Gemma 4、Olmo 3 与 gpt-oss，并以 Kimi K2、MiniMax-01 补充技术路线. 它们是解释机制的版本案例，不组成当前产品排行榜. 来源查阅于 2026 年 9 月 12 日；新版本应沿用本章的方法重新分析，不能继承同一品牌旧版本的架构或许可结论.

比较模型的基本单位

比较单位是一份可以明确定位的模型制品及其使用协议. 品牌相同不保证网络结构相同；结构相同不保证训练数据和行为相同；权重相同也不保证不同模板和运行后端产生完全一致的结果.

报告案例	首次提交日期	本章所据报告版本
DeepSeek-V2	2024-05-07	2405.04434v5
DeepSeek-V3	2024-12-27	2412.19437v2
DeepSeek-R1	2025-01-22	2501.12948v2
Qwen3	2025-05-14	2505.09388v1
Qwen2.5-VL	2025-02-19	2502.13923v1
GLM	2021-03-18	2103.10360v2
GLM-4.5	2025-08-08	2508.06471v1
Kimi K2	2025-07-28	2507.20534v2
MiniMax-01	2025-01-14	2501.08313v1
Mistral 7B	2023-10-10	2310.06825v1
Mixtral 8x7B	2024-01-08	2401.04088v1
Phi-4	2024-12-12	2412.08905v1
gpt-oss	2025-08-08	2508.10925v1
Olmo 3	2025-12-15	2512.13961v2
Gemma 4	2026-07-02	2607.02770v2

表中日期指技术报告首次提交，不等同于模型权重或产品上线日期．带版本号的条目固定本章核对到的 arXiv 修订．Llama 4 没有单独的技术报告，其结构与许可以官方模型卡的固定提交为准，见本章相应小节．各报告的正式文献条目在对应案例中引用．

38.1　模型分析对象

38.1.1　结构、训练及产品协议

稠密模型（Dense Model）在一次前向中执行相应层的全部前馈参数，**稀疏专家模型**（Sparse Mixture-of-Experts Model，MoE）则按路由选择部分专家．二者属于计算结构．Base、Instruct、Reasoning 等名称主要描述训练阶段或预期行为，不能仅凭后缀反推注意力形式．一个推理蒸馏检查点可以保留学生模型的普通 GQA，而不继承教师的潜在注意力或专家结构．

输入和输出模态又是另一维度．文本检查点、视觉语言检查点和语音模型即使属于同一家族，也需要分别判断编码器、连接器和处理器．外部 API 可以把多个模型和工具包装为一个产品入口，因此产品能力不等于某一个公开权重文件的能力．

38.1.2　开放程度分层

“开源”在日常讨论中常被当成二值属性，工程上却至少包含四个可以分别核对的维度：能否取得权重、能否取得运行与训练代码、能否取得数据与训练配方、以及在什么许可下使用．四者互不蕴含，需要分别取证．

表 38.1: 开放程度的四个可分别取证的维度．权重可下载不蕴含其余三项．

维度	可核对的证据	典型形态
权重开放	参数文件、分片索引与模型卡	多数“开源”模型只到这一层．
代码开放	建模与训练代码的公开仓库及其许可证	权重许可与代码许可可以不同．
数据与配方开放	数据清单、处理脚本、训练日志与超参数	公开范围差异最大，常只覆盖一部分．
许可类型	精确版本的许可证与使用政策文本	宽松开源许可、社区许可与自定义许可并存．

开放权重（Open Weights）只承诺可以取得参数文件．Olmo 3 一类制品同时公开权重、代码、数据与训练过程，使预训练链路可以被外部复现和审计 [204]；gpt-oss、Qwen3 与 Gemma 4 采用宽松开源许可 [205, 206, 207]；Llama 4 使用社区许可并附带可接受使用政策，其条款不符合通常意义上的开源许可定义 [208]；DeepSeek 的部分制品使用自定义模型许可．同一品牌的不同代际也可能改变许可，Gemma 与 Kimi 都出现过这种变化．

开放代码、开放训练数据与托管服务可用同样需要独立证据. 权重下载权限不自动说明所有用途均获许可，许可判断应绑定精确版本的条文和实际使用方式，本章不按品牌给出统一商业授权结论. [1]

38.1.3 统一结构描述

记层数为 L，隐藏宽度为 d，查询头数为 h_q，键值头数为 h_{kv}，每头宽度为 d_h. 专家数为 E，每词元激活数为 k，共享专家数为 E_s. 缓存精度每元素占 b_c 字节，总权重和激活权重参数量分别为 P_{total} 与 P_{active}.

比较维度	应明确的对象	直接影响
网络结构	稠密或专家、注意力与状态更新、归一化和位置机制	算术量、参数驻留与可用内核.
训练目标	预训练、监督微调、偏好或可验证奖励、蒸馏	输出行为、推理预算与质量边界.
交互协议	分词器、消息模板、特殊词元、工具格式	输入含义、停止行为与结构解析.
模态路径	文本、图像、音视频及处理器	输入预算、同步关系与额外计算.
制品身份	来源版本、文件摘要、适配器与量化关系	可恢复性、兼容性与变更追踪.
使用边界	精确版本的许可、用途限制和评估条件	候选准入与实际任务适用性.

同一组字段并不能覆盖所有结构. 例如潜在注意力需要压缩维数，状态空间模型需要状态形状，混合架构需要逐层类型表. 遇到不适用字段应解释原因，不能强行把所有模型还原为普通 MHA.

38.1.4 模型制品比较

为把比较方法落到实物，下面选择三个文本制品. 表中 B 表示十亿参数，参数量采用发布者口径；每参数两字节仅用于计算统一精度下的逻辑权重体积，不是下载文件大小或可运行显存. 对稠密模型不另报稀疏意义的激活参数，因为嵌入查表与全部矩阵乘法的参与口径不同.

表 38.2: 固定版本的结构、目标与资源比较. GB 为 10^9 字节；状态按两字节、单序列、标准键值或潜在缓存计算，未计 MTP 附加状态、分页和工作区.

[1] “开放”在研究论文、模型社区和软件许可证讨论中可能采用不同口径. 引用时应说明开放的是权重、代码、数据、配方还是服务访问，而非只写一个形容词.

字段	DeepSeek-V3	Qwen3-8B	GLM-4.5-Air
层数 / 隐藏宽度	$\frac{61}{7168}$	$\frac{36}{4096}$	$\frac{46}{4096}$
注意力	MLA；128 头；$r = 512, d_R = 64$	GQA；32 个 Q 头、8 个 KV 头；$d_h = 128$	GQA；96 个 Q 头、8 个 KV 头；$d_h = 128$
前馈结构	前 3 层稠密；其余 256 选 8，加 1 共享专家	稠密门控前馈；中间宽度 12288	第 1 层稠密；其余 128 选 8，加 1 共享专家
总 / 稀疏激活参数	$\frac{671}{37}$ B	8.2 B / 不适用	$\frac{106}{12}$ B
两字节逻辑权重	1342 GB	16.4 GB	212 GB
每序列每词元状态	$61(512 + 64) \times 2 = 70272$ 字节	$2 \times 36 \times 8 \times 128 \times 2 = 147456$ 字节	$2 \times 46 \times 8 \times 128 \times 2 = 188416$ 字节
训练与行为	预训练含 MTP；监督与强化后训练，文本对话	预训练及后训练；同一检查点切换思考 / 非思考	预训练及后训练；面向推理、代码和工具任务
权重许可声明	DeepSeek License Agreement 1.0	Apache License 2.0	模型卡声明 MIT

配置与模型卡分别固定于官方权重仓库的以下提交：DeepSeek-V3 为 `e815299`，Qwen3-8B 为 `b968826`，GLM-4.5-Air 为 `a24ceef`；完整定位见脚注. [2] 该 DeepSeek 制品声明 FP8 块量化，表中的 1342 GB 因而只表示把报告参数按两字节展开的统一估算；实际存储还要按量化块、尺度、未量化层和附加预测模块核算. GLM 的查询宽度 $96 \times 128 = 12288$ 不等于隐藏宽度 4096，投影仍可合法地从 4096 映射到 12288，不能套用 $d = h_q d_h$ 的默认等式.

发布时点也必须与配置快照分开. Qwen 官方发布公告日期为 2025 年 4 月 29 日，早于本章表中的报告提交；GLM-4.5V 模型卡明确记录 2025 年 8 月 11 日发布. [3]对其余本章案例，模型发布日未在此形成同等粒度的原始公告记录，故保留为未核定字段，不用报告日期冒充. 后续制品清单应同时保留发布事件、报告版本和权重提交三个时间维度.

表中能力栏是训练定位而非统一分数. DeepSeek 报告的生成基准、Qwen 的思考模式评估与 GLM 的工具环境有不同预算和评判条件；若要形成能力比较，应固定同一任务集、模板、生成总长度、工具调用次数与失败口径后再实验. 开放权重使本地测量成为可能，并不把发布者的基准自动变成当前部署的结果.

[2]配置、模型卡及许可材料的固定入口：DeepSeek-V3；Qwen3-8B；GLM-4.5-Air. 核对日期为 2026 年 9 月 8 日. GLM 许可栏依据该提交的模型卡声明；DeepSeek 模型使用该提交的 `LICENSE-MODEL`，不以代码的 MIT 许可证替代.

[3]分别见 Qwen3 发布公告和 GLM-4.5V 固定模型卡的 Project Updates. 这两个日期描述发布事件，不是上述配置提交的生成时间.

38.2 DeepSeek 模型体系

38.2.1 潜在注意力的低秩表示

DeepSeek-V2 把**多头潜在注意力**（Multi-head Latent Attention，MLA）与专家前馈结构结合，前者主要改变键值状态的表示，后者改变前馈计算的参与参数 [166]. 下面用独立记号分析 MLA 的低秩内容路径，位置分量随后加入.

设单个位置隐藏向量为列向量 $h_t \in \mathbb{R}^d$，压缩矩阵 $D \in \mathbb{R}^{r\times d}$，得到潜在状态 $c_t = Dh_t \in \mathbb{R}^r$. 第 i 个头的内容键和值写为

$$k_{t,i} = U_i c_t, \qquad v_{t,i} = V_i c_t, \quad U_i \in \mathbb{R}^{d_k\times r}, \quad V_i \in \mathbb{R}^{d_v\times r}. \tag{38.1}$$

查询内容向量为 $q_{t,i} \in \mathbb{R}^{d_k}$. 利用矩阵乘法结合律，某个历史位置 j 的内容分数为

$$q_{t,i}^{\mathsf{T}} k_{j,i} = q_{t,i}^{\mathsf{T}} U_i c_j = (U_i^{\mathsf{T}} q_{t,i})^{\mathsf{T}} c_j. \tag{38.2}$$

因此可先把当前查询映射到 r 维，再与缓存中的 c_j 相乘，而不逐一重建全部历史键. 注意力权重 $a_{tj,i}$ 确定后，值的加权和同样满足

$$o_{t,i} = \sum_{j\leqslant t} a_{tj,i} V_i c_j = V_i \left(\sum_{j\leqslant t} a_{tj,i} c_j \right). \tag{38.3}$$

这一步把逐历史位置的上投影移到加权求和之后. 若再与输出投影组合，可以继续吸收线性矩阵；非线性、位置相关变换或不相容的量化布局则会改变可融合条件.

上述恒等式对给定低秩参数化精确成立，却不证明任意已经训练好的 MHA 都能无损压成同一个 r. 低秩结构需要在训练中学习；把一个满秩映射事后截断，属于另一个可能引入近似误差的问题.

38.2.2 位置旋转分离

若直接在完整键上施加旋转矩阵 R_j，分数变成

$$(R_t q)^{\mathsf{T}} R_j U_i c_j = q^{\mathsf{T}} R_t^{\mathsf{T}} R_j U_i c_j. \tag{38.4}$$

中间的相对旋转依赖历史位置 j，通常不能与 U_i 任意交换，从而不能只预计算一个与 j 无关的变换查询. 相关位置代数见第七章《位置表示》.

MLA 采用独立的位置分量来保留旋转信息：将内容分数与位置分数相加. 若共享旋转键为 $k_j^R \in \mathbb{R}^{d_R}$，每头旋转查询为 $q_{t,i}^R$，则可写为

$$s_{tj,i} = \frac{(U_i^{\mathsf{T}} q_{t,i})^{\mathsf{T}} c_j + (q_{t,i}^R)^{\mathsf{T}} k_j^R}{\sqrt{d_k + d_R}}. \tag{38.5}$$

内容状态与位置状态分别缓存. 该分工保留低秩内容路径的结合律，又给位置关系留下明确通道；具体查询压缩与投影布局仍应按所选模型版本读取.

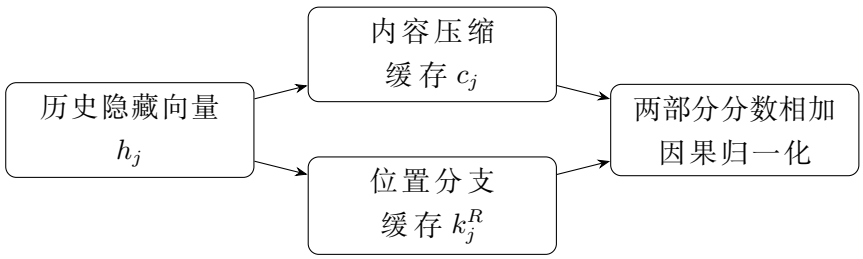


图 38.1: 潜在内容与位置分量的分工．图省略当前查询路径和输出投影，突出历史缓存的身份.

38.2.3　缓存压缩率

在所有层同构、键和值等宽、忽略元数据和分页冗余时，普通 MHA 每序列的缓存字节数为 $2LTh_qd_hb_c$，GQA 为 $2LTh_{kv}d_hb_c$，上述潜在形式为 $LT(r+d_R)b_c$．这些式子描述状态存储，尚未包含权重、临时工作区和运行时保留空间.

取构造配置 $h_q=16$、$h_{kv}=4$、$d_h=64$、$r=128$、$d_R=32$．每层每词元分别存储 2048、512、160 个元素．潜在形式相对于 MHA 的存储比例为 $\frac{160}{2048}=7.8125\%$，相对于 GQA 为 31.25%．这些数字来自给定配置，不能作为某个实际产品的实测节省率.

再取二维内容例题 $U=\begin{bmatrix}1&2\\0&1\end{bmatrix}$、$q=(1,3)^{\mathsf{T}}$、$c=(2,-1)^{\mathsf{T}}$.显式键为 $Uc=(0,-1)^{\mathsf{T}}$，点积为 -3；变换查询为 $U^{\mathsf{T}}q=(1,5)^{\mathsf{T}}$，与 c 的点积仍为 -3．例题核对的是结合律路径，不包括训练质量或内核效率.

38.2.4　V3 及 R1 的训练职责

DeepSeek-V3 报告讨论专家负载控制、多词元预测及训练数值效率 [209]．其中**多词元预测**（Multi-Token Prediction，MTP）是在训练中增加未来词元的预测约束，不应直接等同于服务端已经实现了某一种推测解码．训练目标、辅助模块是否保留、部署如何利用它们，需要分别说明.

V3 的**无辅助损失负载均衡**（Auxiliary-Loss-Free Load Balancing）把用于选专家的分数与用于混合输出的权重分开．设亲和度 $s_{ti}=\sigma(u_t^{\mathsf{T}}e_i)$，令

$$S_t=\mathrm{TopK}_i(s_{ti}+b_i),\qquad g_{ti}=\frac{\mathbf{1}[i\in S_t]s_{ti}}{\sum_{j\in S_t}s_{tj}}. \tag{38.6}$$

偏置 b_i 只改变选择集合；训练步结束后，对过载专家降低偏置，对欠载专家提高偏置．它不是乘入专家输出的概率，也不同于直接用均衡损失梯度修改路由器．报告仍保留很小的序列级辅助均衡项，因此名称不能解释为训练中完全没有任何均衡损失 [209]．实际配置还含节点分组限制和输出缩放，上式只展开选择与加权的关键分工；通用梯度与容量机制见第三十六章《混合专家模型》.

DeepSeek-R1 则把重点放在推理行为的后训练与强化学习 [15]. 基础网络结构、奖励构造和推断时允许的计算预算是三个不同因素. 更长的推理轨迹会增加生成成本，并不自动保证答案正确；奖励与验证器的机制在第二十四章至第二十六章展开. 蒸馏到其他基座的 R1 系列检查点应按学生架构计算资源，不能因名称含 DeepSeek 就套用 MLA 缓存公式.

38.2.5 V4.1 Flash 混合注意力

DeepSeek-V4.1-Flash 将图像与文本作为输入，自回归地产生文本，支持百万词元上下文. 以下依据 2026 年 9 月 10 日核查的官方技术报告，固定模型仓库提交为 `df42c109`；报告分别列出 552B 主干参数与 196B 条件记忆参数，预填充与解码阶段每词元激活参数分别为 8B 和 16B[210]. 这些量分别描述模型容量和计算路径，部署仍须另计权重精度、分片、缓存与工作区.

因果编码器—解码器（Causal Encoder-Decoder，CED）把 40 层主干分为 20 层因果编码器和 20 层解码器. 编码器按序列前缀生成表示，解码器的全局缓存由编码器末层表示投影得到；局部滑动窗口注意力仍使用各层自身的状态. 因果约束在两个部分中都成立，历史位置的表示不能读取未来待生成词元.

令序列长度为 N、主干宽度为 d，编码器输出为 $H^E \in \mathbb{R}^{N\times d}$. 对解码器层 ℓ，报告中的全局缓存投影可以写成

$$C_\ell = H^E W_\ell^{KV}, \qquad Z_\ell = H^E W_\ell^{Z}, \quad W_\ell^{KV} \in \mathbb{R}^{d\times d_c}, \quad W_\ell^{Z} \in \mathbb{R}^{d\times d_z}. \tag{38.7}$$

这里 $C_\ell \in \mathbb{R}^{N\times d_c}$ 为压缩前缓存条目，$Z_\ell \in \mathbb{R}^{N\times d_z}$ 为相应压缩权重表示，后续按压缩规则聚合；跨层复用还会减少实际独立缓存组. 由于生成全局缓存不要求先算出每个解码器层的历史隐藏状态，大部分提示词元可以跳过完整的解码器计算.

局部缓存仍需补齐. 模型采用有界重放，只在解码器重算提示末尾的 w 个词元以近似恢复滑动窗口状态. 若按每个词元经过一层的工作量计数，层数为 L，则预填充工作量由 NL 变为近似

$$\frac{NL}{2} + \frac{wL}{2}, \qquad \frac{\text{CED 工作量}}{\text{完整主干工作量}} \approx \frac{1}{2} + \frac{w}{2N}. \tag{38.8}$$

该估算用于 $N \gg w$ 的长输入，忽略投影、通信和不同注意力算子的成本差异；例如 $N = 64w$ 时比例为 $\frac{65}{128}$，不能直接解释为端到端时延比例. 生成新词元时仍需经过编码器与解码器. 多层局部注意力的理论感受野会随层数扩大，重放一个窗口属于近似恢复，报告中的小幅质量损失结论依赖其评测条件 [210].

38.2.6 Engram 条件记忆

条件记忆（Conditional Memory）通过输入相关的稀疏读取扩大可用表示容量. Engram 以局部 N-gram 为地址线索，从可训练嵌入表读取向量，再由当前隐藏状态

决定这些向量的贡献；MoE 则根据路由选择要执行的专家网络．两者可以同时存在，分别增加查表容量与条件计算容量 [211].

为说明信息流，令 x_t 为当前词元，$g_{t,n}$ 为截至位置 t 的长度 n 的规范化词元片段，$h_{n,k}$ 为第 k 个哈希函数，$E_{n,k} \in \mathbb{R}^{M_{n,k}\times d_e}$ 为对应嵌入表．查表与融合的教学抽象为

$$e_t = \mathrm{Concat}_{n,k}\big(E_{n,k}[h_{n,k}(g_{t,n})]\big), \qquad \widetilde{h}_t = h_t + \alpha_t \odot V e_t. \tag{38.9}$$

其中 $h_t, \widetilde{h}_t \in \mathbb{R}^d$，若拼接宽度为 D，则 $V \in \mathbb{R}^{d\times D}$；门控 α_t 由上下文状态与所读记忆共同计算，并按形状广播．该式只抽象寻址、投影和门控的职责，具体模型还包含归一化和多分支融合．固定片段阶数与哈希头数后，每词元访问的表项数不随表容量增长；这不意味着访存延迟为零．哈希碰撞、低频片段覆盖、分词方式和门控质量都会影响记忆是否有用.

V4.1 Flash 采用两个 Engram 模块，分别位于从零计数的第 1 层和第 14 层，合计 196B 参数. 每个模块使用 2、3、4 阶片段，每阶 8 个哈希头、合计 2048 维嵌入，每头表容量约 1600 万项，取不同素数；嵌入表及键值投影采用 FP8. 相较原始 Engram，该版本省去短因果卷积，并调整嵌入优化方法．因此解释该制品时应以 V4.1 报告为准，不能逐项套用早期论文结构 [210].

表地址由词元确定，可以提前计算，并从主机内存预取，使传输与网络计算重叠；收益取决于带宽、批量与调度．仅按 196B 表参数每参数一字节估算，存储已约为 196GB，另需尺度和运行状态．Engram 保存训练得到的参数化记忆，RAG 访问外部文档，KV 缓存保存本次前缀的中间状态；三者的更新方式、来源可追溯性和存储生命周期各不相同.

38.2.7　CSA2 及缓存精度

压缩稀疏注意力第二代（Compressed Sparse Attention 2，CSA2）进一步沿层维度复用全局缓存和稀疏位置选择．Full 模式生成缓存并执行索引；Reindex 模式复用先前的缓存，以本层索引查询重新选择位置；Reuse 模式连同已选位置一起复用．各层仍保留自己的全局查询与局部窗口缓存．解码器的分层稀疏索引还把后续索引的搜索范围限制在较早层产生的候选池中 [210].

报告给出的全局缓存为每词元 890 字节，结合 CSA2 与 FP4 主缓存约为 V4-Flash 的四分之一；有界重放避免把局部窗口缓存持久化至 SSD，使持久缓存约为前代的八分之一．全局缓存、运行期全部缓存和持久缓存具有不同统计口径，不能把两个缩减比例相乘．按一百万词元计算，全局缓存项约为 890MB，实际服务还需局部缓存、权重、临时张量、分配器及并发请求空间．以上为发布者报告中的结构与资源结果，不能替代指定硬件、后端和任务上的容量测试.

V4.1 Flash 还采用 Single-Pass mHC 残差混合和 DSpark 推测解码，并从预训练阶段联合处理视觉与文本．各机制解决的成本不同：CED 减少长输入的主干计算，

CSA2 减少全局缓存及重复索引，Engram 扩大条件记忆容量，推测解码减少串行生成等待. 能力变化还受预训练数据、后训练与推理预算影响，不能仅凭综合基准的提高推断某一模块的独立收益.

38.3 Qwen 模型体系

38.3.1 稠密模型及专家模型

Qwen3 技术报告同时给出稠密与 MoE 模型. 其文本主干使用 GQA、RoPE、预归一化与门控前馈层，并在查询和键上引入归一化；报告中的 MoE 版本采用细粒度专家且不设共享专家 [206]. 这些细节说明，不能把另一个 MoE 家族的共享专家公式直接移植过来.

同一家族内也存在具体配置差异. 例如该报告列出的 Qwen3-4B 与 Qwen3-8B 都具有 36 层和 32 个查询头、8 个键值头，但嵌入权重是否共享不同. 即使注意力缓存的主要形状相同，参数存储与输出投影布局仍可能不同. 读取家族综述后，还必须定位具体检查点的配置.

从机制上看，GQA 改变每个键值头服务多少查询头；查询键归一化约束相应向量尺度；专家层改变前馈参数的选择. 这三者作用在不同位置，不能用“采用某优化架构”概括为一个无法核算的收益.

38.3.2 推理模式及输出预算

推理模式和直接回答模式改变模型的交互协议及生成过程，可能使用控制词元、模板开关或不同检查点. 给 API 传入一个名称相似的参数，不足以说明目标服务已经实现相同语义；必须核对序列化后的实际输入和返回协议.

设每个请求的隐藏推理或中间输出长度为 T_r，最终答案长度为 T_a. 若两者均由同一个解码器生成，生成步数约为 $T_r + T_a$，容量规划不能只统计用户可见答案. 比较两个模式时，应分别记录正确率、有效完成率、总生成词元、延迟和失败类型. 推理预算是实验条件，而不是无成本的能力开关.

38.3.3 视觉语言分支的额外路径

Qwen2.5-VL 是另一个明确的版本案例，其报告讨论视觉输入处理以及与语言主干的结合 [212]. 视觉语言制品必须同时固定图像或视频预处理、视觉编码器、连接模块和语言分词器. 文本主干的上下文上限不直接等于可输入的图片数量，因为每幅图像可能转化为不同数量的视觉位置.

对图片 I，可以抽象写为 $Z_I = C(F_v(I;\psi);\omega)$，其中 F_v 是视觉编码器，C 将视觉特征映射到语言模型使用的表示空间. 若分辨率策略使视觉位置数从 N_v 增至 $4N_v$，

即使文本不变，后续序列长度和跨模态计算也会变化．详细融合方法见多模态篇；本章强调它们是制品身份和资源模型的一部分．

38.3.4　代码模型分支

Qwen2.5-Coder-7B-Instruct 提供一个任务专用案例：固定配置仍是因果 Transformer，28 层、隐藏宽度 3584、28 个查询头和 4 个键值头，具有 QKV 偏置，输入输出嵌入不共享．它的专用性主要来自代码相关语料、任务构造和后训练，不能从“Coder”后缀推导出一种新的注意力算子．[4]

代码生成的条件可以是自然语言需求，也可以是已有程序的前缀和后缀．对缺失中段 m，目标为 $p_\theta(m \mid c_{\text{pre}}, c_{\text{suf}})$；只要把已知两侧放入模型规定的序列协议，因果解码器仍能逐步生成中段．这里的“因果”约束是序列化后的生成顺序，不要求条件只能来自源文件左侧．代码能否编译、测试能否通过以及是否满足需求则是不同评价层次，不能用自然语言流畅度替代．

配置还说明长上下文条件不能只读宣传数字：该快照的直接位置上限是 32768，模型卡讨论借助 YaRN 处理更长输入．扩展配置、后端支持和短长任务评估必须一起保留，相关相位条件见第三十七章《长上下文建模》．

38.4　Llama 及 Mistral 模型体系

Llama 与 Mistral 是英文社区中使用最广的两条开放权重路线．它们与前述模型家族在结构上有重叠，也有明确差异；与 DeepSeek、Qwen 并列分析可以看出，“开放权重”并不对应单一架构．

38.4.1　Llama 架构演进

Llama 系列早期检查点确立了开放权重解码器的常见配方：预归一化的 RMSNorm、旋转位置编码、门控前馈与分组查询注意力．这一组合并非首次出现，但被大量派生模型复现，成为比较其他结构的基线．因此，遇到一个使用分组查询注意力的开放模型时，注意力形式本身不构成家族识别依据，仍需读取逐层配置．

Llama 4 是原生多模态模型，采用专家混合架构与早期融合，把图像与文本在输入阶段接入同一主干 [208]．该系列公开两个检查点：Scout 每词元激活约 17 十亿参数、含 16 个专家，Maverick 每词元激活约 17 十亿参数、含 128 个专家，二者上下文窗口均为 128K．同代内专家数与计算量已经不同，前代稠密模型的部署经验因此不能直接外推：状态缓存、处理器依赖和模态预算都已改变．

[4] 见官方固定制品配置与模型卡；该卡报告 7.61B 参数、预训练与后训练阶段，声明 Apache-2.0．代码语料及训练方法参见 Qwen2.5-Coder 技术报告．

该系列使用社区许可与可接受使用政策，而不是宽松开源许可；规模、分发、衍生训练与地域条款需要按精确版本的许可文本审查，品牌名本身不构成授权. [5]

38.4.2 Mistral 架构特征

Mistral 7B 用滑动窗口注意力限制每个查询可见的历史范围 [213]. 设窗口宽度为 W，第 ℓ 层的有效感受野 R_ℓ 满足

$$R_1 = W, \qquad R_\ell = R_{\ell-1} + (W-1) \implies R_\ell = 1 + \ell(W-1). \tag{38.10}$$

可见范围随层数线性增长，而每层缓存不再随序列总长度增长：

$$M_{\text{sw}} = 2 b_c L W h_{kv} d_h, \qquad T \gg W. \tag{38.11}$$

窗口是一种结构先验：超出窗口的依赖只能逐层传播，信息在传递中可能衰减. 若 $W = 4096$、$L = 32$，则 $R_L = 1 + 32 \times 4095 = 131041$，覆盖范围超过十万词元；这个数字是层数与窗口的算术结果，不代表该模型在所有长距离任务上的有效能力，相关评估条件见第三十七章《长上下文建模》.

Mixtral 8x7B 把门控前馈替换为每个词元选择 2 个专家的稀疏前馈，报告的总参数约 46.7 十亿，每词元激活约 12.9 十亿 [214]. 这正是第三十六章《混合专家模型》讨论的容量与计算分离：激活参数量描述单次前向涉及的参数，权重驻留仍按总参数量核算.

值得注意，Mixtral 8x7B 的公开配置把滑动窗口设为空值，注意力仍是完整因果形式；滑动窗口属于 Mistral 7B 的该版本选择，不能被外推到同家族的专家型号. 同一家族的注意力与位置设置需要逐检查点读取，这正与前面反复强调的方法一致. 两个制品均以宽松开源许可发布，但其后续型号的许可与开放范围需要另行核对. [6]

表 38.3: 两条开放权重路线的结构差异. 参数规模采用报告口径，本表不构成能力排序.

制品	注意力与位置	前馈结构与激活	开放条件
Mistral 7B	分组查询注意力；滑动窗口 $W = 4096$	稠密门控前馈	宽松开源许可
Mixtral 8x7B	完整因果注意力；分组查询注意力	8 选 2 专家；总 46.7B / 激活 12.9B	宽松开源许可
Llama 4	分组查询注意力；旋转位置编码	专家前馈；原生图文早期融合	社区许可与可接受使用政策

[5] 结构、参数与许可依据 Llama 4 模型卡固定提交；仓库快照另见 Scout 制品提交. 核对日期为 2026 年 9 月 12 日. “17B” 指每词元激活参数，不是总参数.

[6] 配置字段与许可依据固定提交：Mistral-7B-v0.1（滑动窗口 4096、32 查询头、8 键值头）与 Mixtral-8x7B-v0.1（滑动窗口为空、8 专家选 2）. 核对日期为 2026 年 9 月 12 日.

38.5　GLM 模型体系

38.5.1　自回归空白填充的早期目标

早期 GLM 研究**自回归空白填充**（Autoregressive Blank Infilling），将部分文本片段挖空，再按指定次序生成缺失片段 [215]. 设保留上下文为 c，缺失片段为 $s_1, \ldots, s_m$，每个片段序列均包含最后的结束片段标记，其长度也计入该标记；生成顺序为排列 π. 这样结束事件与内容词元一起参与概率分解：

$$p_\theta(s_{\pi_1}, \ldots, s_{\pi_m} \mid c) = \prod_{j=1}^{m} \prod_{t=1}^{|s_{\pi_j}|} p_\theta(s_{\pi_j,t} \mid c, s_{\pi_{<j}}, s_{\pi_j,<t}). \tag{38.12}$$

已知上下文与已经生成的内容可以作为条件，当前片段尚未生成的后缀不能提前泄漏. 这与同时独立预测多个 MASK 位置的 MLM 不同，也与只把原文从左到右生成的单一顺序不同.

以文本“甲去乙，然后返回丙”为例，若挖去“乙”和“丙”，保留上下文可以同时提供两侧约束，但两个待填片段的生成仍具有明确次序. 模型需要知道当前生成的是哪一个空白、空白内部已生成到何处；不能只给一个没有位置含义的 MASK 列表.

38.5.2　现代推理及工具能力的训练

GLM-4.5 报告采用 MoE，并围绕智能体、推理和代码任务组织训练，提供推理与非推理模式 [216]. 它是现代模型案例，不能仅凭 GLM 家族名就推断沿用早期论文的全部注意力掩码和空白填充协议.

工具调用能力至少涉及三层：网络输出调用意图，序列协议编码工具名与参数，外部执行器检查权限并产生观察. 模型输出形式合法，不代表参数符合工具语义；模型希望执行某动作，也不构成用户授权. 训练集中的调用与观察必须保持关联，相关系统设计在智能体架构与应用开发章展开.

评估智能体能力时，工具环境、任务终止条件、重试预算与评判器都属于实验定义. 同一模型在不同环境中可能得到不同完成率，因此技术报告中的智能体分数不能脱离其环境作为通用性能常数.

38.5.3　视觉分支的结构及输入边界

GLM-4.5V 是独立的视觉语言制品，其模型卡指向 GLM-4.5-Air 基座，而不是把任意 GLM-4.5 权重直接接收图像. 固定配置中，视觉编码器深度 24、宽度 1536，输出映射到 4096 维语言空间；空间 patch 宽度 14、空间合并因子 2，另有时间 patch 宽度 2. [7]

[7] 见 GLM-4.5V 固定模型卡及配置. 这些是该提交的结构字段，不表示所有输入固定为同一分辨率；实际网格必须连同图像 / 视频处理器读取.

视觉网格的空间合并 若处理后的网格为 (T_v, H_v, W_v) 且空间维可被二整除，二乘二的空间合并把视觉位置数从 $T_vH_vW_v$ 变为 $\dfrac{T_vH_vW_v}{4}$. 例如单时间网格的 24×24 个 patch 对应 576 个合并前位置、144 个合并后位置. 该算例只说明给定网格的序列缩减，不把任意原图强行缩成同一大小；边界标记和文本还要额外计入长度.

该制品为图像、视频分别设置标记，并配置多维位置分配. 即使语言层数与 Air 相同，视觉处理器、位置协议和停止编号也成为新的兼容条件. 其图像推理、文档理解和界面操作能力应在相应输入及工具环境中评估；视觉表示学到对象，不等于外部执行器已获得操作权限. 多模态融合及视觉序列推导分别见第三十九章《视觉 Transformer》 和多模态篇.

38.6 其他开源模型提供的技术比较

同一套比较方法可以覆盖不同地区与不同开放程度的制品. 下面选取机制上互不重复的补充案例.

38.6.1 Kimi K2 智能体环境

Kimi K2 的技术报告以稀疏模型及智能体能力为重要研究对象 [217]. 作为比较案例，它提示读者把参数效率与交互任务共同分析：一次用户任务可能需要多轮模型调用和工具反馈，单次生成速度无法直接推导任务完成时延.

设一次任务经过 J 轮，每轮模型耗时为 t_j^m，工具耗时为 t_j^u，串行情况下总时间近似为 $\sum_j (t_j^m + t_j^u)$；并行工具调用、重试和缓存则改变这个表达. 即使某个模型单轮稍慢，若完成任务所需轮数更少，总时延也可能下降. 反过来，较高的单轮吞吐不能补偿大量无效工具调用.

38.6.2 MiniMax-01 混合注意力

MiniMax-01 报告研究 Lightning Attention 与混合结构 [218]，可用于理解为何某些模型不在所有层保存完整的逐词元键值. 为说明一般线性状态机制，考虑独立的外积累积模型

$$S_t = S_{t-1} + k_t v_t^{\top}, \qquad y_t = S_t^{\top} q_t, \quad S_t \in \mathbb{R}^{d_k \times d_v}. \tag{38.13}$$

其状态大小不随历史长度增长，但有限矩阵将历史压缩进共同状态，不能逐项访问任意过去词元. 该简化式只说明线性累积原理，不等同于 MiniMax 的完整门控、归一化或实现公式.

将部分层保留为完整注意力，另一部分采用递推状态，会形成不同的容量与信息访问折中. 若完整注意力层数为 L_a，状态层数为 L_s，缓存近似为

$$M(T) = b_c\big(2L_a T h_{kv} d_h + L_s d_k d_v\big), \tag{38.14}$$

其中暂按每个状态层只有一个上述矩阵计算；多头和附加状态需再乘相应系数．只要 $L_a > 0$，整体缓存仍有随 T 线性增长的部分，不能把混合模型称为“任意长度零增长”．

38.6.3　开放许可模型

开放程度不同，能够支持的工程结论也不同．Olmo 3 在 7B 与 32B 两个尺度上发布，除权重外还公开完整模型流程，包括构建该家族所用的每个阶段、检查点、数据点与依赖，训练数据为 Dolma 3[204]．这种完全开放适合作为方法对照：外部可以复现预训练链条并审计数据来源，但它并不因此成为能力上限．

Gemma 4 采用宽松开源许可 [207]，而该家族早期版本使用自有条款．这说明同一品牌的不同代际必须分别取证：一个模型的宽松许可不能回溯适用于旧检查点，反之亦然．该系列同时提供稠密与专家版本，注意力在局部滑动窗口与全局之间交替，并规定末层为全局层；以 26B A4B 为例，它含 128 个专家、每词元选择 8 个并另有 1 个共享专家，激活参数远小于总参数．窗口宽度在不同尺寸上取 512 或 1024，因此部署参数必须按选定检查点读取．

gpt-oss 以宽松开源许可发布文本开放权重模型，并以固定的响应格式描述推理强度与工具调用 [205]．此时协议属于模型行为的一部分：模型卡明确要求按该格式使用，绕过它直接拼接提示可能得到语法合法但语义不同的结果．开放权重也不降低对工具执行权限与运行环境的控制要求．Phi-4 则说明数据质量导向的配方与合成数据可以让一个约 14B 的模型在限定任务上具备竞争力 [219]；许可宽松与体积小都不减少评测义务，知识截止、语言覆盖与长上下文边界仍需按实际任务测量．

表 38.4: 三个宽松许可开放权重制品的字段比较．参数与许可采用发布者口径，配置数值取自各制品的固定提交；表内数值不构成能力比较．

字段	Olmo 3 32B	Gemma 4 26B A4B	gpt-oss-120b
层数 / 隐藏宽度	$\frac{64}{5120}$	$\frac{30}{2816}$	$\frac{36}{2880}$
注意力	48 层局部窗口 4096 与 16 层全局交替	25 层局部窗口 1024 与 5 层全局交替	18 层全局与 18 层局部窗口 128 交替
前馈结构	稠密；中间宽度 27648	128 选 8 专家, 另有 1 共享专家	128 选 4 专家
总 / 激活参数	32 B / 不适用	$\frac{25.2}{3.8}$ B	$\frac{117}{5.1}$ B
位置上限	65536	262144	131072
权重许可	Apache License 2.0	Apache License 2.0	Apache License 2.0

三个制品都允许自由使用与再分发，但结构差异很大：Olmo 3 用稠密前馈配混合注意力，Gemma 4 与 gpt-oss 都使用细粒度专家，而专家数、选择数与注意力交替

比例各不相同. 许可宽松不意味着结构相近，也不意味着权重驻留量可以由激活参数量推算. 固定提交见脚注. [8]

38.6.4 按机制扩展比较对象

Falcon、Pythia、InternLM、ERNIE 及其他开放权重模型可以沿同一方法纳入研究：先选择具体报告和检查点，再确定它带来的新增比较维度，例如领域语料、训练目标、异构专家或模态处理. 没有足够来源时应保留为待研究对象，不补写推测规格.

本章版本案例	主要比较问题	不能据此推断
DeepSeek-V2/V3/R1 及 V4.1 Flash	缓存压缩、专家计算、条件记忆与后训练的分工	蒸馏学生必然继承教师架构.
Qwen3 / Qwen2.5-VL	同代结构变体、交互模式及视觉路径	所有 Qwen 版本共享同一配置.
Llama 4 / Mistral / Mixtral	稠密与稀疏前馈、滑动窗口、社区许可	同一品牌的所有版本共享许可与结构.
早期 GLM / GLM-4.5	空白填充目标与现代任务训练的演进	品牌延续意味着目标与掩码永久不变.
Olmo 3 / Gemma 4 / gpt-oss	开放范围、许可代际差异与响应协议	开放权重等于数据、代码与配方全部开放.
Kimi K2	稀疏模型与多轮智能体任务	单轮生成指标等于任务效率.
MiniMax-01	递推状态与完整注意力的混合	所有层都能使用固定大小缓存.

表格不列统一能力排名. 表 38.2 以三个固定制品示范结构、训练、协议与许可字段的填法；未列入其中的制品仍须分别建立记录，各技术报告的基准数字也不能被拼成未经同条件评估的总榜.

38.7 模型结构还原

38.7.1 制品层次及一致性

模型卡提供发布者声明；配置提供机器可读结构；权重张量提供实际存储事实；模型代码定义如何执行这些张量. 四者相互补充，任何单一文件都不足以证明整套制品可用.

例如配置声明嵌入为 $V \times d$，实际权重却为 $(V + 8) \times d$，可能是新增控制词元，也可能是版本错配. 应同时读取分词器的有效编号范围和输出投影，而不直接把差异

[8] 本小节固定入口：Olmo-3-1125-32B；Gemma-4-26B-A4B-it；gpt-oss-120b；Phi-4. 核对日期为 2026 年 9 月 12 日. Olmo 3 与 Gemma 4 的激活参数口径来自各自报告，gpt-oss 为模型卡口径.

判成错误或默认为无害. 对于连续编号，最大词元编号必须小于实际嵌入行数；存在填充词表行时，行数又可能大于有效词元数.

权重分片索引需要把张量映射到实际文件，文件头中的形状、数据类型与偏移必须相容. 普通未量化矩阵的参数数为各维长度之积；量化打包张量还包含尺度和其他状态，其存储元素数不能直接当成原网络参数量.

38.7.2　格式、适配器及远程代码

模型适配器（Model Adapter）依赖基座参数和目标模块，不能作为完整模型独立解释. 低秩增量、量化权重、多模态处理器和派生容器都需要保留来源关系. 修改基座版本或模块命名后，即使加载程序没有立刻报错，数学组合也可能已改变.

Safetensors 的元数据读取与需要通用对象反序列化的格式不同，但文件格式本身不证明来源可信. 静态解析仍需限制头部大小、验证偏移范围和文件完整性. 未知模型仓库要求执行自定义代码时，应把代码纳入明确的依赖与审阅范围，而不因自动加载接口提供一个开关就视为已经获得信任.

38.7.3　身份清单的伪代码

算法 38.1 建立模型结构与协议清单

输入：固定来源和版本的模型文件、声明文档、允许读取的格式与结构规则. **输出：**证据清单、确定冲突、仍不可判定的项目.

1. 记录来源版本与文件路径；按字节流计算摘要，保留原始文档及获取时间，不执行模型代码.
2. 读取配置，提取逐层类型、维度、头数、专家布局、位置机制、量化声明和处理器依赖.
3. 读取分词器与模板，记录实际编号范围、控制词元和生成终止规则；多模态制品还记录媒体处理参数.
4. 在限定大小与偏移边界下读取权重元数据，将配置声明与张量形状、索引和分片逐项关联.
5. 将明确相容项记为已观察一致，明确矛盾记为冲突，不能判断项记为未知；不得把没有报错当作一致证据.
6. 输出不可变清单及后续运行待验项. 量化、质量、实际内核与工具执行行为由后续受控评估负责.

若文件总字节数为 S，完整摘要至少需要读取 $O(S)$ 字节；只读头部不能提供完整权重摘要. 身份清单不等于运行或任务质量证明.

38.8 模型选型条件

38.8.1 硬约束及同条件比较

先明确输入模态、部署地点、允许用途、可用显存与时延上限，再在可行候选中比较质量. 开放范围与许可在这一步就是准入条件，而不是上线前才补做的合规检查. 资源模型至少写成

$$M_{\text{peak}} = M_{\text{weights}} + M_{\text{state}} + M_{\text{workspace}} + M_{\text{runtime}}, \tag{38.15}$$

其中峰值工作区随批量、上下文和后端变化. MoE 没有卸载或分片时，权重项仍按总参数计算，不能按激活参数量估算全部驻留显存.

许可与开放范围同样构成硬约束. 候选准入应固定仓库标识、许可证文本与文件摘要，再分别判断权重、代码和数据各自允许的用途. 完全开放的制品允许复现训练链路，便于定位数据与配方问题；只开放权重的制品可以做权重级实验，但数据来源与训练细节的可核查范围有限. 两者都能形成可用系统，差别在于出现质量或合规问题时能够追溯到哪一层.

任务比较应固定评测数据与版本、模板、采样、思考预算、工具环境、后端和硬件；每一项不能固定时，就把它作为比较条件而非隐藏变量. 报告中的分数属于来源证据，自己执行得到的数字属于实测证据；二者须明确标注.

38.8.2　适配过程中保留语义边界

统一业务消息格式有助于上层应用维护，但不同模型仍需要各自的序列化适配. 适配器应负责消息角色、工具参数结构、媒体位置、结束标记和错误恢复，而不对所有模型强行使用同一串提示文本. 共同业务语义不等于相同词元序列.

评估时还应分开模型错误、模板错误、解析错误和工具执行失败. 某次任务未完成，既可能是模型推理错误，也可能是输出已正确但协议解析丢失字段. 只有按系统层次定位，才能判断更换模型是否真的解决问题.

第三十九章　视觉 Transformer

图像是一张具有局部邻接关系的二维网格，而标准 Transformer 接收一列向量. 两种表示之间的转换不仅涉及张量形状，还涉及像素信息保留、空间坐标、全局交互和计算预算. 把图像展成一个长向量，会失去可独立交互的位置；把每个像素作为一个位置，则会使全局注意力代价过高. **视觉 Transformer**（Vision Transformer，ViT）以局部补丁作为序列单元，在两者之间选择一种可训练的表示粒度 [220].

本章研究视觉编码与分类. 图像编码器如何连接语言模型，以及视觉生成如何建立概率模型，分别由多模态与生成模型相关章节展开. 视觉编码器输出一个向量，不表示它已经获得语言对齐能力；分类准确率也不能单独证明其细粒度空间表示适用于定位任务.

39.1　图像补丁表示

39.1.1　形状及坐标

补丁（Patch）是图像中按规则取出的局部像素区域. 先考虑无重叠、无填充的矩形补丁，图像张量采用批量、通道、高、宽的顺序. 表 39.1 统一本章符号，所有坐标从零开始.

表 39.1: 视觉编码的主要符号.

符号	含义或形状
$I \in \mathbb{R}^{B\times C\times H\times W}$	预处理后的图像批量
P_h, P_w	补丁高度与宽度
$G_h = \dfrac{H}{P_h},\ G_w = \dfrac{W}{P_w}$	补丁网格的行数与列数
$N = G_h G_w,\ F = CP_hP_w$	补丁数量与每个补丁的像素分量数
$d, a, d_h = \dfrac{d}{a}$	隐藏维度、注意力头数与每头维度
$L = N + 1,\ T$	含分类标记的序列长度、编码层数
$K, \rho d$	类别数、前馈中间层宽度

标准补丁化的前提

H 与 W 分别可被 P_h 与 P_w 整除；补丁不重叠、不丢弃边界像素；输入的通道顺序与像素归一化已经确定．固定按补丁行优先、块内通道优先及像素行优先排列．不同排列同样可以定义有效模型，但必须同时变换后续权重和位置映射．

设网格坐标为 (r,s)，补丁内坐标为 (u,v)，通道为 c．**补丁化**（Patchification）可以写成精确的索引映射：

$$n = rG_w + s, \qquad f = (cP_h + u)P_w + v, \tag{39.1}$$

$$X_{b,n,f} = I_{b,c,rP_h+u,sP_w+v}, \qquad X \in \mathbb{R}^{B\times N\times F}. \tag{39.2}$$

由 n 可以恢复 $r = \lfloor \frac{n}{G_w} \rfloor$、$s = n \bmod G_w$，由 f 可以恢复 c,u,v．因此，式 (39.2) 是一个双射，元素总数满足 $BNF = BCHW$．在这些前提下，切分只是重排，没有学习参数，也没有丢失像素．

这里必须区分逻辑重排与物理复制．张量布局允许时，某些重排可以由步长视图表示；后续矩阵运算若要求连续内存，仍可能发生复制．因此，元素数相等不能推出额外内存开销为零．[1]

39.1.2　线性投影及信息保留

补丁嵌入（Patch Embedding）将每个补丁映射到共同隐藏空间：

$$E = XW_E + \mathbf{1}b_E^{\mathsf{T}}, \qquad W_E \in \mathbb{R}^{F\times d},\quad b_E \in \mathbb{R}^d,\quad E \in \mathbb{R}^{B\times N\times d}. \tag{39.3}$$

同一个 W_E 作用于所有补丁，参数量为 $Fd + d$，不随图像补丁数增长．与像素重排不同，投影可能损失信息：若 $d < F$，则 $\operatorname{rank}(W_E) \leqslant d < F$，必然存在非零行向量 δ 满足 $\delta W_E = 0$．补丁 x 与 $x+\delta$ 因而具有完全相同的嵌入．即使 $d \geqslant F$，也只有矩阵具有满行秩时才可能从嵌入唯一恢复补丁．

这并不意味着压缩一定有害．分类目标通常不要求保留每个像素扰动，模型可以学习舍弃与标签无关的变化．但对于细小文字、边缘缺陷等任务，若有用变化进入投影的零空间，后续注意力无法恢复它．补丁粒度与投影维度因此是任务信息与计算资源之间的结构选择．

[1] 库中的滑窗提取操作往往也支持重叠和填充；只有核尺寸、步幅及边界规则满足本节前提时，才对应这里的无重叠补丁化．

39.2　补丁投影及卷积的等价性

令二维卷积核为 $\mathcal{K}\in\mathbb{R}^{d\times C\times P_h\times P_w}$，步幅为 (P_h,P_w)，无填充、无膨胀．常用深度学习卷积操作实际按不翻转卷积核的互相关形式计算：

$$Y_{b,j,r,s}=b_{E,j}+\sum_{c=0}^{C-1}\sum_{u=0}^{P_h-1}\sum_{v=0}^{P_w-1}\mathcal{K}_{j,c,u,v}I_{b,c,rP_h+u,sP_w+v}. \tag{39.4}$$

将参数对应为

$$\mathcal{K}_{j,c,u,v}=(W_E)_{(cP_h+u)P_w+v,j}, \tag{39.5}$$

再令 $n=rG_w+s$，式 (39.4) 就逐项等于式 (39.3) 的 $E_{b,n,j}$．卷积输出的 $[B,d,G_h,G_w]$ 还需按同一行优先规则重排为 $[B,N,d]$．这是一种参数与运算的等价，并不意味着两种实现对任意内存布局和硬件都具有相同速度．

从梯度也能看出这种对应．把批量与补丁位置合并，令 $G=\dfrac{\partial\mathcal{L}}{\partial E}$，则

$$\frac{\partial\mathcal{L}}{\partial W_E}=X^{\mathsf{T}}G,\qquad \frac{\partial\mathcal{L}}{\partial X}=GW_E^{\mathsf{T}},\qquad \frac{\partial\mathcal{L}}{\partial b_E}=\sum_{b,n}G_{b,n,:}. \tag{39.6}$$

在无重叠情况下，X 的梯度按逆映射放回图像即可．若使用重叠滑窗，一个像素会被多次取出，其梯度必须将各窗口贡献相加，不能用一次覆盖写回．

考虑单通道、单图像，令

$$I=\begin{bmatrix}1&2&3&4\\5&6&7&8\\9&10&11&12\\13&14&15&16\end{bmatrix},\quad P_h=P_w=2,\quad W_E=\begin{bmatrix}1&1\\1&-1\\1&1\\1&-1\end{bmatrix},\quad b_E=0. \tag{39.7}$$

按式 (39.1) 得到

$$X=\begin{bmatrix}1&2&5&6\\3&4&7&8\\9&10&13&14\\11&12&15&16\end{bmatrix},\qquad XW_E=\begin{bmatrix}14&-2\\22&-2\\46&-2\\54&-2\end{bmatrix}. \tag{39.8}$$

第一个输出通道计算四个像素之和，第二个通道计算左右差．对应卷积核分别为

$$\mathcal{K}_0=\begin{bmatrix}1&1\\1&1\end{bmatrix},\qquad \mathcal{K}_1=\begin{bmatrix}1&-1\\1&-1\end{bmatrix}. \tag{39.9}$$

卷积第一通道输出网格 $\left[\begin{smallmatrix}14&22\\46&54\end{smallmatrix}\right]$，第二通道输出全为 -2．重排后得到同一个 4×2 矩阵．这个构造算例说明像素索引与参数映射的等价，不是分类实验，也没有说明这两个人工核具有良好的视觉表征能力．

39.2.1　重叠、填充及边界

一般滑窗在高度方向的输出数为

$$G_h = \left\lfloor \frac{H + 2p_h - \delta_h(P_h - 1) - 1}{S_h} \right\rfloor + 1, \tag{39.10}$$

其中 p_h, S_h, δ_h 分别为对称填充、步幅和膨胀系数，宽度方向同理. 步幅小于窗口尺寸会产生重叠；步幅过大可能跳过像素；不能整除时，无填充实现可能丢弃尾部. 它们仍可与相同滑窗规则的线性投影对应，但已不满足前述双射.

对变尺寸图像采用填充时，需要区分完全由填充构成的补丁与部分有效补丁. 注意力掩码只能排除整个位置，不能自动去除一个补丁内部的无效像素影响. 通常应先明确裁剪或填充策略、有效区域以及目标任务是否允许改变长宽比，再决定补丁规则.

39.3　二维位置及序列组装

39.3.1　空间位置编码

不含位置编码的注意力对序列置换具有等变性. 令 Π 为位置置换矩阵，则线性映射满足 $Q' = \Pi Q, K' = \Pi K, V' = \Pi V$. 逐行 softmax 满足

$$\mathrm{softmax}(\Pi S \Pi^{\mathsf{T}}) = \Pi\, \mathrm{softmax}(S) \Pi^{\mathsf{T}}, \tag{39.11}$$

因此

$$\mathrm{Attn}(\Pi Z) = \Pi\, \mathrm{Attn}(Z). \tag{39.12}$$

逐位置归一化、前馈层和残差都保持这一性质. 若固定分类标记的位置，仅置换其余补丁，分类位置输出不变；均值池化同样无法区分这些排列. 因此，单纯把上方补丁排在数组前面，并没有使模型知道“上方”的含义. 补丁内部的像素坐标由投影权重区分，补丁之间的空间坐标则需要另行提供.

分类标记（Classification Token，CLS）是一个跨图像共享的可训练向量 $e_{\mathrm{cls}} \in \mathbb{R}^d$. 标准序列初始化为

$$Z^{(0)} = [e_{\mathrm{cls}}; E_0; \ldots; E_{N-1}] + P, \quad P \in \mathbb{R}^{(N+1)\times d}. \tag{39.13}$$

位置参数按批量广播，分类向量本身不含当前图像信息，其依赖图像的表示是在编码器中通过交互形成的.

39.3.2　位置参数的二维语义

绝对位置嵌入（Absolute Positional Embedding）为每个位置提供一个与内容相加的向量. 标准 ViT 可以把参数存成一张一维表，但表的第 $1 + rG_w + s$ 行实际对应

二维网格坐标 (r, s). 因此，“存成一维” 不等于 “只能表达一维位置”. 另一种设计是分解为 $P_{r,s} = P_r^h + P_s^w$，其参数更少，但位置表的可表达形式受到可加性约束.

二维位置插值（Two-Dimensional Positional Interpolation）用于将固定网格的位置参数适配到另一网格. 须先分离 CLS 行，再把补丁部分恢复为 $G_h \times G_w \times d$，按空间坐标插值，然后重新展平. 把整个序列当作一条线插值，可能在旧网格一行末尾和下一行开头之间建立不符合二维邻接的混合.

以**双线性插值**（Bilinear Interpolation）为例，明确采用角点对齐且新网格两边均大于一，目标坐标 (i, j) 对应源坐标

$$x = i\frac{G_h - 1}{G'_h - 1}, \qquad y = j\frac{G_w - 1}{G'_w - 1}. \tag{39.14}$$

令 $r = \lfloor x \rfloor, s = \lfloor y \rfloor, \alpha = x - r, \beta = y - s$，并将邻接下标限制在合法边界，则每个通道独立计算

$$\begin{aligned} P'_{i,j} = {} & (1-\alpha)(1-\beta)P_{r,s} + (1-\alpha)\beta P_{r,s+1} \\ & + \alpha(1-\beta)P_{r+1,s} + \alpha\beta P_{r+1,s+1}. \end{aligned} \tag{39.15}$$

这是一个明确的教学插值约定. 采用像素中心对齐或双三次插值时，坐标和权重都会不同；退化为单行、单列的网格也应显式规定采样位置. 复用已有权重时不能无意混用这些规则. [2]

例如，一个位置通道的源网格为 $\left[\begin{smallmatrix} 0 & 2 \\ 4 & 6 \end{smallmatrix}\right]$. 按角点对齐扩展至 3×3，结果为

$$\begin{bmatrix} 0 & 1 & 2 \\ 2 & 3 & 4 \\ 4 & 5 & 6 \end{bmatrix}. \tag{39.16}$$

中心来自四个角各 $\dfrac{1}{4}$ 的加权和. 分类位置若原为 10，仍单独保留为 10，不能加入这张空间网格. 这里被插值的是学习到的位置向量，不是输入图像像素.

位置插值假设原网格参数能够作为某种平滑空间场重新采样. 形状匹配并不能证明这种平滑假设适合所有学习结果，也不能保证高分辨率下的物体尺度与训练分布一致. 更换补丁尺寸还会改变 W_E 的输入维度，单独插值位置表无法解决这种权重不兼容.

[2] 原始 ViT 在提高分辨率微调时采用位置插值. 插值是参数适配操作；某个实现具体选用的插值核、对齐模式和抗锯齿选项，仍应与其检查点配置一致.

39.4　视觉编码及读出

39.4.1　双向编码层

每层采用**前置层归一化**（Pre-Layer Normalization，Pre-LN）残差结构：

$$U^{(t)} = Z^{(t-1)} + \mathrm{MHA}(\mathrm{LN}(Z^{(t-1)})), \tag{39.17}$$

$$Z^{(t)} = U^{(t)} + \mathrm{MLP}(\mathrm{LN}(U^{(t)})), \qquad t = 1, \dots, T. \tag{39.18}$$

此处归一化沿每个位置的隐藏维度进行，前馈网络逐位置共享参数．多头注意力的查询、键和值均来自同一视觉序列，正常有效补丁之间不施加语言模型的因果掩码．填充位置则需要有效性掩码．注意力和反向传播的通用推导见第五章《注意力机制》，这里展开其视觉含义．

单个注意力头对第 i 个位置计算

$$a_{ij} = \frac{\exp(\frac{q_i^{\top} k_j}{\sqrt{d_h}})}{\sum_{m \in \mathcal{V}} \exp(\frac{q_i^{\top} k_m}{\sqrt{d_h}})}, \qquad o_i = \sum_{j \in \mathcal{V}} a_{ij} v_j, \tag{39.19}$$

$\mathcal{V}$ 为有效位置集合．一个补丁可以依据内容和位置从其他区域收集信息：局部边缘可以与远处轮廓组合，分类位置也可以汇集多个区域．全局可访问只表示存在信息通路，并不保证模型实际利用了需要的区域．

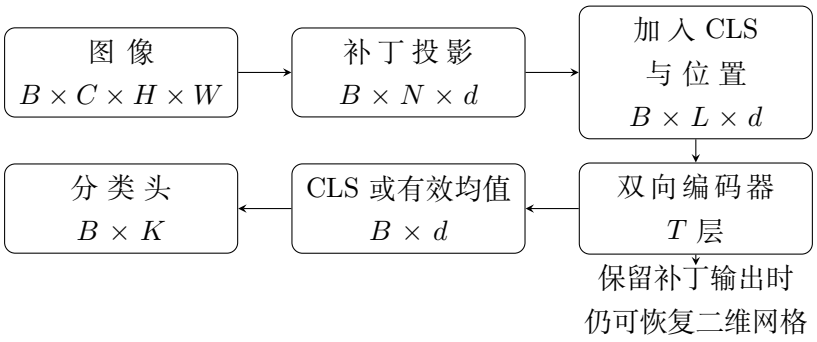


图 39.1: ViT 的表示路径．分类读出压缩空间序列；密集预测可以保留补丁表示．

39.4.2　CLS 及平均池化

在最终归一化后令 $\widehat{Z} = \mathrm{LN}(Z^{(T)})$．CLS 读出采用 $h = \widehat{Z}_0$；**全局平均池化**（Global Average Pooling，GAP）可以采用

$$h = \frac{1}{N_{\mathrm{valid}}} \sum_{n=1}^{N} m_n \widehat{Z}_n, \qquad N_{\mathrm{valid}} = \sum_{n=1}^{N} m_n, \tag{39.20}$$

其中 $m_n \in \{0,1\}$，且 $N_{\text{valid}} > 0$. 前者通过专门位置学习信息汇集，后者对最终补丁表示赋相同读出权重，但这些补丁已经经过内容相关的全局交互，因而不能把 GAP 误解为对原始像素简单取平均.

两者也产生不同的直接梯度路径. 若分类损失对 h 的梯度为 g，GAP 对每个有效最终补丁的直接梯度为 $\dfrac{g}{N_{\text{valid}}}$，CLS 则在读出位置接收直接梯度，再经各层注意力传递至补丁. 结构差异不构成任一读出方式普遍更好的证明，替换预训练模型的读出方式也可能需要重新训练与调整优化设置.

分类头令 $o = hW_C + b_C$，其中 $W_C \in \mathbb{R}^{d\times K}$，并以 $p_k = \dfrac{\exp(o_k)}{\sum_j \exp(o_j)}$ 得到类别概率. 对于单标签监督 y，损失为 $-\log p_y$. 密集定位任务则需要保留 N 个补丁特征，恢复 $G_h \times G_w$ 网格，并通过任务头处理空间分辨率；单个 CLS 无法直接替代这条空间输出路径.

39.5 分辨率及计算预算

39.5.1 参数及运算规模

对每层标准多头注意力，忽略偏置、归一化和逐元素算子，以一次乘加作为一个计数单位，四个投影约需 $4BLd^2$ 次乘加；QK^{T} 与注意力乘 V 合计 $2BL^2d$；前馈网络中间维度为 ρd 时约需 $2BL\rho d^2$. 因此

$$\text{MACs}_{\text{layer}} \approx B\left[(4+2\rho)Ld^2 + 2L^2d\right]. \tag{39.21}$$

若按一次乘法和一次加法各计一次浮点运算，主要项再乘二. 这个计量约定必须与报告一致.

标准注意力的逻辑分数共有 BaL^2 个元素. 若显式以每元素 s 字节存储，单张分数张量占 $sBaL^2$ 字节；训练还包括其他激活、梯度和优化器状态. 融合注意力内核可能不物化整张矩阵，故逻辑规模不是实际峰值显存的等式. 相关实现见高效注意力章节.

固定 d、C 与图像尺寸时，补丁投影的主要乘加量为 $BNFd = BCHWd$，在无重叠规则下与补丁面积无关；但其参数数 $Fd + d$ 会随补丁面积变化. 减小补丁主要增加编码器序列长度，而不是简单地按同一倍数增加所有模块. 这一区分有助于定位成本来源.

39.5.2 规模比较

设图像为 224×224，采用方形补丁与一个 CLS. 下表是由整数公式计算出的结构数量，不是实测显存或运行时间. 忽略 CLS，补丁边长减半使 N 增长四倍，二次

表 39.2: 固定图像尺寸下的序列规模.

P	网格	N	L	每头 L^2
32	7×7	49	50	2500
16	14×14	196	197	38809
8	28×28	784	785	616225

项增长十六倍. 保留 CLS 后比值分别为 $\frac{38809}{2500} \approx 15.52$ 和 $\frac{616225}{38809} \approx 15.88$，不能把十六倍写成含 CLS 情况的精确等式.

以 $d = 768, \rho = 4, B = 1, P = 16$ 为例，一层线性投影与前馈的主要项为

$$12 \times 197 \times 768^2 = 1394343936, \tag{39.22}$$

注意力两次矩阵乘法为

$$2 \times 197^2 \times 768 = 59610624. \tag{39.23}$$

在这一构造配置中，前者明显更大；不能仅因注意力存在二次项，就断定任何分辨率下它都支配运算. 两类项相等约发生在 $L = (2 + \rho)d$，本例为 4608. 这一交点只比较乘加次数，尚未计入带宽、并行度和算子调度.

若固定 $P = 16$ 而把输入变为 384×384，则网格为 24×24，$N = 576, L = 577$，每头逻辑分数为 332929. 二维尺寸同时扩大时，N 按面积增长，注意力二次项按面积的平方增长. 与改变补丁大小不同，这种调整保持补丁投影形状不变，但仍需处理位置参数与视觉尺度分布.

结构预算与质量是不同问题

更细补丁提高空间采样粒度，同时增加编码器序列成本；更高分辨率可能保留输入细节，但也改变内容尺度和位置分布. 两者都不保证分类或定位质量单调提高. 比较质量时应同时固定数据划分、预处理、优化预算和评价口径，并说明哪些变量实际发生了变化.

39.6　归纳偏置及训练边界

39.6.1　卷积先验及全局交互

归纳偏置（Inductive Bias）是模型结构或学习过程对可学习规律的偏好. 卷积通过局部连接和跨位置共享核编码空间先验；在适当边界与步幅条件下，其输出随输入平移而相应平移，这称为**平移等变性**（Translation Equivariance）. 带步幅的卷积仅对与采样网格兼容的平移具有相应性质，不能无条件套用逐像素平移等变的说法.

ViT 的补丁投影同样包含局部性与权重共享，并非没有任何视觉先验. 它的编码器允许补丁全局交互，而绝对位置表可使不同位置具有不同作用. 一像素平移可能改

变多个补丁内部的像素组合，不能简单视为补丁序列的整数置换. 因此，式 (39.12) 的序列置换等变不等于图像平移等变.

原始 ViT 强调了预训练数据规模与迁移效果的关系；后续数据高效视觉 Transformer 研究说明，训练配方、正则化及蒸馏也会显著改变数据利用效率 [221]. 不能把某一配方在某个数据规模下的结果写成“Transformer 天生需要固定数量图像”的定律. 模型容量、目标域和数据增强共同决定泛化条件.

39.6.2 数据增强的目标效应

数据增强（Data Augmentation）从变换分布 $a \sim \mathcal{A}$ 采样图像变换，训练目标相应成为

$$\min_{\theta} \frac{1}{M} \sum_{i=1}^{M} \mathbb{E}_{a\sim\mathcal{A}} \left[\ell(f_\theta(a(I_i)), y_i) \right]. \tag{39.24}$$

随机裁剪和翻转隐含“变换后标签仍有效”的假设. 裁剪移除了唯一目标、镜像改变文字含义或左右具有临床语义时，标签保持假设可能失败. 因此，增强不能仅按强弱选择，还应按任务的语义不变性选择. 验证阶段通常使用固定处理流程，以免随机输入变化掩盖模型差异.

混合增强（Mixup）在线性组合输入时同时组合标签 [222]：

$$\widetilde{I} = \lambda I_i + (1-\lambda) I_j, \quad \widetilde{y} = \lambda y_i + (1-\lambda) y_j, \quad 0 \leqslant \lambda \leqslant 1. \tag{39.25}$$

原方法从 Beta 分布采样 λ. 由于交叉熵对目标标签线性，有

$$-\sum_{k} \widetilde{y}_k \log p_k = \lambda \ell(p, y_i) + (1-\lambda) \ell(p, y_j). \tag{39.26}$$

例如 $\lambda = 0.7$，两类标签混合为 $(0.7, 0.3)$，预测为 $(0.8, 0.2)$，则损失为 $-0.7 \log 0.8 - 0.3 \log 0.2 \approx 0.6390$. 这是对合成输入施加的软监督，不表示混合图像在现实世界中天然具有同样的类别概率.

39.6.3 迁移策略及输入协议

线性探测（Linear Probing）冻结视觉主干，仅训练新的线性分类头；全量微调更新主干与分类头；部分解冻则在两者之间选择可训练层. 冻结主干减少优化器状态，却不自动证明其表示适合新域. 若主干前面还存在需要训练的模块，冻结权重的层仍需向输入传播梯度，这与参数高效微调中的原则相同.

小数据下可从预训练表示与较保守的更新范围开始，依据独立验证选择策略. 层间学习率衰减可以让较浅层更新更小；参数指数移动平均对历次参数进行平滑. 两者以及混合精度、梯度累积的通用机制见训练章节，是否提升当前视觉任务仍需受控比较，不能把配方叠加数目当作有效性的证据.

发布时必须绑定权重、架构配置、图像处理器和标签映射. 预处理中的通道顺序、像素范围、缩放插值、裁剪位置、均值与标准差共同定义输入分布. 若训练使用 $x = \frac{\frac{p}{255} - \mu}{\sigma}$，部署直接把 p 代入同一个归一化式，便不是一个微小误差，而是显著改变输入尺度. 分类头列数正确也不保证标签语义正确：标签索引发生置换时，模型输出的数值可以完全合理，显示的类别却整体错误.

算法 39.1　视觉编码的确定性推理路径

输入： 图像批量、固定预处理协议、补丁与位置网格配置、T 层权重、分类标签映射.

输出： 类别概率，以及按需要保留的补丁表征.

状态： 有效区域、补丁网格 (G_h, G_w)、序列 Z、层计数 t.

1. 按声明的通道、范围、缩放与归一化处理图像；根据边界规则裁剪或填充，并记录有效区域.
2. 由式 (39.1) 确定补丁次序，使用式 (39.3) 或等价卷积投影. 保持权重排列与像素排列一致.
3. 检查位置表的源网格；若目标网格不同，分离 CLS 后按声明的二维规则插值. 若补丁投影尺寸不兼容，终止并报告配置错误.
4. 加入 CLS 与位置嵌入，建立有效位置掩码. 每个有效查询必须至少有一个可见键；不得将全屏蔽行直接送入普通 softmax.
5. 对 $t = 1, \dots, T$ 执行式 (39.18)，保持序列长度和隐藏维度不变；推理时停用训练期随机失活.
6. 按固定的 CLS 或有效均值读出方式产生 h，计算分类分数与概率，并用绑定的标签映射解释输出.

停止条件与不变量： 完成 T 层与分类头后返回；每个补丁的序列位置始终对应同一空间位置. 配置不兼容或无有效输入时显式终止，不静默更换预处理或读出策略.

39.7　视觉模型的评价边界

机制等价与任务有效性需要不同证据. 共享参数后的线性投影和卷积一致性支持算子对应；补丁化后按逆映射恢复像素支持坐标正确；这些都不证明训练后分类准确. 反过来，较高分类准确率也可能掩盖错误预处理被模型偶然补偿的问题. 评价视觉系统还应控制同源图像跨划分泄漏、近重复、类别不平衡与采集域变化.

注意力图显示式 (39.19) 中的内部权重，不直接等于像素的因果重要性. 多头投

影、值向量、残差和后续非线性共同决定输出. 遮挡分析也改变了输入分布，因此应说明遮挡规则和比较对象，而不是把一张热图当作解释有效性的最终结论.

第四十章　多模态建模

一幅图像和描述它的句子可以指向同一场景，却具有不同的数据结构．图像由空间采样构成，文本由离散词元构成；音频具有密集时间采样，视频同时具有空间和时间维度．多模态模型必须先建立可计算的接口，再通过学习使这些接口保留任务所需的关系．两组向量具有相同维数，只说明能够相加或拼接，不能证明它们表达了相同含义．

矩阵反向传播承接第三章《神经网络基础》，空间与时间位置的作用承接第七章《位置表示》．本章围绕三个问题展开：感知数据如何成为序列；跨模态关系通过什么目标学习；模型如何在不泄漏答案的条件下利用这些序列．我们先讨论视觉与文本，再把相同的接口原则推广到音频和视频．图像生成所需的像素或潜变量解码目标与本章的条件文本生成不同，不能由“可以输入图像”推导出“可以输出图像”．

40.1　模态、任务及表示接口

模态（Modality）是信息的观测或表达方式．**视觉语言模型**（Vision-Language Model，VLM）联合利用视觉和语言信息，可以实现匹配、分类、问答或条件生成，但具体能力由输入输出接口和训练目标决定．图文对比模型输出匹配分数，不必包含文本解码器；感知型生成模型读取图像并输出文本，也不必包含图像生成器．

本章的张量与学习约定

批量大小为 B．图像空间尺寸可被给定块大小整除，不能整除时的补齐和有效位置由处理器显式记录．视觉表示与文本表示均为实值张量；对比学习例题中每个图像和同批同序号文本构成一个正对，归一化前特征非零，温度 $\tau > 0$．条件生成部分只对指定文本答案计损失，输入图像及提示是已知条件．构造数例用于数学说明，不代表模型性能测量．

表 40.1: 本章主要符号．矩阵推导未写批量轴时，对每个样本分别应用．

符号	含义与形状
I	图像批量，$\mathbb{R}^{B\times C_{\text{img}}\times H\times W}$.
P_h, P_w, N_v	块高、块宽和视觉词元数，$N_v = (\frac{H}{P_h})(\frac{W}{P_w})$.
X_v, Z_v	编码特征 $B \times N_v \times d_v$，连接后特征 $B \times M \times d$.
d_v, d, d_c	视觉隐藏维、语言隐藏维和对比表示维，三者可不同.
u_i, v_j	第 i 幅图像和第 j 条文本的单位对比向量，均在 $\mathbb{R}^{d_c}$.
S, R, C	对比逻辑值、按行与按列归一化的概率矩阵，均为 $B \times B$.
Q, K, V, A	重采样查询 $M \times d_k$、键 $N_v \times d_k$、值 $N_v \times d$、权重 $M \times N_v$.
m_j	目标位置 j 是否参与损失的二值掩码，与可见性掩码不同.
f_s, h, F	音频采样率（样本/秒）、声学窗口步长（样本）、视频选取帧数.

视觉编码器通常产生局部特征序列；图文检索还会通过池化等操作得到全局向量. 全局向量适合快速比较整幅图像与文本的关系，局部序列则保留供计数、位置和文字识别使用的空间信息. 只保留一条全局表示，不等于原图全部细节都被可恢复地编码.

40.2　视觉词元化

40.2.1　切块、投影及空间位置

补丁（Patch）是视觉序列中的局部输入单元. 把每个 $P_h \times P_w$ 块展平为 $x_n \in \mathbb{R}^{C_{\text{img}}P_hP_w}$，使用共享矩阵

$$e_n = x_n W_e + b_e, \qquad W_e \in \mathbb{R}^{C_{\text{img}}P_hP_w\times d_v} \tag{40.1}$$

映射到隐藏维. 卷积核大小与步幅均为块大小、无重叠且采用相同展平顺序时，这与一次卷积后展平空间轴等价. 块投影建立局部线性组合，后续编码器使各位置交互. **视觉 Transformer**（Vision Transformer，ViT）采用块序列进行图像建模，是这一表示路线的代表 [220].

二维位置必须以位置表示、相对偏置或结构化计算等方式进入模型；展平顺序本身只建立数组地址，并不会使无位置自注意力自动理解上下左右. 多图输入还需区分图像编号与图内坐标. 高分辨率分块处理时，局部裁剪坐标应能映射回全图，否则模型难以判断不同裁剪中的对象是否为同一对象.

若 $H = W = 224$、$P_h = P_w = 16$，则 $N_v = 14 \times 14 = 196$. 将两边都增大到 448 而保持块大小，得到 784 个块，是原来的 4 倍. 若编码器采用完整自注意力，其位置对数从 196^2 增至 784^2，为原来的 16 倍；这是注意力矩阵规模比较，不是端到端运行时间的必然倍数. 投影、前馈、内存访问和实现方式同样影响耗时.

40.2.2　分辨率及有效信息

提高分辨率可以保留小字和局部结构，但缩放插值不会创造原始图像没有的信息．固定缩放可能破坏长宽比；裁剪可能移除问题所需区域；把整页文档压到很小尺寸可能保留布局而丢掉数字．预处理应记录缩放、裁剪、补齐和坐标变换，使预测的区域能够正确返回原图．

动态分辨率（Dynamic Resolution）根据输入尺寸或内容调整视觉序列长度．它要求批量中的有效长度和补齐位置可追踪，并把视觉词元预算纳入请求限制．可变长度不是无上限长度；多图、多页和视频需要在空间细节与覆盖范围之间分配预算．对于需要精确读取局部内容的问题，保留相关高分辨率区域可能比均匀增加所有帧的细节更合适，但选择区域的错误也必须计入系统误差．

40.3　视觉及文本的对比对齐

40.3.1　双向匹配目标

跨模态对齐（Cross-modal Alignment）通过训练使不同模态的相关表示在任务所需意义上对应．对齐可以表现为全局匹配、局部区域与词语对应，或使语言模型能够利用视觉条件生成；这些目标不能互相替代．

对比语言图像预训练（Contrastive Language-Image Pre-training，CLIP）学习图像和文本的共享匹配空间 [223]．下面给出其双向对比目标的一般形式．设编码器及投影输出非零向量 a_i, b_j，归一化为

$$u_i = \frac{a_i}{\|a_i\|_2}, \quad v_j = \frac{b_j}{\|b_j\|_2}, \quad S_{ij} = \frac{u_i^{\top} v_j}{\tau}. \tag{40.2}$$

S_{ij} 是相似度经温度缩放后的逻辑值．对图像 i 在批内文本中分类，对文本 j 在批内图像中分类，分别得到

$$R_{ij} = \frac{e^{S_{ij}}}{\sum_k e^{S_{ik}}}, \qquad C_{ij} = \frac{e^{S_{ij}}}{\sum_k e^{S_{kj}}}. \tag{40.3}$$

矩阵 R 各行和为 1，矩阵 C 各列和为 1；二者通常不同．单正对假设下，对角线为目标，平均损失为

$$\mathcal{L}_{v\to t} = -\frac{1}{B}\sum_i \log R_{ii}, \tag{40.4}$$

$$\mathcal{L}_{t\to v} = -\frac{1}{B}\sum_j \log C_{jj}, \tag{40.5}$$

$$\mathcal{L}_{\mathrm{con}} = \frac{1}{2}(\mathcal{L}_{v\to t} + \mathcal{L}_{t\to v}). \tag{40.6}$$

两个方向分别约束“一幅图像应匹配哪条文本”和“一条文本应匹配哪幅图像”．仅优化一个方向，不会自动对另一个方向施加相同的列竞争关系．

40.3.2　编码器梯度

对一行的损失展开 $-S_{ii}+\log\sum_k e^{S_{ik}}$，对 S_{ij} 求导得到 $R_{ij}-\delta_{ij}$；按列展开同理得到 $C_{ij}-\delta_{ij}$. 因此

$$G_{ij}:=\frac{\partial\mathcal{L}_{\text{con}}}{\partial S_{ij}}=\frac{R_{ij}+C_{ij}-2\delta_{ij}}{2B}. \tag{40.7}$$

将 $U,V_c\in\mathbb{R}^{B\times d_c}$ 定义为各 $u_i^{\mathsf{T}},v_i^{\mathsf{T}}$ 按行排列的矩阵，有 $S=\dfrac{UV_c^{\mathsf{T}}}{\tau}$. 利用微分

$$\mathrm{d}S=\frac{\mathrm{d}U\,V_c^{\mathsf{T}}+U\,\mathrm{d}V_c^{\mathsf{T}}}{\tau}, \tag{40.8}$$

代入 $\mathrm{d}\mathcal{L}=\operatorname{tr}(G^{\mathsf{T}}\mathrm{d}S)$ 并整理迹，得到

$$\nabla_U\mathcal{L}=\frac{GV_c}{\tau},\qquad \nabla_{V_c}\mathcal{L}=\frac{G^{\mathsf{T}}U}{\tau}. \tag{40.9}$$

每幅图像的梯度不仅来自其配对文本，还来自所有参与分母的候选；“负样本”是目标函数中的竞争关系，而不是只在正对损失之外额外添加一个惩罚项.

归一化还有一层雅可比. 令 $r=\|a\|_2$、$u=\dfrac{a}{r}$，则

$$\mathrm{d}r=u^{\mathsf{T}}\mathrm{d}a, \tag{40.10}$$

$$\mathrm{d}u=\frac{\mathrm{d}a}{r}-\frac{a}{r^2}\mathrm{d}r=\frac{1}{r}(I_{d_c}-uu^{\mathsf{T}})\mathrm{d}a, \tag{40.11}$$

$$\nabla_a\mathcal{L}=\frac{1}{r}(I_{d_c}-uu^{\mathsf{T}})\nabla_u\mathcal{L}. \tag{40.12}$$

它去除纯径向分量，使一阶更新作用于方向；范数很小时梯度可能很大. 实现中使用带 ε 的稳定归一化时，应按实际定义理解分段或近似梯度，不能假定与非零范数公式处处相同.

若使用可学习缩放 $\alpha=e^s=\dfrac{1}{\tau}$，则 $S_{ij}=\alpha u_i^{\mathsf{T}}v_j$，因此

$$\frac{\partial\mathcal{L}}{\partial s}=\sum_{ij}G_{ij}S_{ij}. \tag{40.13}$$

缩放增大会使分布更集中，但错误配对或近似重复样本也可能被更强地惩罚. 温度影响优化几何，不能解释为模型已经校准的事实置信度.

40.3.3　双样本对比损失

取 $B=2,d_c=2$，图像和文本单位向量均为 $u_1=v_1=(1,0)^{\mathsf{T}}$、$u_2=v_2=(0,1)^{\mathsf{T}}$，取 $\tau=1$. 则

$$S=\begin{bmatrix}1&0\\0&1\end{bmatrix},\qquad R=C=\begin{bmatrix}p&1-p\\1-p&p\end{bmatrix},\quad p=\frac{e}{e+1}\approx0.731059. \tag{40.14}$$

两方向损失均为 $-\log p = \log(1+e^{-1}) \approx 0.313262$，故平均损失不变．式 40.7 给出

$$G = \begin{bmatrix} -0.134471 & 0.134471 \\ 0.134471 & -0.134471 \end{bmatrix}. \tag{40.15}$$

由于 V_c 为单位矩阵，$\nabla_U \mathcal{L} = G$．若 $a_1 = u_1$，式 40.12 去掉第一坐标方向，得到 $\nabla_{a_1}\mathcal{L} = (0, 0.134471)^{\mathsf{T}}$．梯度下降会让第一幅图像表示朝远离第二条文本的方向移动；这不是把所有分量机械地向正对增加.对数缩放梯度为两个对角项之和，即 -0.268942，在此构造数据上下降更新会增大缩放．若批内都是错误标注，则这种扩大间隔也可能扩大错误.

40.3.4　负样本、多正对及语义边界

批内负例（In-Batch Negative）把同批其他配对的样本作为竞争候选，计算方便但依赖标签含义．两幅相同图像、同一场景的不同描述，或两条都正确的简短字幕，可能被误当负样本．这类**假负例**（False Negative）会使目标要求语义相近项彼此远离．简单扩大批量不保证监督更干净.

若图像 i 的合法文本集合为 $\mathcal{P}_i$，可定义归一化多正对目标 $T_{ij} = \dfrac{1}{|\mathcal{P}_i|}$ $(j \in \mathcal{P}_i)$，使用 $-\sum_{j} T_{ij} \log R_{ij}$；另一选择是 $-\log \sum_{j \in \mathcal{P}_i} R_{ij}$．前者鼓励所有正对按目标分布获得概率，后者只要求正对集合获得总质量，可能由某一个容易正对承担．两者不是相同目标，应根据标注语义选择.

困难负样本应在保持真实不匹配的前提下增加辨别难度，例如对象相同但关系不同．只利用背景、文件水印或文本模板区分配对，会形成捷径．图像增强也应保留文本所描述的事实：随机裁剪如果裁掉唯一红色物体，原字幕就可能不再成立．对比对齐能够组织全局匹配空间，却不保证精确计数、否定、空间关系或细粒度字符都已学会.

跨设备汇集表示扩大候选集合时，需明确远端表示是否参与梯度和损失如何归一化．将远端表示当常量与使用可微汇集产生不同梯度；不能仅因前向分数矩阵相同就声称训练等价．这里的分布式实现应与全局批量目标保持一致.

40.4　多模态连接器

40.4.1　逐位置投影

连接器（Connector）把感知编码器输出转换为语言侧可用的表示．最简单的**投影器**（Projector）执行

$$Z = X_v W_p + \mathbf{1} b_p^{\mathsf{T}}, \qquad W_p \in \mathbb{R}^{d_v \times d}, \tag{40.16}$$

对每个位置共享参数，将 $N_v \times d_v$ 变为 $N_v \times d$. 它改变特征维数，不改变位置数. 非线性多层投影器可以增加表示变换能力，但只要逐位置应用，仍不直接完成跨位置压缩.

设 $D_Z = \dfrac{\partial \mathcal{L}}{\partial Z}$，则

$$\nabla_{W_p}\mathcal{L} = X_v^{\mathsf{T}} D_Z, \quad \nabla_{X_v}\mathcal{L} = D_Z W_p^{\mathsf{T}}, \quad \nabla_{b_p}\mathcal{L} = \sum_n (D_Z)^{\mathsf{T}}_{n,:}. \tag{40.17}$$

批量情况下对样本轴继续求和. 若视觉编码器被冻结，可以不更新其参数；投影器仍需接收从语言损失传来的 D_Z. 训练数据决定它学习什么视觉信息，而矩阵形状本身无法决定其语义.

40.4.2 查询重采样

重采样器（Resampler）使用较少输出位置汇聚较长视觉序列. 对单个样本，一层查询读取可写为

$$K = X_v W_K, \quad V = X_v W_V, \tag{40.18}$$

$$A = \mathrm{softmax}_{\mathrm{row}}\left(\frac{QK^{\mathsf{T}}}{\sqrt{d_k}}\right), \tag{40.19}$$

$$Z_v = AV. \tag{40.20}$$

$Q \in \mathbb{R}^{M \times d_k}$ 为可训练查询，输出长度为 M. 每一输出是值向量的加权组合，不是选择某一个原始块. 固定查询也不意味着权重固定，因为 A 随输入键而改变. 查询间可加入自注意力与前馈层；这些设计改变表达能力，但不改变“输出长度受控”的接口目的.

查询式连接器有不同具体实现. BLIP-2 的**查询变换器**（Querying Transformer，Q-Former）用可学习查询从冻结图像编码器提取特征，并通过表示学习与生成学习两个阶段连接语言模型 [224]；Flamingo 使用重采样与语言侧视觉读取模块 [225]. 本章式 40.20 只抽取共有的单层读取原理，不把它冒充这些完整模型. [1]

对于固定注意力矩阵 A，若 $M < N_v$，其秩至多为 M，存在非零方向 ΔV 使 $A\Delta V = 0$. 所以仅从 AV 不能恢复任意 V. 完整重采样器的 A 也依赖输入，这个固定矩阵论证不能直接穷尽其非线性行为，但已说明“较少查询输出”没有自动的无损保证. 任务所需的局部信息是否保留，必须依靠训练目标和细粒度评估.

[1] 投影、重采样与前缀是不同维度的设计选择：前两者描述表示变换，前缀描述语言模型如何接收结果. 一个系统可以同时使用三者，因此不宜把它们列为互斥的三个模型类别.

40.4.3　重采样梯度

取 $N_v=2, M=1, d_k=1, d=2$，$Q=[1]$，$K=[0,\log 3]^{\mathsf{T}}$，值为

$$V=\begin{bmatrix}2 & 0\\ 0 & 4\end{bmatrix}. \tag{40.21}$$

由于缩放因子为 1，权重 $A=[\frac{1}{4},\frac{3}{4}]$，输出为 $Z_v=[\frac{1}{2},3]$. 它既不是两个值的简单平均 $[1,2]$，也不是原始某一行.

为说明梯度，取 $\mathcal{L}=\frac{1}{2}\|Z_v-Z^{*}\|^2$，$Z^{*}=[1,2]$，则 $D_Z=[-\frac{1}{2},1]$. 首先

$$D_V=A^{\mathsf{T}}D_Z=\begin{bmatrix}-\frac{1}{8} & \frac{1}{4}\\ -\frac{3}{8} & \frac{3}{4}\end{bmatrix}, \qquad D_A=D_Z V^{\mathsf{T}}=[-1,4]. \tag{40.22}$$

对注意力逻辑值 $r=[0,\log 3]$，Softmax 导数给出

$$(D_r)_i=A_i\left((D_A)_i-\sum_j A_j(D_A)_j\right). \tag{40.23}$$

加权和为 $-\frac{1}{4}+3=\frac{11}{4}$，故 $D_r=[-\frac{15}{16},\frac{15}{16}]$. 进一步有

$$D_Q=D_rK=\frac{15}{16}\log 3, \qquad D_K=D_r^{\mathsf{T}}Q=[-\frac{15}{16},\frac{15}{16}]^{\mathsf{T}}. \tag{40.24}$$

损失通过值路径和键路径同时传回视觉表示. 冻结视觉编码器只停止其参数更新，不代表连接器无需学习注意力分配.

40.5　融合位置及计算代价

40.5.1　前缀拼接

前缀融合（Prefix Fusion）把 M 个视觉向量与 N_t 个文本嵌入沿序列轴拼接，得到 $B\times(M+N_t)\times d$. 视觉向量是连续条件，不要求具有文本词表 ID. 视觉占位符可以指示插入位置，但插入后真实位置数必须由处理器和模型契约一致确定.

若所有位置进入密集自注意力，长度平方项相对纯文本增加

$$(N_t+M)^2-N_t^2=2N_tM+M^2. \tag{40.25}$$

例如 $N_t=512$，保留 196 个视觉位置时增加 239120 个位置对；压缩到 32 个视觉位置时增加 33792 个位置对. 这里比较的是位置交互量，尚未计编码器和重采样开销，也不等于显存或延迟按同样比例下降.

40.5.2　交叉注意力及交错融合

交叉注意力融合（Cross-attention Fusion）让语言隐藏状态作为查询，独立视觉记忆作为键和值．单层交互量与 N_tM 成正比，文本自注意力仍有自身的 N_t^2 项．它保留模态边界，可在指定语言层读取视觉信息，但需要新增层、参数及相应推理实现．推理时视觉键值可以按层缓存；是否划算取决于读入层数、视觉长度和生成长度．

图文交错序列则还需明确每段文本可以读取哪些图像．全前缀方案允许后续文本读取全部已知视觉条件；含时间顺序的交错数据可以只开放此前可见媒体．所谓“早期”或“晚期”融合只有相对于具体网络边界才有意义：在两个独立编码器之后拼接，仍可能发生在语言主干的第一层之前．

统一序列建模进一步把多模态输入放入共同主干，但共享主干不意味着共享输出分布．生成离散声学或视觉词元需要相应词表与解码器；生成连续潜变量可能需要不同损失．架构应说明编码、融合与输出头三条边界，不能只画一个接收所有模态的方框．

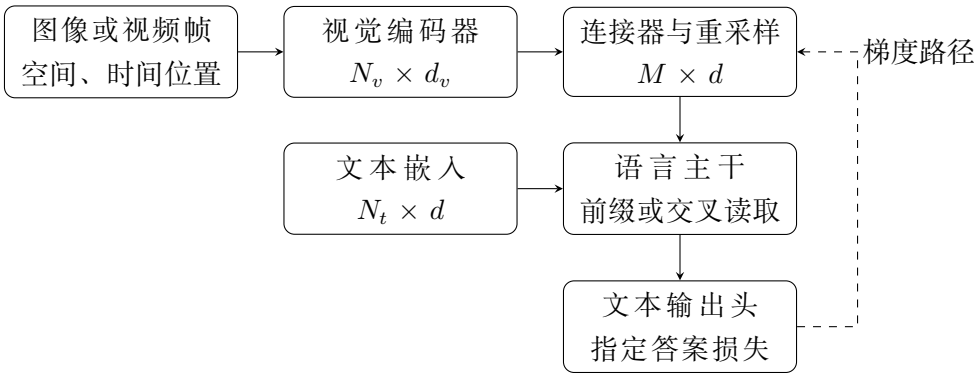


图 40.1: 感知型视觉语言模型的模块边界．冻结参数不会自动切断对上游可训练输入所需的梯度．

40.6　监督区域及分阶段训练

40.6.1　可见性掩码及监督掩码

注意力掩码（Attention Mask）决定一个位置可以读取哪些其他位置；**损失掩码**（Loss Mask）决定哪些目标参与优化．视觉与用户提示可以不承担预测标签，却仍是后续答案的有效条件．把“不监督”实现为“不可读取”，会移除模型完成任务所需的信息．

设拼接序列为 $x_1, \ldots, x_L$，其中前 M 个是视觉向量，后面包含提示和答案．语言模型在位置 $j-1$ 输出分布 q_{j-1}，预测目标位置 j 的文本词元 y_j．令 $m_j \in \{0,1\}$ 表

示该目标是否受监督，$N_s = \sum_{j=2}^{L} m_j > 0$，则

$$\mathcal{L}_{\text{gen}} = -\frac{1}{N_s} \sum_{j=2}^{L} m_j \log q_{j-1}(y_j). \tag{40.26}$$

掩码附着于目标位置，再与前一位置的逻辑值配对．有的库在内部执行移位，有的接口要求外部移位；无论接口如何，数学含义都必须与式 40.26 一致．重复移位和完全没有移位都会使监督错位．

因果可见性的一种简单定义是 $A_{ij}^{\text{mask}} = 0$ 当 $j \leqslant i$ 且位置有效，否则为 $-\infty$．它加到注意力逻辑值中，防止看到未来答案．视觉前缀内部也可以采用双向可见，但不能让它读取未来答案后再作为答案的条件，否则会形成绕过因果掩码的信息泄漏．

取 $M = 2$，随后是起始标记、一个提示词元、两个答案词元及结束标记．位置与标签关系如下：

表 40.2: 目标掩码与上下文有效性的区别．位置 5 的标签由位置 4 的输出预测．

位置	内容	该位置作为上下文	该位置的目标标签
1	V_1	有效	忽略
2	V_2	有效	忽略
3	起始标记	有效	忽略
4	提示 P	有效	忽略
5	答案 A_1	有效	A_1
6	答案 A_2	有效	A_2
7	结束标记	有效	结束标记

若三个目标的正确词元概率分别为 0.5、0.25 和 0.8，则

$$\mathcal{L}_{\text{gen}} = -\frac{\log 0.5 + \log 0.25 + \log 0.8}{3} = \frac{\log 10}{3} \approx 0.767528. \tag{40.27}$$

起始位置、视觉位置和提示位置没有对应的直接目标损失；但是 A_1 的概率由提示位置 4 的输出预测，因此不能把位置 4 的输出也一并忽略．若将视觉向量加入后仍沿用原文本的绝对标签位置，答案标签就会错位两个位置．

对于批量中的不同回答长度，按所有受监督词元求平均与先按每样本求平均再对样本平均不同．前者给予长答案更大权重，后者使每个样本总权重相同．应在训练契约中明确选择；全掩码样本的 $N_s = 0$ 需要单独处理，不能产生未定义除法，也不能通过加入虚假标签凑数．

40.6.2　跨模态梯度传播

设视觉条件为 $Z_v = f_\phi(I)$，语言模型为 g_θ，则

$$\nabla_\phi \mathcal{L} = \left(\frac{\partial Z_v}{\partial \phi}\right)^{\mathsf{T}} \left(\frac{\partial g_\theta}{\partial Z_v}\right)^{\mathsf{T}} \nabla_{g_\theta} \mathcal{L}. \tag{40.28}$$

即使只在答案位置计算损失，答案仍可通过注意力依赖视觉条件，从而产生 $\nabla_\phi \mathcal{L}$. 冻结语言参数 θ 意味着不更新这些参数，不意味着把 g_θ 当作对输入常量的无梯度过程. 如果连接器仍需训练，必须保留对其输入输出所需的反向路径. 相反，若视觉编码器冻结且其输入也不需要梯度，可以缓存或停止该编码器内部的反向计算，但连接器部分仍保留计算图.

> **冻结参数与停止梯度必须分开**
>
> 参数是否更新、输入是否参与反向、位置是否被读取、目标是否计入损失，是四个不同问题. 它们分别决定优化范围、梯度路径、信息流与监督区域；混用这四种控制，会使张量形状正确而学习机制错误.

40.6.3　指令学习接口

先冻结视觉编码器和语言模型、只训练连接器，可以利用已有表示并降低待优化参数规模. 该阶段通常让视觉条件支持描述或其他明确文本目标. 它不保证语言主干已经能完成复杂视觉推理，也不能补回冻结编码器从未保留的细节.

随后可用交错图文或多模态指令数据训练连接器与部分主干. **视觉指令微调**（Visual Instruction Tuning）以图像、问题和答案组织监督，使模型学习遵循视觉相关任务请求；LLaVA 工作是这一训练路线的代表 [13]. 联合解冻增强调整能力，但也会增加语言能力遗忘、视觉表示漂移及对合成描述偏差的拟合. 是否解冻、解冻多少层和是否保留纯文本数据，应由目标能力与评估决定，而非固定套用阶段表.

不同目标可以组合为

$$\mathcal{L} = \lambda_c \mathcal{L}_{\text{con}} + \lambda_g \mathcal{L}_{\text{gen}} + \lambda_m \mathcal{L}_{\text{match}}, \qquad \lambda_c, \lambda_g, \lambda_m \geqslant 0, \tag{40.29}$$

其中匹配损失判断一对图文是否一致. 各项的监督区域和梯度可达参数可能不同，系数只有在损失归一化方式已明确时才有可比较意义. 若对比表示读取了其配对文本再参与匹配，就可能用标签捷径完成任务；若生成目标的视觉查询读取未来答案，也会产生泄漏. 多任务训练需要为每个目标分别定义信息可见性.

训练与部署还必须绑定同一组处理器规则，包括图像归一化、裁剪与块序、视觉占位符、文本分词器、会话模板以及特殊标记. 处理器不是可任意替换的文件读取工具，而是模型输入分布的一部分. 编码器、连接器、语言主干和输出解码器分别具有版本及适用条件，组合制品应保存完整依赖关系.

算法 40.1　多模态条件生成的训练步骤

输入为媒体、提示、答案及其长度和坐标元数据；输出为损失和本阶段允许更新的参数梯度.

1. 依据固定版本处理器解码媒体，记录尺寸、裁剪、时间戳与有效位置；按预算选择媒体片段，保持答案所需监督与输入一致.
2. 计算感知特征，经投影及可选重采样得到语言侧表示，保存每个媒体对象到输出位置区间的映射.
3. 构造文本嵌入，并在约定位置插入媒体表示；生成独立的上下文有效性、因果可见性和目标损失掩码.
4. 只执行一次目标移位，将目标位置 j 对齐到输出位置 $j-1$；无有效目标的样本按契约跳过或报数据错误.
5. 前向计算答案分布，按选定的词元或样本归一化计算损失；若有辅助对比目标，使用其单独的配对和可见性规则.
6. 沿需要的输入梯度路径反向，只更新本阶段开放的参数. 记录有效媒体与文本长度、监督词元数和各目标损失.
7. 处理完当前批次后结束此步；按既定训练步数、数据轮次与独立验证准则控制整个训练过程. 阶段切换时保存模型及处理器契约.

不变量是媒体位置映射与标签长度始终一致，所有受监督答案只依赖允许的条件，冻结语言权重不阻断连接器需要的梯度.

40.7　音频表示及时间压缩

40.7.1　波形、声学特征及文本

采样率（Sampling Rate）是单位时间的离散样本数. 若单通道波形以 f_s 样本/秒记录 T_a 秒，样本数约为 f_sT_a. 把长波形逐样本作为语言序列通常过长，因此声学编码器通过窗口特征或学习的下采样形成较短表示.

一种常见表示来自**短时傅里叶变换**（Short-Time Fourier Transform，STFT）. 设窗口长 L_w、步长 h、窗函数 $w[n]$，不补齐边界时

$$X[m,k]=\sum_{n=0}^{L_w-1}x[mh+n]w[n]e^{-\frac{2\pi ikn}{L_w}}. \tag{40.30}$$

频率滤波和对数变换可形成对数梅尔谱等特征；这些操作聚合频谱信息，并不与原波形一一可逆. 窗口宽度决定局部时间范围，步长决定帧密度. 语音识别模型利用声学

条件预测文本，Whisper 工作展示了基于大规模语音文本监督的这一任务路线 [226]；语音合成则需产生可解码的声学表示，二者的输出目标不同.

若 $f_s = 16000$、时长 2 秒，则有 32000 个样本. 取 $L_w = 400$、$h = 160$，对应 25 毫秒窗口和 10 毫秒步长，不补齐时窗口数为

$$N_a = 1 + \left\lfloor \frac{32000 - 400}{160} \right\rfloor = 198. \tag{40.31}$$

编码器若再以步幅 2 并按“奇数补齐一帧”的规则下采样一次，则输出 99 个声学位置. 这里明确了边界规则；中心补齐或不同卷积核会得到不同长度，不能只用“每秒约 100 帧”替代精确形状契约.

40.7.2 离散声学词元及信息取舍

声学词元（Acoustic Token）可以指连续声学表示位置，也可以指音频编解码器产生的离散码. 使用时应说明类型. 离散码序列可由自回归模型预测，再由声学解码器恢复波形；码率、帧率和每帧码本数共同决定序列量与重建质量. 语言文本只表示部分语音内容，通常不完整保留音色、韵律和环境声音.

因此，先转写再交给语言模型的级联系统便于复用文本能力，但会丢失非语言线索并传播转写错误；直接声学融合有机会保留这些信息，却要求相应训练数据与时间处理. 对于语音理解，应分别评估语义、说话人、噪声和事件；对于语音输出，还需评估时延、可懂度、音色一致性及授权范围. 多种输入模态并不自动赋予所有输出模态的能力.

40.8 视频采样、同步及流式因果性

40.8.1 帧预算及观测边界

视频可表示为带时间戳的图像序列 $\{(I_f, t_f)\}_{f=1}^{F}$. 逐帧编码后，未压缩视觉长度约为 FN_v；每帧重采样到 M 个位置时为 FM，全视频联合重采样则可进一步限制总长度. 不同策略分别保留空间与时间信息，不能只看输出形状相同就视为等价.

设均匀采样间隔为 Δt. 在忽略端点、事件持续 $\delta < \Delta t$ 且事件起点相位均匀分布的构造模型下，至少一个采样时刻落入事件的概率为

$$p_{\text{hit}} = \frac{\delta}{\Delta t}. \tag{40.32}$$

例如每秒取 2 帧，间隔 0.5 秒，持续 0.1 秒的事件命中概率仅为 0.2. 这是采样相位假设下的几何概率，不是视觉模型召回率；即使命中帧，模型仍可能识别错误. 没有任何采样帧记录的短暂动作，后续更大语言模型也不能从缺失观测中可靠恢复.

周期采样还可能使快速重复动作混淆．任务驱动采样可以增加变化处的帧密度，但事件检测器会引入自己的漏检．需要长时间范围与局部细节时，可先粗采样定位候选区间，再保留原时间戳细采样；粗阶段没有覆盖的事件仍可能被遗漏．视频不是一组无序照片，帧顺序与间隔都是输入含义的一部分．

40.8.2　声画同步及时钟偏差

时间对齐（Temporal Alignment）把不同模态的观测映射到共同时间轴．数组下标不能直接互相比较：音频第 100 个窗口与视频第 100 帧通常发生在不同时间．可将某音频设备的时间 t_a 映射为参考时间

$$t_{\mathrm{ref}} = \alpha t_a + \beta, \tag{40.33}$$

其中 β 表示起始偏移，α 表示时钟尺度误差．恒定偏移只需平移；时钟漂移会随时长积累，不能仅在开头对齐一次．

例如假定音频时钟每秒多计 10^{-4} 秒，经过 600 秒会产生 0.06 秒偏差．若还有 0.08 秒起始偏移，末端未校正偏差为 0.14 秒．这个构造例子说明短片段可忽略的误差在长会话中可能改变事件对应关系．时间戳应说明是窗口起点、中心还是终点；窗口中心只能定位代表时间，不表示编码器从未读取其后的样本．

对视频帧 t_f 与音频窗口代表时间 t_m，可按 $|t_f - (\alpha t_m + \beta)| \leqslant \epsilon_t$ 建立候选对应关系．容差 ϵ_t 的选择取决于任务分辨率和采样精度；容差过宽会混合相邻事件，过窄则出现无匹配．对应关系可为多对多，不能强制每帧恰好配一个音频位置．

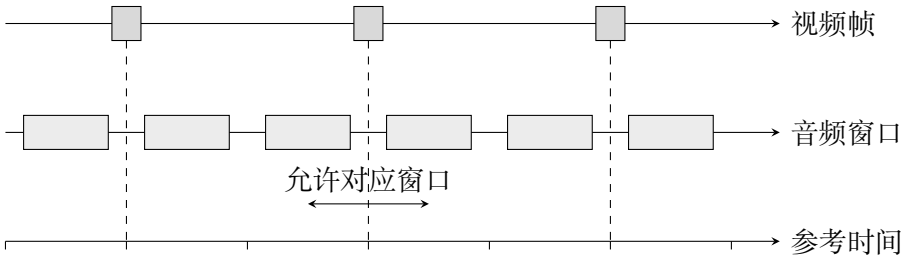


图 40.2: 不同模态通过时间区间对应，而非按数组位置一一配对．图为采样关系示意，不代表实际媒体．

40.8.3　流式输入的可用时刻

流式因果性（Streaming Causality）要求某时刻输出只依赖当时已经到达并被允许使用的数据．若一个声学特征来自窗口 $[t, t + 25\mathrm{ms}]$，它最早在窗口结束后才完整可用；使用中心时间戳不消除这 12.5 毫秒的未来等待．双向声学编码器还可能读取后续窗口，因此训练时的注意力范围决定真实前瞻量．

视频与音频具有不同到达抖动．融合层可以等待一个有限同步窗口，再对尚未到达模态标记缺失；无限等待则破坏服务时延，直接沿用旧帧则可能错误绑定到新语音．

背压（Backpressure）让下游处理能力约束上游发送或缓存速度，避免媒体队列持续增长．丢帧、降采样或分段策略应保存时间空洞，不能把剩余序列重新编号后伪装成连续观测．

流式输出还需区分暂定转写与最终片段，处理用户打断、输出取消和语音文本同步．多模态模型输出操作建议时，感知置信度不能代替执行授权；对视觉界面或物理环境的理解错误应在受控执行边界被拦截．预测未来状态的模型也不能仅凭生成画面的逼真程度成为可靠控制器．

40.9 模型能力及制品一致性

40.9.1 形状语义一致性

形状检查可证明编码器、连接器和语言模型能够连接；监督位置检查可证明学习目标没有明显错位；二者都不能证明视觉理解已学成．随机初始化模型随图像变化而改变逻辑值，只说明图像存在计算路径．输出变化是否与正确视觉事实对应，需要有标注、可控制的任务样本．

可构造只改变图像中一个属性而保持问题不变的样本对，检查答案是否按该属性改变；对计数任务改变对象数量，对空间任务交换对象位置，对文字读取任务替换局部字符．还应使用遮去图像、打乱帧序和错配音频的对照．如果去掉图像后仍答对，可能是问题本来不依赖图像，也可能是模型利用了语言先验；需要通过平衡样本区分这两种解释．

评估应分别覆盖文字识别、图表、细粒度属性、空间关系、多图指代、时间定位与声画对应．证据区域和时间区间有助于判断答案依据，但模型输出一个框并不能自动证明框内支持结论．图文对比匹配高分与条件生成流畅同样不能替代这些任务指标．

40.9.2 依赖及能力边界

模型制品应保存感知编码器、连接器、语言主干、输出解码器、预处理器、分词器、模板、精度与后端配置．多组件许可和使用条件也应逐一核对，不能把语言主干的许可自动推广到视觉编码器、训练数据和整个组合模型．本章不根据品牌名称给出开放程度或能力排序．

服务预算需同时记录分辨率、图像数、帧率、媒体时长、压缩前后视觉及声学长度、文本长度和生成长度．高分辨率、多页与长视频不是同一种压力来源；入口应在解码前后都控制资源，避免压缩文件体积小却解码后极大的输入耗尽服务资源．

媒体中的文字和声音仍属于外部数据，不能改变系统权限．跨模态提示注入、身份或声音冒用、来源不明素材与敏感元数据分别需要相应的数据和执行边界．输出图像、音频和视频的系统还需保留来源与生成记录，并明确允许的使用范围．感知理解、内容生成和行动执行具有不同的错误后果，应该分别验收．

第四十一章　扩散模型的概率原理

自回归模型沿序列位置逐步生成，扩散模型沿噪声尺度逐步生成. **扩散模型**（Diffusion Model）先规定一个容易采样的扰动过程，再学习与其对应的逆向生成过程. 它的基础不是“把模糊图像变清晰”的视觉直觉，而是一个潜变量概率模型：观测数据被连接到一系列带噪潜变量，训练通过可计算的条件分布约束反向转移.

本章以**去噪扩散概率模型**（Denoising Diffusion Probabilistic Model，DDPM）为主要构造 [227]，逐步推导前向边缘、条件后验和训练目标，并讨论不同输出参数化与采样路径. 第四十二章《条件视觉生成》再讨论文本条件、潜空间、网络主干和视觉生成系统，避免将概率模型、网络结构与应用条件混为一体.

41.1　随机变量及前向链

41.1.1　符号及假设

设数据 $x_0 \in \mathbb{R}^D$ 服从数据分布 q_{data}. 图像可以按通道和空间位置展平为 D 维向量；所有推导逐元素成立，不要求网络实际展平图像. 带噪变量 $x_1, \dots, x_T$ 与数据同形，$t \in \{1, \dots, T\}$ 表示离散噪声步，$t = 0$ 专指干净端点.

符号	含义
D, T	单个样本的标量维数和前向链总步数.
β_t, α_t	每步噪声方差，以及信号方差保留系数 $\alpha_t = 1 - \beta_t$.
$\bar{\alpha}_t$	累计保留系数 $\prod_{s=1}^{t} \alpha_s$，约定 $\bar{\alpha}_0 = 1$.
a_t, b_t	边缘信号和噪声标准差，$a_t = \sqrt{\bar{\alpha}_t}$、$b_t = \sqrt{1 - \bar{\alpha}_t}$.
ϵ, ξ_t	标准高斯随机变量；边缘总噪声与单步噪声应区分.
$\widetilde{\mu}_t, \widetilde{\beta}_t$	给定干净样本时反向条件后验的均值和标量方差.
μ_θ, σ_t^2	模型反向条件分布的均值及固定方差.
$\epsilon_\theta, \widehat{x}_0, v_\theta$	噪声预测、干净样本预测和速度参数化预测.

离散高斯扩散的推导条件

本章取固定日程 $0 < \beta_t < 1$，前向各步使用相互独立且与 x_0 独立的标准高斯噪声. 除输出端点另作说明，反向模型取具有正方差的各向同性高斯分布. 数据预处理和尺度保持固定. 高斯条件后验公式针对 $t \geqslant 2$；$t = 1$ 给定 x_0 后发生退化，不能直接代入非退化高斯 KL 公式.

41.1.2 递推扰动

前向过程（Forward Process）定义为

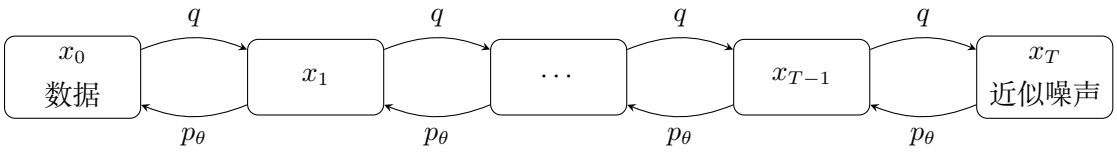
$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1}), \qquad q(x_t \mid x_{t-1}) = \mathcal{N}(\sqrt{\alpha_t} x_{t-1}, \beta_t I_D). \tag{41.1}$$

等价的采样表示为 $x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{\beta_t} \xi_t$，其中 $\xi_t \sim \mathcal{N}(0, I_D)$. 前一状态被缩放后再叠加噪声，使单位方差高斯保持单位方差；这不意味着任意原始数据在每个时刻都具有单位方差.

若原数据均值与协方差分别为 μ_0, Σ_0，则后面将得到

$$\mathbb{E}[x_t] = a_t \mu_0, \qquad \mathrm{Cov}(x_t) = \bar{\alpha}_t \Sigma_0 + (1 - \bar{\alpha}_t) I_D. \tag{41.2}$$

因此噪声化会逐渐削弱原数据的均值、尺度和相关结构. 只有累计信号足够小，末端分布才适合用标准高斯近似；“步数很多”本身不是充分的尺度说明.



41.2 前向边缘分布的闭式

41.2.1 累积噪声

将前向递推连续代入，得到

$$x_t = \left(\prod_{r=1}^{t} \sqrt{\alpha_r} \right) x_0 + \sum_{s=1}^{t} \left(\sqrt{\beta_s} \prod_{r=s+1}^{t} \sqrt{\alpha_r} \right) \xi_s. \tag{41.3}$$

空乘积取 1. 独立高斯的线性组合仍为高斯，因此条件均值是 $\sqrt{\bar{\alpha}_t} x_0$，条件协方差是标量 V_t 乘 I_D，其中

$$V_t = \sum_{s=1}^{t} \beta_s \prod_{r=s+1}^{t} \alpha_r. \tag{41.4}$$

用 $\beta_s = 1-\alpha_s$ 将每项写成两个乘积之差，求和发生望远镜消去，得到 $V_t = 1-\prod_{s=1}^{t}\alpha_s = 1-\bar{\alpha}_t$．也可以使用递推 $V_t = \alpha_t V_{t-1} + \beta_t$、$V_0 = 0$ 归纳证明．

于是**前向边缘分布**（Forward Marginal Distribution）为

$$q(x_t \mid x_0) = \mathcal{N}(a_t x_0, b_t^2 I_D), \qquad x_t = a_t x_0 + b_t \epsilon, \quad \epsilon \sim \mathcal{N}(0, I_D). \tag{41.5}$$

式 (41.5) 的 ϵ 表示累积扰动的归一化总噪声，不是原链中最后一步的 ξ_t．训练只需要某个时刻的 (x_0, x_t)，因此可以直接采样该边缘，无须先生成 $x_1, \ldots, x_{t-1}$．

41.2.2　同边缘分布的路径差异

固定同一对 (x_0, ϵ)，改变 t 并画出 $a_t x_0 + b_t \epsilon$，可以清楚观察信号与噪声的相对尺度．不过，这是一种共享总噪声的跨时刻耦合，通常不是式 (41.1) 独立增量前向链的一条样本路径．每个时刻边缘正确，并不能确定所有时刻的联合分布．

这一差别对采样器尤其重要：训练可以只依赖边缘加噪配对，而不同反向生成路径可能共享这些训练配对．后面的 DDIM 正是利用这种区分构造不同的采样过程．

41.2.3　信噪比及末端先验

以数据单位方差为比较口径，**信噪比**（Signal-to-Noise Ratio，SNR）为

$$\mathrm{SNR}_t = \frac{a_t^2}{b_t^2} = \frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}. \tag{41.6}$$

当原数据方差不是 1 时，某坐标的实际信号噪声方差比还要乘该坐标的数据方差．因此预处理尺度会改变相同日程对应的学习任务．

取末端先验 $p(x_T) = \mathcal{N}(0, I_D)$，条件末端 KL 可以直接计算：

$$D_{\mathrm{KL}}(q(x_T \mid x_0) \| p(x_T)) = \frac{1}{2}\left[\bar{\alpha}_T \|x_0\|^2 + D\{-\bar{\alpha}_T - \log(1-\bar{\alpha}_T)\}\right]. \tag{41.7}$$

固定 D 且数据具有有限二阶矩时，$\bar{\alpha}_T \to 0$ 使其期望趋于零．有限 T 下直接从标准高斯开始采样仍有先验近似误差；数据维数和尺度也会影响该误差，而不仅是单个系数看起来是否接近零．

41.3　反向条件后验的配方推导

41.3.1　可计算的后验及未知逆分布

生成希望得到 $q(x_{t-1} \mid x_t)$，但它需要对未知的干净数据后验积分：

$$q(x_{t-1} \mid x_t) = \int q(x_{t-1} \mid x_t, x_0) q(x_0 \mid x_t)\, \mathrm{d}x_0. \tag{41.8}$$

即使给定 x_0 的条件是高斯，该混合分布通常也不是单一高斯. 模型以高斯反向转移近似它，并不能由前向转移是高斯直接推出逆转移严格高斯.

训练时 x_0 已知,因此可使用**反向条件后验**(Reverse Conditional Posterior)$q(x_{t-1} \mid x_t, x_0)$. 令 $y = x_{t-1}$，根据前向马尔可夫性质和贝叶斯公式，

$$q(y \mid x_t, x_0) \propto q(x_t \mid y) q(y \mid x_0). \tag{41.9}$$

取负对数并去掉与 y 无关的常数，得到

$$-2 \log q(y \mid x_t, x_0) = \frac{\|x_t - \sqrt{\alpha_t} y\|^2}{\beta_t} + \frac{\|y - \sqrt{\bar{\alpha}_{t-1}} x_0\|^2}{1 - \bar{\alpha}_{t-1}} + C. \tag{41.10}$$

展开平方，整理为 $A_t \|y\|^2 - 2 h_t^{\mathsf{T}} y + C'$，其中

$$A_t = \frac{\alpha_t}{\beta_t} + \frac{1}{1 - \bar{\alpha}_{t-1}} = \frac{1 - \bar{\alpha}_t}{\beta_t (1 - \bar{\alpha}_{t-1})}, \tag{41.11}$$

$$h_t = \frac{\sqrt{\alpha_t}}{\beta_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1 - \bar{\alpha}_{t-1}} x_0. \tag{41.12}$$

A_t 是标量精度，$h_t \in \mathbb{R}^D$. 配方为

$$A_t \|y - A_t^{-1} h_t\|^2 + C'', \tag{41.13}$$

因此后验方差与均值分别为

$$\widetilde{\beta}_t = A_t^{-1} = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t, \tag{41.14}$$

$$\widetilde{\mu}_t = \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1 - \bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t} (1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} x_t. \tag{41.15}$$

两项系数来自精度加权，而非任意的图像插值比例；一般不能假定二者之和恰好为 1.

41.3.2 端点退化

$t = 1$ 时，给定 x_0 就已经知道 x_{t-1}，所以该条件分布是集中在 x_0 的点质量. 式 (41.14) 形式上给出零方差，表示退化而不是可直接取对数的普通高斯密度. 训练把这一端点单独写成数据重建项；采样端点也应依据明确输出约定处理. [1]

41.4 变分目标

41.4.1 Jensen 不等式及下界

反向过程（Reverse Process）定义模型联合分布

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t). \tag{41.16}$$

[1]若数据是离散像素，严格像素概率需要将连续密度在对应像素区间积分；连续高斯密度值不能直接当作离散像素概率. 本章先采用连续数据口径讲解概率构造.

直接积分得到 $p_\theta(x_0)$ 很困难．引入已知的 $q(x_{1:T}\mid x_0)$，有

$$\log p_\theta(x_0)=\log \mathbb{E}_{q(x_{1:T}|x_0)}\left[\frac{p_\theta(x_{0:T})}{q(x_{1:T}\mid x_0)}\right] \tag{41.17}$$

$$\geqslant \mathbb{E}_q[\log p_\theta(x_{0:T})-\log q(x_{1:T}\mid x_0)]. \tag{41.18}$$

右侧称为**证据下界**（Evidence Lower Bound，ELBO）．本章令 $\mathcal{L}_{\mathrm{VLB}}$ 为其负值，因而 $-\log p_\theta(x_0)\leqslant \mathcal{L}_{\mathrm{VLB}}(x_0)$．所谓“下界”作用于对数似然，取负号后是负对数似然的上界．

差距可以写成

$$\mathcal{L}_{\mathrm{VLB}}(x_0)+\log p_\theta(x_0)=D_{\mathrm{KL}}(q(x_{1:T}\mid x_0)\|p_\theta(x_{1:T}\mid x_0))\geqslant 0. \tag{41.19}$$

所以优化界不等于每次更新都直接最大化精确似然；还存在近似后验与模型后验之间的差距．

41.4.2　变分分解

固定 x_0 后，前向路径可以反向分解为

$$q(x_{1:T}\mid x_0)=q(x_T\mid x_0)\prod_{t=2}^{T} q(x_{t-1}\mid x_t,x_0). \tag{41.20}$$

把它代入负下界，按相同变量的对数比组合，得到

$$\mathcal{L}_{\mathrm{VLB}}(x_0)=\underbrace{D_{\mathrm{KL}}(q(x_T\mid x_0)\|p(x_T))}_{L_T} \tag{41.21}$$

$$+\sum_{t=2}^{T}\underbrace{\mathbb{E}_{q(x_t|x_0)}D_{\mathrm{KL}}(q(x_{t-1}\mid x_t,x_0)\|p_\theta(x_{t-1}\mid x_t))}_{L_{t-1}} \tag{41.22}$$

$$\underbrace{-\mathbb{E}_{q(x_1|x_0)}\log p_\theta(x_0\mid x_1)}_{L_0}. \tag{41.23}$$

L_T 将加噪终点与先验对齐；L_{t-1} 约束每一步逆转移；L_0 描述生成端点如何赋予数据概率．固定日程和先验时，L_T 不依赖 θ，可以从梯度计算中省去，但计算完整似然界时仍须计入．

41.4.3　固定方差下的均值误差

令 $p_\theta(x_{t-1}\mid x_t)=\mathcal{N}(\mu_\theta(x_t,t),\sigma_t^2 I_D)$，且 $\sigma_t^2>0$ 固定．对 $t\geqslant 2$，高斯 KL 为

$$D_{\mathrm{KL}}=\frac{\|\widetilde{\mu}_t-\mu_\theta\|^2}{2\sigma_t^2}+\frac{D}{2}\left[\frac{\widetilde{\beta}_t}{\sigma_t^2}-1+\log\frac{\sigma_t^2}{\widetilde{\beta}_t}\right]. \tag{41.24}$$

第二部分在上述假设下与 θ 无关，所以均值学习化为加权平方误差．如果反向方差也由网络预测，比例项与对数项一般都依赖参数，不能删除后仍宣称优化完整变分目标．

41.5 噪声预测及输出参数化

41.5.1 噪声预测目标

由 $x_t = a_t x_0 + b_t \epsilon$ 得 $x_0 = \dfrac{x_t - b_t \epsilon}{a_t}$. 将其代入式 (41.15)，化简为

$$\widetilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{b_t} \epsilon \right). \tag{41.25}$$

于是用**噪声预测**（Noise Prediction）网络 $\epsilon_\theta(x_t, t)$ 定义

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{b_t} \epsilon_\theta(x_t, t) \right). \tag{41.26}$$

两者相减并平方，得到依赖参数的逐步项

$$L_{t-1} = \mathbb{E}\left[w_t \| \epsilon - \epsilon_\theta(x_t, t) \|^2 \right] + C_t, \qquad w_t = \frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1 - \bar{\alpha}_t)}. \tag{41.27}$$

这里期望包含干净数据和加噪变量；C_t 与均值参数无关. 该公式说明噪声预测来自概率目标的重新参数化，而非需要模型辨认一份在图像中可唯一分离的随机纹理.

41.5.2 简化损失的权重效应

常用**简化噪声损失**（Simplified Noise Loss）为

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{x_0,\, t \sim \text{Unif}\{1,\dots,T\},\, \epsilon} \| \epsilon - \epsilon_\theta(a_t x_0 + b_t \epsilon, t) \|^2. \tag{41.28}$$

它省去或修改了式 (41.27) 中的时间权重，并将端点也纳入噪声回归. 它与变分构造密切相关，但不是完整式 (41.23) 的无条件等价表达. 只有保留对应权重及端点概率项，才能恢复所选概率模型的完整界.

若希望用非均匀时间分布 $r_t > 0$ 估计内部时间项之和，则

$$\sum_{t=2}^{T} \mathbb{E}[\ell_t] = \mathbb{E}_{t \sim r}\left[\frac{\ell_t}{r_t} \right], \tag{41.29}$$

其中 r 在 $\{2, \dots, T\}$ 上归一化，$\ell_t = w_t \| \epsilon - \epsilon_\theta \|^2$. 改变抽样分布却不补偿 $\dfrac{1}{r_t}$，会改变目标，而不仅改变方差. 批内再除以 D 是损失尺度约定；报告损失时必须说明是每样本平方和还是每元素均值.

网络学习的是条件统计量

给定 x_t, t，平方损失的总体最优预测为 $\mathbb{E}[\epsilon \mid x_t, t]$. 因为 x_0 未知，同一个带噪输入可能来自多个干净样本，网络通常无法恢复训练过程中某次独立噪声抽样的全

部随机细节．逐步生成也不是对某条历史加噪轨迹进行确定性倒放．

上述结论可由平方展开证明：令 $m=\mathbb{E}[\epsilon\mid x_t,t]$，则对任意预测 f，条件风险为 $\mathbb{E}[\|\epsilon-m\|^2\mid x_t,t]+\|m-f\|^2$，交叉项因条件均值为零消失．

41.5.3　干净样本及速度参数化

干净样本预测（Clean Sample Prediction）直接输出 $\widehat{x}_0$，对应噪声估计为 $\dfrac{x_t-a_t\widehat{x}_0}{b_t}$．

速度参数化（Velocity Parameterization）定义

$$v=a_t\epsilon-b_tx_0. \tag{41.30}$$

该参数化见渐进蒸馏研究 [228]．因 $a_t^2+b_t^2=1$，有一个正交变换及其逆：

$$\begin{bmatrix}x_t\\ v\end{bmatrix}=\begin{bmatrix}a_t & b_t\\ -b_t & a_t\end{bmatrix}\begin{bmatrix}x_0\\ \epsilon\end{bmatrix},\qquad \widehat{x}_0=a_tx_t-b_tv_\theta,\quad \widehat{\epsilon}=b_tx_t+a_tv_\theta. \tag{41.31}$$

令 $a=\cos\phi,b=\sin\phi$，则对固定 (x_0,ϵ)，$\dfrac{\mathrm{d}x}{\mathrm{d}\phi}=-bx_0+a\epsilon=v$．因此“速度”对应这条角度参数化路径，不等于任意连续时间下的 $\dfrac{\mathrm{d}x}{\mathrm{d}t}$，更不能直接等同于线性流匹配中的 $\epsilon-x_0$．

参数化之间在非退化端点可以转换，但无权重平方损失并不相同．对同一固定 x_t 的预测误差，

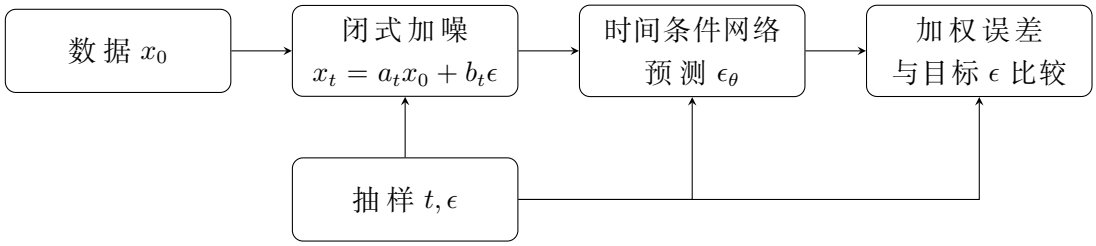
$$\|\widehat{\epsilon}-\epsilon\|^2=\frac{a_t^2}{b_t^2}\|\widehat{x}_0-x_0\|^2,\quad \|\widehat{x}_0-x_0\|^2=b_t^2\|v_\theta-v\|^2,\quad \|\widehat{\epsilon}-\epsilon\|^2=a_t^2\|v_\theta-v\|^2. \tag{41.32}$$

因此等权预测 x_0、ϵ 或 v 隐含不同时间权重．靠近 $a_t=0$ 时，从噪声预测除以 a_t 恢复 x_0 可能放大误差；靠近 $b_t=0$ 时，从 x_0 反推噪声也有相应问题．有限精度、端点选择和裁剪规则均属于采样协议．

41.6　时间条件及训练路径

时间条件告诉网络当前噪声尺度．可以把离散 t 映射到正弦和余弦特征，再通过多层感知机注入主干，也可以使用对数信噪比 $\lambda_t=\log(\dfrac{a_t^2}{b_t^2})$．索引相同但日程不同，会对应不同恢复任务；模型输入的时间单位、归一化与日程必须共同确定．端点处对数信噪比可能发散，应采用训练定义的有限范围或专门端点处理．

对图像批量，x_0,ϵ,x_t 通常形状为 $[B,C,H,W]$，时间索引为 $[B]$．取出的 a_t,b_t 重排为 $[B,1,1,1]$ 后广播；预测输出应与 x_t 同形．概率推导不限定主干采用卷积 U-Net 还是 Transformer，它们改变函数表达与计算结构，不改变加噪变量的定义．



算法 41.1 噪声预测训练

输入为数据分布、固定日程、预测类型、时间抽样分布与损失权重；输出为训练后的预测网络.

1. 抽取干净样本 x_0，执行固定预处理；独立抽取时间 t 和标准高斯 ϵ.
2. 由累计日程直接构造 $x_t = a_t x_0 + b_t \epsilon$，不运行完整前向链.
3. 输入 x_t 和约定的时间条件；按预测类型构造目标 ϵ、x_0 或 $a_t \epsilon - b_t x_0$.
4. 根据选定目标及时间抽样规则计算权重，执行反向传播和参数更新；完整变分训练还需对应的端点与方差项.
5. 达到预定训练步数或明确定义的训练停止条件后输出权重和日程、预测类型、数据尺度配置.

不变量为目标与采样噪声来自同一配对、时间与累计日程一致、所有样本使用相同预处理约定. 训练步数与扩散时间步是两种不同计数.

41.7 DDIM 采样

41.7.1 随机反向链

生成不具有已知干净样本. 它从 $x_T \sim \mathcal{N}(0, I_D)$ 开始，在每一步预测噪声，再依据反向分布更新

$$x_{t-1} = \mu_\theta(x_t, t) + \sigma_t \xi_t, \qquad \xi_t \sim \mathcal{N}(0, I_D). \tag{41.33}$$

内部步骤的随机变量应重新抽取. 常见样本生成在 $t = 1$ 使用确定的输出均值，不再加入噪声；如果模型定义了非退化的输出似然，则严格从该似然抽样与输出均值是不同约定，需要分别说明.

选择 $\sigma_t^2 = \widetilde{\beta}_t$ 或其他方差会改变采样过程. 不能只删除部分训练时间索引，却继续使用原来的相邻单步 α_t, β_t 更新. 跨步转移需要按两个保留时间之间的累计系数重建.

41.7.2　共享边缘的不同路径

去噪扩散隐式模型（Denoising Diffusion Implicit Model，DDIM）构造与相同加噪边缘兼容的另一类路径，并允许确定性采样 [229]．设从时间 t 跳至更早时间 $s<t$，记模型恢复的干净样本为 $\widehat{x}_0=\dfrac{x_t-b_t\widehat{\epsilon}}{a_t}$，一种更新写为

$$x_s=a_s\widehat{x}_0+\sqrt{b_s^2-\sigma_{t\to s}^2}\,\widehat{\epsilon}+\sigma_{t\to s}\xi, \tag{41.34}$$

其中

$$\sigma_{t\to s}=\eta\sqrt{\frac{1-\bar{\alpha}_s}{1-\bar{\alpha}_t}}\sqrt{1-\frac{\bar{\alpha}_t}{\bar{\alpha}_s}},\qquad 0\leqslant\eta\leqslant 1. \tag{41.35}$$

在此范围及合法日程下，根号项非负．若给定真实 x_0 且 $\widehat{\epsilon}=\dfrac{x_t-a_tx_0}{b_t}$，新状态噪声由两个独立标准高斯的线性组合构成，方差相加为 b_s^2，所以保留 $q(x_s\mid x_0)$ 边缘．这种构造不要求各时间的联合分布等于原来的前向马尔可夫链．

$\eta=0$ 时，没有额外随机项，给定初始噪声、网络及时间网格后路径确定．相邻时间且 $\eta=1$ 时，方差成为 $\widetilde{\beta}_t$，均值与对应后验均值参数化一致．跨步 $\eta=1$ 则对应重构后的粗粒度系数，不能等同于完整原链的所有中间模型调用．

“边缘构造兼容”不保证有限能力网络在任意稀疏网格上生成同样质量．网络误差、离散化路径和时间覆盖会相互作用；减少步数是改变采样算法与误差条件，必须结合既定模型评价．[2]

算法 41.2　反向生成

输入为训练后的网络、兼容日程、预测类型、递减时间网格和更新规则；输出为生成样本．

1. 从约定末端先验抽取 x_T；固定模型和采样配置．
2. 对网格中的当前时间 t，将 x_t 与时间条件送入网络，并按预测类型转换为更新公式所需的噪声或干净样本估计．
3. 使用当前 t 与目标 s 对应的系数计算均值和方差；随机采样时抽取新的独立噪声，确定性规则则不添加随机项．
4. 按指定端点规则更新至 $s=0$，终止循环，按数据表示执行逆预处理或交给后续解码器．

不变量为时间严格递减、状态与系数对应、预测类型兼容、方差非负．生成路径不读取真实 x_0，也不通过生成循环执行参数训练．

[2]确定性采样仍以随机初始噪声为输入，因而输出整体仍可构成分布．确定性指给定初始状态后的映射，不表示所有请求生成同一结果．

取构造日程 $\beta_1 = 0.1, \beta_2 = 0.2$，则 $\alpha_1 = 0.9, \alpha_2 = 0.8$、$\bar{\alpha}_1 = 0.9, \bar{\alpha}_2 = 0.72$. 设 $x_0 = 1$，边缘噪声 $\epsilon = 0.5$，得到

$$x_2 = \sqrt{0.72} + 0.5\sqrt{0.28} \approx 1.11310. \tag{41.36}$$

条件后验精度为 $A_2 = \dfrac{0.8}{0.2} + \dfrac{1}{0.1} = 14$，所以 $\widetilde{\beta}_2 = \dfrac{1}{14} \approx 0.071429$. 线性项为

$$h_2 = \frac{\sqrt{0.8}}{0.2} x_2 + \frac{\sqrt{0.9}}{0.1}, \qquad \widetilde{\mu}_2 = \frac{h_2}{14} \approx 1.03320. \tag{41.37}$$

也可由噪声形式计算 $\widetilde{\mu}_2 = \dfrac{x_2 - \frac{0.2 \times 0.5}{\sqrt{0.28}}}{\sqrt{0.8}}$，得到同一均值. 这个后验仍有非零方差，并不是直接恢复 $x_0 = 1$.

若网络预测 $\widehat{\epsilon} = 0.4$，其均值误差为

$$\mu_\theta - \widetilde{\mu}_2 = \frac{0.2(0.5 - 0.4)}{\sqrt{0.8}\sqrt{0.28}} \approx 0.04226. \tag{41.38}$$

选择 $\sigma_2^2 = \widetilde{\beta}_2$，高斯方差项为零，KL 仅为

$$\frac{(0.04226)^2}{\frac{2}{14}} \approx 0.0125. \tag{41.39}$$

噪声权重同样给出 $w_2 = \dfrac{0.2^2}{[2(\frac{1}{14})(0.8)(0.28)]} = 1.25$，乘噪声误差平方 $(0.1)^2$ 得 0.0125.

再以真实噪声估计考察 $2 \to 1$ 更新. DDIM 在 $\eta = 0$ 时给出 $x_1 = \sqrt{0.9} + \sqrt{0.1} \times 0.5 \approx 1.10680$；$\eta = 1$ 时更新均值约为 1.03320，标准差为 $\sqrt{\dfrac{1}{14}}$. 两者并不矛盾：前者选择共享总噪声的确定性耦合，后者采用具有条件方差的路径. 本例为了手算故意只用两步，$\bar{\alpha}_2 = 0.72$ 远未接近零，不能将其作为从标准高斯生成真实数据的充分日程.

41.8　误差边界及参数化一致性

前向闭式正确，只证明扰动分布被实现；它不证明网络已学习条件统计量. 训练损失下降只说明选定加权回归目标改善；它不必对应精确似然、感知质量或多样性的单调改善. 反向采样每一步依赖前一步生成状态，模型可能在偏离训练边缘的区域继续运行，误差不能简单当作相互独立的单步噪声求和.

需要保持一致的对象包括数据尺度、累计日程、时间映射、预测类型、反向方差、采样网格和端点规则. 网络输出形状相同并不能证明参数化相同. 对 $\widehat{x}_0$ 裁剪后，如果更新仍使用未重新关联的噪声估计，也会改变所实现的算法；这种处理必须明确写入采样定义.

前向加噪、时间嵌入、噪声预测和单步均值必须遵循相同的时间步与参数化约定. 固定样本与同一噪声的分镜用于比较尺度，不应误认为独立增量链的实际轨迹；调度

器集中保存系数与更新协议，不是任意可替换的循环包装．条件 U-Net、文本交叉注意力、无分类器引导和 VAE 属于第四十二章《条件视觉生成》的条件生成扩展，流匹配和视频进一步改变路径或数据结构，不能仅凭张量同形直接互换目标．

第四十二章　条件视觉生成

条件生成要求模型在多样的视觉分布中满足指定信息：文本描述、参考图像、空间布局或时间条件. 它既涉及概率路径，也涉及表示压缩、网络结构和求解器. 第四十一章《扩散模型的概率原理》推导了加噪与反向过程；本章解释条件如何进入网络、引导如何改变预测，以及潜空间和视频扩展如何改变成本与误差.

离散视觉词元还可以采用自回归分解逐项生成，再由相应解码器恢复图像；该路线与反向扩散不同，表示接口见第四十章《多模态建模》. 因此，视觉输出并不必然意味着去噪过程.

42.1　条件生成的对象及符号

本章主要符号见表 42.1.

表 42.1: 条件视觉生成的主要符号.

符号	含义
x, z, c	像素样本、潜变量和条件表示.
H, W, C_x, f, C_z	图像高宽、像素通道数、空间压缩因子与潜通道数.
N, M, d, d_c	视觉位置数、条件位置数、视觉与条件特征宽度.
a_t, b_t	扰动路径中的信号系数与噪声系数.
s_t, ϵ_θ, w	得分函数、噪声预测与本章 CFG 引导系数.
τ, u_θ	从噪声走向数据的连续时间与速度场.
F, F', p_t, p_h, p_w	原始及潜空间帧数与时空补丁尺寸.

文本生成图像关注条件遵循与分布质量，图像编辑还要求保留未编辑区域，视频生成进一步要求跨帧身份、运动与时间一致性. 条件可以作为序列、空间张量或全局向量进入网络，不必都转换成一段文本. 训练时必须明确每种条件的缺失表示、对齐关系和可见范围.

42.2　时间条件及去噪主干

42.2.1　U-Net 的多尺度路径

U 形网络（U-Net）由下采样路径、瓶颈与上采样路径构成．下采样减少空间位置数，扩大后续局部算子的有效感受野；跳跃连接将较高分辨率特征传给对应上采样层．它并不自动保证全局建模，感受野仍取决于层数、卷积与注意力配置．

时间编码可采用

$$e_{2j}(t)=\sin(\omega_j t),\qquad e_{2j+1}(t)=\cos(\omega_j t),\tag{42.1}$$

再经多层感知机形成调制向量．对特征 h，常见条件调制为

$$h'=\gamma(e_t,c)\odot \mathrm{Norm}(h)+\beta(e_t,c).\tag{42.2}$$

γ,β 沿空间维广播；其形状必须与通道一致．离散步编号、归一化时间与对数信噪比不是同一输入，替换时间编码时必须保持训练契约．

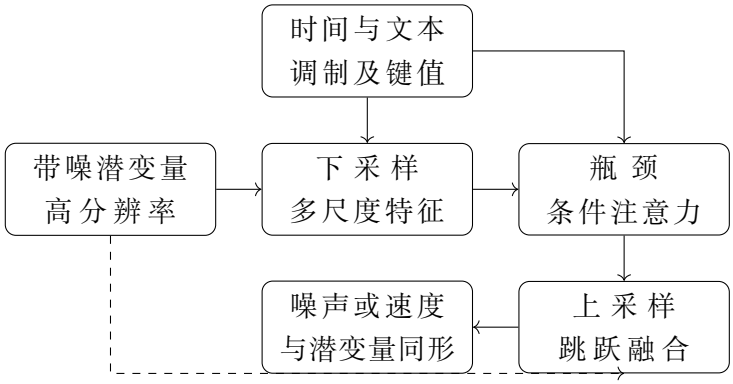


图 42.1: 条件 U-Net 的结构关系；跳跃连接传递特征，不绕过去噪任务．

图 42.1 中的时间与语义条件在不同尺度影响预测，跳跃特征仍需经过上采样分支的可训练计算．

42.2.2　交叉注意力及联合注意力

视觉特征 $X\in\mathbb{R}^{N\times d}$ 和条件特征 $C\in\mathbb{R}^{M\times d_c}$ 通过

$$Q=XW_Q,\quad K=CW_K,\quad V=CW_V,\qquad A=\mathrm{softmax}\left(\frac{QK^{\top}}{\sqrt{d_h}}+B\right)V\tag{42.3}$$

建立交叉注意力．这里 $W_Q\in\mathbb{R}^{d\times d_h}$，$W_K\in\mathbb{R}^{d_c\times d_h}$，$W_V\in\mathbb{R}^{d_c\times d_v}$，输出 $A\in\mathbb{R}^{N\times d_v}$；$B$ 排除条件填充等不可见位置．单头打分成本随 NMd_h 增长，而非视觉自注意力的 N^2d_h．

把视觉与文本映射到兼容宽度后联合处理，可以让两种表示相互更新，但会产生约 $(N+M)^2$ 的注意力配对数. 是否双向更新、是否使用独立投影和调制，是具体架构选择. 文本编码器、分词器、截断规则和空条件表示都属于制品，不可只替换编码器名称.

42.2.3 Transformer 主干及目标相互独立

扩散 Transformer（Diffusion Transformer，DiT）把潜变量切分为补丁并使用 Transformer 预测去噪目标 [230]. 若潜图为 $H_z \times W_z \times C_z$、补丁边长为 p，则位置数为 $N = \dfrac{H_z W_z}{p^2}$，每补丁输入宽度为 $p^2 C_z$. 输出投影恢复每补丁所需通道，再重排为潜图.

时间或类别可调制归一化与残差门控. 补丁变小会增加位置数；边长减半使 N 增至 4 倍，完整自注意力配对数增至 16 倍. 主干为 Transformer 并不意味着训练目标必为流匹配：噪声预测、速度参数化和路径速度回归都可以由不同主干实现.

42.3 无分类器引导的推导及边界

42.3.1 噪声外推

得分函数（Score Function）在本节指输入密度的梯度 $s_t(x) = \nabla_x \log p_t(x)$，不是分类分数. 给定条件 c，贝叶斯关系给出

$$\nabla_x \log p_t(c \mid x) = s_t(x \mid c) - s_t(x). \tag{42.4}$$

将条件信息梯度乘以 w 加到无条件得分上，得到

$$\widehat{s}_t = (1-w) s_t(x) + w s_t(x \mid c). \tag{42.5}$$

对同一高斯扰动 $x_t = a_t x_0 + b_t \epsilon$，密度可微且交换积分与微分合法时，

$$\nabla_{x_t} \log p_t(x_t \mid c) = \mathbb{E}\left[-\frac{x_t - a_t x_0}{b_t^2} \,\middle|\, x_t, c \right] \tag{42.6}$$

$$= -\frac{\mathbb{E}[\epsilon \mid x_t, c]}{b_t}. \tag{42.7}$$

因此噪声回归的最优条件均值与得分成比例. 在相同 $b_t > 0$ 下，式 (42.5) 对应

$$\widehat{\epsilon} = \epsilon_\theta(x_t, t, \varnothing) + w\left[\epsilon_\theta(x_t, t, c) - \epsilon_\theta(x_t, t, \varnothing)\right]. \tag{42.8}$$

无分类器引导（Classifier-Free Guidance，CFG）通过条件丢弃训练，让生成模型提供两种估计，无须另训分类器 [231]. 丢弃的是语义条件，不是时间；空条件的编码方式也必须在训练中确定.

42.3.2　外推精度边界

本章 $w=0$ 返回无条件预测，$w=1$ 返回普通条件预测，$w>1$ 为外推．有的文献写成 $(1+s)\epsilon_c - s\epsilon_u$，其 $s=w-1$，不能按相同数值比较．

在固定时间且归一化积分存在时，式 (42.5) 是形式密度 $p_t(x\mid c)^w p_t(x)^{1-w}$ 的对数梯度．但不同时间的这种密度族未必满足原前向扩散关系，因此不能据此断言最终采样分布恰为 $p_0(x\mid c)^w p_0(x)^{1-w}$．模型近似与求解误差还会进一步改变结果．

设无条件与条件预测分别为 $(0.2,-0.1)$ 与 $(0.5,0.3)$，则 $w=2$ 得 $(0.8,0.7)$．这说明差向量被放大，并不说明图像更正确．若两支预测误差为 e_u, e_c，引导误差满足

$$\|e_{\text{cfg}}\| \leqslant |1-w|\|e_u\| + |w|\|e_c\|. \tag{42.9}$$

大引导可能放大误差、改变多样性和产生伪影．以负面提示替代空条件时，基线已变成另一个条件分支，不再是真正无条件密度；其语义应明确说明．[1]

42.4　潜空间表示及重建限制

42.4.1　VAE 目标及重参数化

变分自编码器（Variational Autoencoder，VAE）以 $q_\phi(z\mid x)$ 近似后验，解码器为 $p_\psi(x\mid z)$．由 Jensen 不等式，

$$\log p_\psi(x) \geqslant \mathbb{E}_{q_\phi} \log p_\psi(x\mid z) - D_{\mathrm{KL}}(q_\phi(z\mid x)\|p(z)). \tag{42.10}$$

对对角高斯 $q_\phi=\mathcal{N}(\mu, \operatorname{diag}\sigma^2)$ 与标准正态先验，KL 为

$$\frac{1}{2}\sum_j(\mu_j^2+\sigma_j^2-\log\sigma_j^2-1). \tag{42.11}$$

重参数化 $z=\mu_\phi(x)+\sigma_\phi(x)\odot\eta$、$\eta\sim\mathcal{N}(0,I)$ 将随机输入与参数分离，使梯度沿均值与尺度传播．实际视觉自编码器可能加入感知或对抗重建目标，故不能将所有潜空间编码器都称为仅优化上述标准 ELBO．

潜空间扩散（Latent Diffusion）先学习压缩表示，再在该表示上训练生成过程 [232]．图像为 $H\times W\times C_x$、压缩率为 f 时，潜元素数为 $\dfrac{HWC_z}{f^2}$，原图与潜空间元素比为 $\dfrac{f^2C_x}{C_z}$．例如 $512\times512\times3$ 到 $64\times64\times4$，比值为 48，空间位置数减少 64 倍．元素比不等于端到端加速比，编码、解码和主干通道均有成本．

[1] 将两支合并为一个批次可以减少某些调度开销，但仍需处理两组条件前向，不等于只计算一支的成本．

42.4.2 缩放协议及不可恢复信息

若生成器训练使用 $z' = sz$，采样结束后必须先得到 $z = \dfrac{z'}{s}$ 再解码. 遗漏缩放改变解码器输入分布；不能将一个模型的常数复制给其他制品. 像素归一化、色域、裁剪、潜通道、后验采样或均值选择也必须一致.

编码器丢失的信息不会因为后续生成器更大而被可靠恢复. 细小文字、线条与局部身份细节尤其需要独立验证. 自编码器重建误差和生成误差属于不同来源：先对真实输入做编码解码对照，才能判断瓶颈是否已出现在表示层.

42.5 流匹配及连续路径

42.5.1 条件速度回归

本节以 $\tau \in [0,1]$ 从噪声走向数据，端点记为 Z_0 与 Z_1，区别于第四十一章《扩散模型的概率原理》从干净样本出发的离散加噪记号. 取一个端点联合分布，定义线性路径

$$Z_\tau = (1-\tau)Z_0 + \tau Z_1, \qquad U = Z_1 - Z_0. \tag{42.12}$$

流匹配（Flow Matching）可以通过条件路径监督速度场 [233]. 平方损失

$$\mathcal{L} = \mathbb{E}\|u_\theta(Z_\tau, \tau, c) - U\|^2 \tag{42.13}$$

在固定 (z, τ, c) 下的最优解为 $u^*(z,\tau,c) = \mathbb{E}[U \mid Z_\tau = z, c]$. 证明来自平方展开：任意预测与条件均值之差的平方，加上与预测无关的条件方差，构成全部条件风险.

对光滑紧支撑测试函数 φ，若微分与期望可以交换，

$$\frac{d}{d\tau}\mathbb{E}[\varphi(Z_\tau)] = \mathbb{E}[\nabla\varphi(Z_\tau)^\top U] = \mathbb{E}[\nabla\varphi(Z_\tau)^\top u^*(Z_\tau,\tau,c)]. \tag{42.14}$$

因此该边缘速度满足对应密度路径的弱形式连续性方程. 若速度场具备适当正则性与可积性，求解 $\dfrac{dZ_\tau}{d\tau} = u^*(Z_\tau,\tau,c)$ 可以运输这些边缘分布. 端点配对可以独立，也可来自其他耦合；它会改变条件速度与学习难度.

42.5.2 直线路径的生成边界

训练样本路径是直线，但同一中间位置可能对应多对端点，其条件平均速度会随位置与时间变化. 故学习到的边缘轨迹不必是各训练端点之间的那条直线，不能从线性插值直接推出一次网络调用即可精确生成.

Euler 更新为 $z_{k+1} = z_k + h_k u_\theta(z_k, \tau_k, c)$. 若场足够光滑，单步局部截断误差为二阶、固定区间的全局误差通常为一阶；网络误差另计. 以一维 $u(z) = z, z(0) = 1$ 为

例，精确终点为 e，单步 Euler 得 2，两步得 2.25，都不精确．该构造说明减少步数需要相应的求解或训练依据．

扩散中的 v 参数化常表示信号与噪声的特定线性组合，流匹配的 u 则是路径导数．两者都可能被接口称为 velocity，却不能只凭名称互换．预测对象、时间方向和求解器必须共同确定．

42.6　编辑、视频及联合媒体

图像编辑将参考图、遮罩或结构控制作为额外条件．若要求未编辑区域严格保持，应区分像素级约束与模型倾向；潜空间遮罩经过编码解码可能影响边缘及邻近区域．多次去噪中的已知区域替换还需匹配当前噪声尺度，不能直接把干净像素与高噪状态混合后宣称符合原模型．

视频潜变量可写为 $[B, C_z, F', H', W']$，时空补丁数为

$$N = \frac{F'}{p_t}\frac{H'}{p_h}\frac{W'}{p_w}, \tag{42.15}$$

这里假设尺寸可整除，真实实现的补齐与边界另计．设每帧有 S 个空间位置，共 F' 帧，全时空注意力配对数为 F'^2S^2；空间与时间分解注意力约为 $F'S^2 + SF'^2$，但其连接结构不同，不能称为完全相同算子的无损替换．

例如 16 个潜帧，每帧 32×32 位置，补丁为 $1 \times 2 \times 2$，则 $S = 256, N = 4096$，完整打分矩阵每头有约 1678 万项．帧数翻倍使全注意力项增至 4 倍，而空间分支仅翻倍．帧率、时长与潜时间压缩共同决定 F'；某些因果时间编码器对首帧采用特殊处理，不能简单假设所有模型都有 $F' = \dfrac{F}{f_t}$．

音视频联合生成需要共同时间基准、采样率与对齐条件．视觉连续不证明声画同步，片段看似合理也不证明物理动力学正确．所谓世界模型还需要动作条件、状态覆盖和闭环任务验证；视频观感不能替代控制稳定性证据．

42.7　训练、采样及制品一致性

算法 42.1　计算过程

条件潜空间生成的数据流．

1. 固定像素预处理、编码器与潜缩放，取得训练潜变量及条件表示；记录条件缺失规则．
2. 按训练目标采样时间和噪声，构造扰动或路径状态，计算噪声、样本或路径速度目标．
3. 训练主干，确保输出形状、监督对象与时间权重一致；条件丢弃只按已定义

策略执行.

4. 推理时从匹配先验初始化，用匹配的预测参数化和时间方向逐步求解；需要引导时组合两支预测.

5. 恢复潜缩放并解码，保留精确组件版本、生成配置与输出来源记录.

6. 分别评价重建限制、条件遵循、质量、多样性、编辑保持与视频时间一致性；不以单张图代替分布评估.

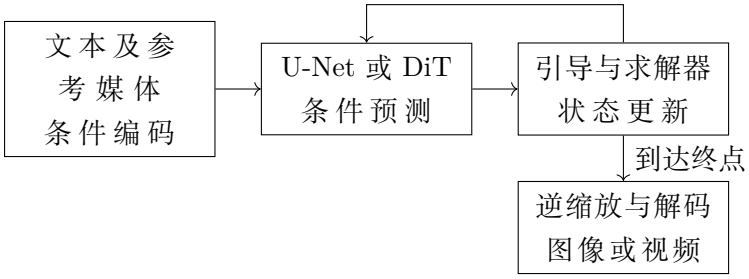


图 42.2: 预测网络与求解器反复协作，解码仅在规定终点执行.

图 42.2 显示预测与状态更新的分工；只有组件协议相容时，循环才具有所声明的概率含义. 质量评估需同时检查条件遵循、分布多样性及任务细节：图文相似度只检查所用表示空间中的接近程度，不能独立证明计数、否定或局部文字正确；某种分布距离很小，也不能证明每个条件下都生成了正确结果.

服务预算应同时限制分辨率、帧数、采样步数、候选数和并发；请求数本身不足以描述工作量. 组件清单包含主干、文本编码器、分词器、自编码器、缩放、求解器与控制模块. 代码可获取、权重可获取和特定部署用途允许是不同事实，应依据所用精确制品记录，不能继承同家族其他版本的结论.

中文术语索引

A

安全防护, 133

B

半结构化稀疏, 434
半精度浮点, 347
包含率, 239
饱和误差, 422
暴露偏差, 106
保障论证, 133
八位浮点, 347, 428
贝尔曼方程, 300
背压, 392, 408, 525
变分自编码器, 540
变更数据捕获, 181
边际收益, 157
编码器—解码器, 84
标签词, 152
标签基数, 408
闭合环负载, 394
比较并交换, 203, 208
补偿事务, 209
补丁, 501, 513
补丁化, 502
补丁嵌入, 502
部分可观测马尔可夫决策过程, 193, 299
不可变制品, 395

C

裁剪, 422
采样率, 522
参考策略, 316
参数高效微调, 264, 268
策略滞后, 329
层归一化, 81
测试时计算, 157, 331
测试污染, 333
差分隐私, 142
长短期记忆, 4
长度分桶, 364, 389
长期记忆, 200
超售比, 352
查询, 60
查询变换器, 517
查询分解, 177
查询改写, 176
乘积量化, 166
程序性记忆, 200
成员推断, 141
成组调度, 359
持久性, 355
持久执行, 208
重采样器, 517
重排, 174
重排序, 167
稠密检索, 163, 173
稠密模型, 484
稠密专家混合, 455
Chrome 开发者工具协议, 214
词项, 160
词元 F1, 125
词元间时延, 387
词元级 RAG, 172
词元级蒸馏, 440
词元物化, 243
簇重采样, 129
存活时间, 183

存活探针, 400

D

答案规范化, 155
大规模多任务语言理解, 119
大规模多学科多模态理解, 120
待估目标量, 117
单指令多线程, 346
倒排列表, 161
倒排索引, 161
倒排文件向量索引, 165
倒数排名融合, 166
得分函数, 539
得分函数估计, 302
电能使用效率, 361
迭代检索, 177
低精度训练, 427
低秩分解, 436
低秩适配, 269
动量, 248
动态分辨率, 514
动态量化, 424
动态填充, 264
动作定位, 213
动作价值, 300
短时傅里叶变换, 522
对比学习, 163
对比语言图像预训练, 514
对称量化, 421
队首阻塞, 393
多查询注意力, 72, 378
多词元预测, 488
多候选择优, 156, 331
多轮共指消解, 120
多头潜在注意力, 380, 487
多头注意力, 68, 76, 378
多智能体系统, 201
多租户隔离, 360

E

二维位置插值, 505

F

F1 分数, 124
发布清单, 246
仿射变换, 13
仿射量化, 420
反量化, 421
反事实评估, 184
反向过程, 529
反向条件后验, 529
反应式架构, 196
非对称距离计算, 166
非结构化剪枝, 431
非易失性存储器快速接口, 354
非一致内存访问, 350
分布式数据并行, 366
分层可导航小世界图, 165
分发, 458
分块预填充, 389
分类标记, 504
分片, 245
分页键值缓存, 381
分页优化器, 274
分组查询注意力, 72, 379
分组划分, 242
分组量化, 423
浮点操作数, 336
负缓存, 183
服务端发送事件, 398
服务降级, 404
幅值剪枝, 431
副作用, 203

G

概率校准, 124
概率密度函数, 14
概率质量函数, 14
干净样本预测, 532
高带宽内存, 347
高斯误差线性单元, 78
高斯误差线性门控单元, 79
高速串行计算机扩展总线, 350
割带宽, 352
隔离令牌, 209
个人可识别信息, 140, 239
工具调用, 203

共享专家, 464
工作记忆, 200
工作流, 197
观测, 193
广义优势估计, 304
规划执行架构, 196
轨迹, 298
轨迹回放, 218
规模定律, 6, 292
归纳偏置, 9, 508
归一化折损累计增益, 169
归一化指数函数, 21, 63
归约散播, 353, 367
过程监督, 321
过程奖励模型, 321
故障域, 397

H

海森矩阵, 432
合并, 458
合并访存, 348
核采样, 110
合成数据, 243
合成指令数据, 262
核对, 209
红队评估, 146
宏平均, 124
后门, 139
后置层归一化, 82
滑动窗口注意力, 418, 474
缓存淘汰, 382
幻觉, 107
环形缓存, 384
回放数据, 266
恢复训练, 443
会话模板, 261
混合检索, 166, 174
混合精度, 366
混合增强, 509
混合专家模型, 454
混淆矩阵, 124

I

InfiniBand 网络, 352

J

Jaccard 相似度, 240
假负例, 164, 516
键, 60
检查点, 252
检查与使用时差, 138, 211
监督微调, 256
监督学习, 8
监督者架构, 201
奖励投机, 319, 328
简化噪声损失, 531
间接提示注入, 136
检索增强生成, 171, 479
剪枝, 431
键值缓存, 357, 376, 473
键值缓存量化, 420
交叉编码器, 167, 174
交叉熵, 12
交叉注意力, 61
交叉注意力融合, 519
教师强制, 90, 104, 285
结构化剪枝, 433
结构模式, 138
结果监督, 321
结果奖励模型, 321
解耦权重衰减, 250
解析失败, 155
截止期限, 399
集合通信, 353
激活重计算, 339, 366
激活感知权重量化, 426
激活加权低秩近似, 437
级联调用, 404
仅编码器, 84
进程序号, 365
近端策略优化, 307
技能, 196
精确率, 124
精确率—召回率曲线, 125
精确去重, 239

精确最近邻搜索, 164
静态量化, 424
静态批处理, 388
仅解码器, 84
仅权重量化, 420
金丝雀发布, 406
近似最近邻搜索, 165
计算机操作, 213
旧策略, 316
就绪探针, 400
继续预训练, 285
基于人类反馈的强化学习, 313
基于 Transformer 的双向编码表示, 446
基于优先级的流量控制, 352
基准测试, 117
局部敏感哈希, 240
拒答, 156, 179
绝对位置嵌入, 504
拒绝采样微调, 322
均方根归一化, 82
矩阵吸收, 380

K

开放环负载, 394
开放权重, 484
课程学习, 243, 290
科恩一致性系数, 127
可观测性, 407
可靠性, 133
可验证奖励强化学习, 320
可执行验证器, 126
控制面, 357, 396
快照清单, 180
跨模态对齐, 514
库尔贝克—莱布勒散度, 12
困惑度, 22, 105, 125, 287
扩散模型, 526
扩散 Transformer, 539

L

来源跨度, 245
蓝绿部署, 406
老化, 393
类型化记录, 237
量化尺度, 420
量化低秩适配, 274
量化感知训练, 427
量化校准, 424
连接器, 516
连续片段, 240
连续批处理, 388
连续提示调优, 276
离策略学习, 306
零点, 420
零方差组, 324
零冗余优化器, 368
零样本提示, 150
领域本体, 188
流匹配, 541
流式多处理器, 346
流式读取, 245
流式因果性, 524
流水并行, 370
留一基线, 324
漏放, 145
路由器, 454

M

马尔可夫决策过程, 298
美国数学邀请赛, 119
每秒输入输出操作数, 354
每输出词元时间, 363, 431, 478
蒙特卡洛估计, 304
门控网络, 454
门控线性单元, 79
面向召回的摘要评估指标, 126
幂等键, 209
幂等性, 138, 203, 246, 399
命名实体识别, 452
命中率, 169
模式适配器, 237
模态, 512
模型路由, 403
模型评审, 127
模型上下文协议, 206
模型适配器, 498

模型提取, 142
模型压缩, 419
模型制品, 395

N
难负例, 164
脑浮点 16, 347
内激活抑制与放大适配, 275
N-gram 语言模型, 40
逆文档频率, 162
n 元语言模型, 4

O
欧氏距离, 164

P
排空, 400
排列不变性, 93
排列等变性, 93
帕累托前沿, 131
偏好对, 311
偏置微调, 277
批内负例, 164, 516
评测污染, 139
评测运行框架, 118
评分量表, 127
评估污染, 239
平滑量化, 426
瓶颈适配器, 275
平均倒数排名, 169
平均精确率, 125
平均输出词元时间, 387
平移等变性, 508

Q
恰好一次, 245
潜变量, 171
强化学习, 9, 298
强扩展, 373
抢占, 359, 393
前 k 次采样通过率, 331
潜空间扩散, 540
前 k 项采样, 110
嵌入, 56
前向边缘分布, 528
前向过程, 527
前置层归一化, 82, 506
前缀缓存, 382
前缀解码器, 86
前缀融合, 518
前缀微调, 276
启动风暴, 355
启动探针, 400
切片评估, 130
情景记忆, 200
期望校准误差, 125
企业资源计划, 189
奇异值分解, 435
全参数微调, 268
全归约, 353, 367
全互换, 353, 367
全互换通信, 465
全局平均池化, 506
全收集, 353, 367
权重绑定, 57
权重分解低秩适配, 273
确定性变换, 238
去噪扩散概率模型, 526
去噪扩散隐式模型, 534

R
人类最后的考试, 119
任务单元, 186
融合反量化, 429
融合以太网远程直接内存访问, 352
容量因子, 459
软标签, 437
软件物料清单, 140
软提示, 276
软专家混合, 465
弱扩展, 373

S
删除标记, 168
删除血缘, 247
熵, 12
上下文并行, 371

上下文窗口, 468
上下文校准, 153
上下文学习, 5, 150
上下文压缩, 176, 479
少样本提示, 150
沙箱验证, 333
设备亲和性, 350
设计效应, 236
深层提示调优, 276
生成式模型训练后量化, 424
生成式预训练 Transformer, 103
声学词元, 523
时间对齐, 524
事件响应, 146
视觉 Transformer, 501, 513
视觉语言模型, 512
视觉指令微调, 521
示例库, 151
适配器路由, 279
适配器注册表, 279
事务发件箱, 209
时序差分, 304
首词元时延, 363, 387, 431
首词元延迟, 478
受试者工作特征曲线, 125
双编码器, 163
双线性插值, 505
双语评估替代指标, 125
双重量化, 274
数据防泄漏, 141
数据工程, 235
数据面, 357, 396
数据模式, 237
数据契约, 180
数据投毒, 139
数据新鲜度, 179
数据血缘, 237
数据增强, 509
束搜索, 112
数字签名, 140
四比特正态浮点, 274
Sigmoid 线性单元, 79
思维链, 143, 237, 322
思维链提示, 153
算术强度, 341, 348
算子融合, 414
速度参数化, 532
随机变量, 14
随机对照实验, 131
随机失活, 63, 91, 229, 272
损失缩放, 252
损失掩码, 519
缩放点积注意力, 61
索引快照, 168
Swish 门控线性单元, 79

T

探索不足, 329
贪心解码, 111
条件独立, 19
条件计算, 454
条件记忆, 489
替代目标, 306
梯度范数裁剪, 252
梯度累积, 365
提示微调, 276
提示注入, 136, 182
投影器, 516
图编译, 415
推测解码, 415
推理强度, 330
推理与行动交替, 197
推理蒸馏, 323
图形处理器, 346

U

U 形网络, 538

W

完全分片数据并行, 368
完全匹配, 125
WebDriver 双向协议, 214
威尔逊区间, 128
未归一化分数, 104
伪量化, 427
微平均, 124

威胁模型, 134
位置插值, 470
温度混合, 242
温度缩放, 109
屋顶线模型, 348
无分类器引导, 539
无辅助损失负载均衡, 488
无监督学习, 8
误拦, 145

X

线程束, 346
先到先服务, 393
向量雅可比积, 29
显式拥塞通知, 352
线性探测, 509
下一句预测, 451
协调遗漏, 394
写时复制, 382
卸载, 369
细粒度专家, 464
行动者—评论家, 305
行为安全, 133
形状分桶, 415
信任边界, 134
新鲜度, 200
信息检索, 160
信息价值, 199
信息熵, 19
信用分配, 303
信噪比, 528
稀疏奖励, 322
稀疏检索, 173
稀疏率, 434
稀疏专家混合, 456
稀疏专家模型, 484
稀疏注意力, 475
旋转位置编码, 96, 470
学习率预热, 251
学习排序, 167
序列并行, 371
序列打包, 243, 364
序列结束符, 103
序列级 RAG, 172
序列级蒸馏, 441
序列开始符, 103
序列装箱, 288
循环神经网络, 4
训练后量化, 427
训练块, 245

Y

雅可比矩阵, 63
样本单位, 236
研究生级难检索问答, 119
掩码语言建模, 448
验证器, 156, 320
压缩稀疏注意力第二代, 490
页锁定内存, 351
硬标签, 437
因果编码器—解码器, 489
因果掩码, 64
因果注意力, 64
影子流量, 406
隐私, 133
引用计数, 382
一前向一反向调度, 370
意向处理分析, 131
一致性, 355
涌现能力, 6
优势函数, 301
有效吞吐, 394
远程直接内存访问, 352
元数据, 354
元数据过滤, 167
越狱, 136
预填充, 375, 477
预填充—解码分离, 401
预填充与解码分离, 358
余弦衰减, 251
余弦相似度, 58, 164
预训练, 285
语义记忆, 200
语义切分, 160

Z

在策略学习, 306
再量化, 429
灾难性遗忘, 266
噪声预测, 531
增量解码, 375, 477
张量并行, 369
张量核心, 347
占用率, 346
召回率, 124, 169
证据下界, 530
证据忠实度, 179
证据组装, 175
整流线性单元, 78
正弦位置编码, 94
折损累计增益, 169
值, 60
支持集, 15
置换检验, 129
直接内存访问, 351
直接偏好优化, 315
直接提示注入, 136
指令微调, 256
指令遵循评估, 120
智能体, 193
智能体执行与控制系统, 195
制品完整性, 140
知识蒸馏, 419
直通估计器, 427
执行图回放, 415
置信区间, 128
制造执行系统, 189
中间表示蒸馏, 441
中间填充, 291
中间信息利用退化, 480
中文大规模多任务语言理解, 119
中央处理器, 350
重要性采样, 242, 306
状态机, 197
状态价值, 300
专家并行, 371, 465
准确率, 124
准入控制, 392
逐通道量化, 423
逐位置前馈网络, 78
注意力机制, 60
注意力掩码, 519
注意力蒸馏, 441
逐张量量化, 423
自回归, 102
自回归空白填充, 494
自监督学习, 8
自举, 304
自适应低秩适配, 273
自适应矩估计, 249
自一致性, 154, 331
自助重采样, 129
自注意力, 61
纵深防御, 135
总拥有成本, 361
最大内积搜索, 164
最大似然估计, 12
最佳匹配评分, 162
最近舍入, 421
最小哈希, 240
最小权限, 137
组相对策略优化, 318, 323

英文术语索引

Symbols

n-gram Language Model, 4
4-bit NormalFloat, 274
8-bit Floating Point, 347, 428

A

Absolute Positional Embedding, 504
Abstention, 156, 179
Accuracy, 124
Acoustic Token, 523
Action Value, 300
Activation Recomputation, 339, 366
Activation-Aware Weight Quantization, 426
Activation-weighted Low-rank Approximation, 437
Actor–Critic, 305
Adapter Registry, 279
Adapter Routing, 279
Adaptive Low-Rank Adaptation, 273
Adaptive Moment Estimation, 249
Admission Control, 392
Advantage Function, 301
Affine Quantization, 420
Affine Transformation, 13
Agent, 193
Agent Harness, 195
Aging, 393
All-Gather, 353, 367
All-Reduce, 353, 367
All-to-All, 353, 367, 465
American Invitational Mathematics Examination, 119
Answer Canonicalization, 155
Approximate Nearest Neighbor Search, 165
Arithmetic Intensity, 341, 348
Artifact Integrity, 140
Assurance Case, 133
Asymmetric Distance Computation, 166
Attention, 60
Attention Distillation, 441
Attention Mask, 519
Autoregression, 102
Autoregressive Blank Infilling, 494
Auxiliary-Loss-Free Load Balancing, 488
Average Precision, 125

B

Backdoor, 139
Backpressure, 392, 408, 525
Beam Search, 112
Beginning of Sequence, 103
Behavior Policy, 316
Bellman Equation, 300
Benchmark, 117
Best Matching 25, 162
Best-of-K Selection, 156
Best-of-N, 331
Bidirectional Encoder Representations from Transformers, 446
Bilinear Interpolation, 505
Bilingual Evaluation Understudy, 125
Bisection Bandwidth, 352
BitFit, 277
Blue–Green Deployment, 406
Bootstrap, 129
Bootstrapping, 304

Bottleneck Adapter, 275
Brain Floating Point 16, 347

C

Cache Eviction, 382
Canary Release, 406
Capacity Factor, 459
Catastrophic Forgetting, 266
Causal Attention, 64
Causal Encoder-Decoder, 489
Causal Mask, 64
Central Processing Unit, 350
Chain of Thought, 143, 237, 322
Chain-of-Thought Prompting, 153
Change Data Capture, 181
Chat Template, 261
Checkpoint, 252
Chinese Massive Multitask Language Understanding, 119
Chrome DevTools Protocol, 214
Chunked Prefill, 389
Classification Token, 504
Classifier-Free Guidance, 539
Clean Sample Prediction, 532
Clipping, 422
Closed-Loop Load, 394
Cluster Bootstrap, 129
Coalesced Memory Access, 348
Cohen's Kappa, 127
Collective Communication, 353
Combine, 458
Compare-and-Swap, 203, 208
Compensating Transaction, 209
Compressed Sparse Attention 2, 490
Computer Use, 213
Conditional Computation, 454
Conditional Independence, 19
Conditional Memory, 489
Confidence Interval, 128
Confusion Matrix, 124
Connector, 516
Consistency, 355
Containment, 239
Context Compression, 176, 479
Context Parallelism, 371
Context Window, 468
Contextual Calibration, 153
Continued Pretraining, 285
Continuous Batching, 388
Contrastive Language-Image Pre-training, 514
Contrastive Learning, 163
Control Plane, 357, 396
Coordinated Omission, 394
Copy-on-Write, 382
Cosine Decay, 251
Cosine Similarity, 58, 164
Counterfactual Evaluation, 184
Credit Assignment, 303
Cross Encoder, 167
Cross-Attention, 61
Cross-attention Fusion, 519
Cross-encoder, 174
Cross-Entropy, 12
Cross-modal Alignment, 514
Curriculum Learning, 243, 290

D

Data Augmentation, 509
Data Contract, 180
Data Engineering, 235
Data Freshness, 179
Data Lineage, 237
Data Loss Prevention, 141
Data Plane, 357, 396
Data Poisoning, 139
Deadline, 399
Decode, 477
Decoder-only, 84
Decoupled Weight Decay, 250
Defense in Depth, 135
Deletion Lineage, 247
Denoising Diffusion Implicit Model, 534
Denoising Diffusion Probabilistic Model, 526
Dense Mixture of Experts, 455

Dense Model, 484
Dense Retrieval, 163, 173
Dequantization, 421
Design Effect, 236
Deterministic Transformation, 238
Device Affinity, 350
Differential Privacy, 142
Diffusion Model, 526
Diffusion Transformer, 539
Digital Signature, 140
Direct Memory Access, 351
Direct Preference Optimization, 315
Direct Prompt Injection, 136
Discounted Cumulative Gain, 169
Dispatch, 458
Distributed Data Parallel, 366
Domain Ontology, 188
Double Quantization, 274
Draining, 400
Dropout, 63, 91, 229, 272
Dual Encoder, 163
Durability, 355
Durable Execution, 208
Dynamic Padding, 264
Dynamic Quantization, 424
Dynamic Resolution, 514

E

Embedding, 56
Emergent Abilities, 6
Encoder–Decoder, 84
Encoder-only, 84
End of Sequence, 103
Enterprise Resource Planning, 189
Entropy, 12
Episodic Memory, 200
Estimand, 117
Euclidean Distance, 164
Evaluation Contamination, 139, 239
Evaluation Harness, 118
Evidence Assembly, 175
Evidence Faithfulness, 179
Evidence Lower Bound, 530
Exact Deduplication, 239
Exact Match, 125
Exact Nearest Neighbor Search, 164
Exactly-Once, 245
Example Bank, 151
Executable Verifier, 126
Execution Graph Replay, 415
Expected Calibration Error, 125
Expert Parallelism, 371, 465
Explicit Congestion Notification, 352
Exposure Bias, 106

F

F1 Score, 124
Failure Domain, 397
Fake Quantization, 427
False Negative, 145, 164, 516
False Positive, 145
Feed-Forward Network, 78
Fencing Token, 209
Few-Shot Prompting, 150
Fill-in-the-Middle, 291
Fine-Grained Expert, 464
First-Come, First-Served, 393
Floating-point Operations, 336
Flow Matching, 541
Forward Marginal Distribution, 528
Forward Process, 527
Freshness, 200
Full Fine-Tuning, 268
Fully Sharded Data Parallel, 368
Fused Dequantization, 429

G

Gang Scheduling, 359
Gated Linear Unit, 79
Gating Network, 454
Gaussian Error Linear Unit, 78
GELU-Gated Linear Unit, 79
Generalized Advantage Estimation, 304
Generative Pre-trained Transformer, 103
Global Average Pooling, 506
Goodput, 394

GPTQ, 424
Gradient Accumulation, 365
Gradient Norm Clipping, 252
Graduate-Level Google-Proof Question Answering, 119
Graph Compilation, 415
Graphics Processing Unit, 346
Greedy Decoding, 111
Grounding, 213
Group Relative Policy Optimization, 318, 323
Group Split, 242
Group-Wise Quantization, 423
Grouped-Query Attention, 72, 379

H

Half Precision Floating Point, 347
Hallucination, 107
Hard Label, 437
Hard Negative, 164
Head-of-Line Blocking, 393
Hessian Matrix, 432
Hierarchical Navigable Small World, 165
High Bandwidth Memory, 347
Hit Rate, 169
Humanity's Last Exam, 119
Hybrid Retrieval, 166, 174

I

Idempotency, 138, 203, 246, 399
Idempotency Key, 209
Immutable Artifact, 395
Importance Sampling, 242, 306
In-Batch Negative, 164, 516
In-Context Learning, 5, 150
Incident Response, 146
Incremental Decoding, 375
Index Snapshot, 168
Indirect Prompt Injection, 136
Inductive Bias, 9, 508
InfiniBand, 352
Information Entropy, 19
Information Retrieval, 160
Infused Adapter by Inhibiting and Amplifying Inner Activations, 275
Input/Output Operations Per Second, 354
Instruction Tuning, 256
Instruction-Following Evaluation, 120
Insufficient Exploration, 329
Intention-to-Treat Analysis, 131
Inter-Token Latency, 387
Intermediate Representation Distillation, 441
Inverse Document Frequency, 162
Inverted File Index, 165
Inverted Index, 161
Iterative Retrieval, 177

J

Jaccard Similarity, 240
Jacobian Matrix, 63
Jailbreak, 136

K

Key, 60
Key–Value Cache, 473
Key-Value Cache, 357, 376
Key-Value Cache Quantization, 420
Knowledge Distillation, 419
Kullback–Leibler Divergence, 12

L

Label Cardinality, 408
Latent Diffusion, 540
Latent Variable, 171
Layer Normalization, 81
Learning Rate Warmup, 251
Learning to Rank, 167
Least Privilege, 137
Leave-one-out Baseline, 324
Length Bucketing, 364, 389
Linear Probing, 509
Liveness Probe, 400
LLM-as-a-Judge, 127
Locality-Sensitive Hashing, 240
Logits, 104
Long Short-Term Memory, 4

Long-Term Memory, 200
Loss Mask, 519
Loss Scaling, 252
Lost in the Middle, 480
Low-Precision Training, 427
Low-Rank Adaptation, 269
Low-rank Factorization, 436

M

Macro Average, 124
Magnitude Pruning, 431
Manifest, 246
Manufacturing Execution System, 189
Marginal Utility, 157
Markov Decision Process, 298
Masked Language Modeling, 448
Massive Multi-discipline Multimodal Understanding, 120
Massive Multitask Language Understanding, 119
Matrix Absorption, 380
Maximum Inner Product Search, 164
Maximum Likelihood Estimation, 12
Mean Reciprocal Rank, 169
Membership Inference, 141
Metadata, 354
Metadata Filtering, 167
Micro Average, 124
MinHash, 240
Mixed Precision, 366
Mixture of Experts, 454
Mixup, 509
Modality, 512
Model Adapter, 498
Model Artifact, 395
Model Cascade, 404
Model Compression, 419
Model Context Protocol, 206
Model Extraction, 142
Model Routing, 403
Momentum, 248
Monte Carlo Estimation, 304
Multi-Agent System, 201
Multi-Head Attention, 68, 76, 378
Multi-Head Latent Attention, 380
Multi-head Latent Attention, 487
Multi-Query Attention, 72, 378
Multi-Round Coreference Resolution, 120
Multi-tenant Isolation, 360
Multi-Token Prediction, 488

N

N-gram Language Model, 40
Named Entity Recognition, 452
Negative Cache, 183
Next Sentence Prediction, 451
Noise Prediction, 531
Non-Uniform Memory Access, 350
Non-Volatile Memory Express, 354
Normalized Discounted Cumulative Gain, 169
Nucleus Sampling, 110

O

Observability, 407
Observation, 193
Occupancy, 346
Off-Policy Learning, 306
Offload, 369
On-Policy Learning, 306
One-Forward-One-Backward, 370
Open Weights, 484
Open-Loop Load, 394
Operator Fusion, 414
Outcome Reward Model, 321
Outcome Supervision, 321
Oversubscription Ratio, 352

P

P-Tuning, 276
P-Tuning v2, 276
Paged KV Cache, 381
Paged Optimizer, 274
Parameter-Efficient Fine-Tuning, 264, 268
Pareto Frontier, 131
Parse Failure, 155

Partially Observable Markov Decision Process, 193, 299
Pass at k, 331
Patch, 501, 513
Patch Embedding, 502
Patchification, 502
Per-Channel Quantization, 423
Per-Tensor Quantization, 423
Peripheral Component Interconnect Express, 350
Permutation Equivariance, 93
Permutation Invariance, 93
Permutation Test, 129
Perplexity, 22, 105, 125, 287
Personally Identifiable Information, 140, 239
Pinned Memory, 351
Pipeline Parallelism, 370
Plan-and-Execute Architecture, 196
Policy Lag, 329
Position Interpolation, 470
Post-Layer Normalization, 82
Post-Training Quantization, 427
Posting List, 161
Power Usage Effectiveness, 361
Pre-Layer Normalization, 82, 506
Precision, 124
Precision–Recall Curve, 125
Preemption, 359, 393
Preference Pair, 311
Prefill, 375, 477
Prefill/Decode Disaggregation, 358
Prefill–Decode Disaggregation, 401
Prefix Cache, 382
Prefix Decoder, 86
Prefix Fusion, 518
Prefix Tuning, 276
Pretraining, 285
Priority-based Flow Control, 352
Privacy, 133
Probability Calibration, 124
Probability Density Function, 14
Probability Mass Function, 14
Procedural Memory, 200
Process Reward Model, 321
Process Supervision, 321
Product Quantization, 166
Projector, 516
Prompt Injection, 136, 182
Prompt Tuning, 276
Provenance Span, 245
Proximal Policy Optimization, 307
Pruning, 431

Q

Quantization Calibration, 424
Quantization Scale, 420
Quantization-Aware Training, 427
Quantized Low-Rank Adaptation, 274
Query, 60
Query Decomposition, 177
Query Rewriting, 176
Querying Transformer, 517

R

RAG-Sequence, 172
RAG-Token, 172
Random Variable, 14
Randomized Controlled Experiment, 131
Rank, 365
RDMA over Converged Ethernet, 352
Re-ranking, 174
Reactive Architecture, 196
Readiness Probe, 400
Reasoning and Acting, 197
Reasoning Distillation, 323
Reasoning Effort, 330
Recall, 124, 169
Recall-Oriented Understudy for Gisting Evaluation, 126
Receiver Operating Characteristic, 125
Reciprocal Rank Fusion, 166
Reconciliation, 209
Recovery Training, 443
Rectified Linear Unit, 78
Recurrent Neural Network, 4

Red Teaming, 146
Reduce-Scatter, 353, 367
Reference Counting, 382
Reference Policy, 316
Reinforcement Learning, 9, 298
Reinforcement Learning from Human Feedback, 313
Reinforcement Learning with Verifiable Rewards, 320
Rejection Sampling Fine-Tuning, 322
Reliability, 133
Remote Direct Memory Access, 352
Replay Data, 266
Requantization, 429
Reranking, 167
Resampler, 517
Retrieval-Augmented Generation, 171, 479
Reverse Conditional Posterior, 529
Reverse Process, 529
Reward Hacking, 319, 328
Ring Buffer, 384
Roofline Model, 348
Root Mean Square Normalization, 82
Rotary Position Embedding, 96, 470
Round to Nearest, 421
Router, 454
Rubric, 127

S

Safety, 133
Sampling Rate, 522
Sampling Unit, 236
Sandbox Verification, 333
Saturation Error, 422
Scaled Dot-Product Attention, 61
Scaling Law, 6, 292
Schema, 138, 237
Schema Adapter, 237
Score Function, 539
Score-Function Estimation, 302
Security, 133
Self-Attention, 61
Self-Consistency, 154, 331
Self-Supervised Learning, 8
Semantic Chunking, 160
Semantic Memory, 200
Semi-structured Sparsity, 434
Sequence Packing, 243, 288, 364
Sequence Parallelism, 371
Sequence-level Distillation, 441
Server-Sent Events, 398
Service Degradation, 404
Shadow Traffic, 406
Shape Bucketing, 415
Shard, 245
Shared Expert, 464
Shingle, 240
Short-Time Fourier Transform, 522
Side Effect, 203
Sigmoid Linear Unit, 79
Signal-to-Noise Ratio, 528
Simplified Noise Loss, 531
Single Instruction, Multiple Threads, 346
Singular Value Decomposition, 435
Sinusoidal Positional Encoding, 94
Skill, 196
Slice Evaluation, 130
Sliding-Window Attention, 418, 474
SmoothQuant, 426
Snapshot Manifest, 180
Soft Label, 437
Soft Mixture of Experts, 465
Soft Prompt, 276
Softmax, 21, 63
Software Bill of Materials, 140
Sparse Attention, 475
Sparse Mixture of Experts, 456
Sparse Mixture-of-Experts Model, 484
Sparse Retrieval, 173
Sparse Reward, 322
Sparsity, 434
Speculative Decoding, 415
Startup Probe, 400
Startup Storm, 355
State Machine, 197

State Value, 300
Static Batching, 388
Static Quantization, 424
Straight-Through Estimator, 427
Streaming, 245
Streaming Causality, 524
Streaming Multiprocessor, 346
Strong Scaling, 373
Structured Pruning, 433
Supervised Fine-Tuning, 256
Supervised Learning, 8
Supervisor Architecture, 201
Support, 15
Surrogate Objective, 306
Swish-Gated Linear Unit, 79
Symmetric Quantization, 421
Synthetic Data, 243
Synthetic Instruction Data, 262

T

Task Unit, 186
Teacher Forcing, 90, 104, 285
Temperature Scaling, 109
Temperature-Based Mixing, 242
Temporal Alignment, 524
Temporal Difference, 304
Tensor Core, 347
Tensor Parallelism, 369
Term, 160
Test Contamination, 333
Test-Time Compute, 157
Test-time Compute, 331
Threat Model, 134
Time per Output Token, 363, 387, 431, 478
Time to First Token, 363, 387, 431, 478
Time to Live, 183
Time-of-Check to Time-of-Use, 138, 211
Token F1, 125
Token Materialization, 243
Token-level Distillation, 440
Tombstone, 168
Tool Calling, 203
Top-k Sampling, 110
Total Cost of Ownership, 361
Training Block, 245
Trajectory, 298
Trajectory Replay, 218
Transactional Outbox, 209
Translation Equivariance, 508
Trust Boundary, 134
Two-Dimensional Positional Interpolation, 505
Typed Record, 237

U

U-Net, 538
Unstructured Pruning, 431
Unsupervised Learning, 8

V

Value, 60
Value of Information, 199
Variational Autoencoder, 540
Vector–Jacobian Product, 29
Velocity Parameterization, 532
Verbalizer, 152
Verifier, 156, 320
Vision Transformer, 501, 513
Vision-Language Model, 512
Visual Instruction Tuning, 521

W

Warp, 346
Weak Scaling, 373
WebDriver BiDi, 214
Weight Tying, 57
Weight-Decomposed Low-Rank Adaptation, 273
Weight-Only Quantization, 420
Wilson Interval, 128
Workflow, 197
Working Memory, 200

Z

Zero Point, 420
Zero Redundancy Optimizer, 368
Zero-Shot Prompting, 150
Zero-variance Group, 324

参考文献

[1] Claude E. Shannon. “A Mathematical Theory of Communication”. In: *The Bell System Technical Journal* 27 (1948), pp. 379–423, 623–656. URL: https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf.

[2] Yoshua Bengio et al. “A Neural Probabilistic Language Model”. In: *Journal of Machine Learning Research* 3 (2003), pp. 1137–1155. URL: https://www.jmlr.org/papers/v3/bengio03a.html (visited on 09/11/2026).

[3] Tomáš Mikolov et al. “Recurrent Neural Network Based Language Model”. In: *Proceedings of Interspeech 2010*. 2010, pp. 1045–1048. DOI: 10.21437/Interspeech.2010-343. URL: https://www.isca-archive.org/interspeech_2010/mikolov10_interspeech.html (visited on 09/11/2026).

[4] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735 (visited on 09/11/2026).

[5] Matthew E. Peters et al. *Deep Contextualized Word Representations*. 2018. arXiv: 1802.05365. URL: https://arxiv.org/abs/1802.05365v2 (visited on 09/11/2026).

[6] Ashish Vaswani et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems* 30 (2017). arXiv: 1706.03762. URL: https://arxiv.org/abs/1706.03762.

[7] Alec Radford et al. *Improving Language Understanding by Generative Pre-Training*. OpenAI, 2018. URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf (visited on 09/11/2026).

[8] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of NAACL-HLT*. 2019. URL: https://aclanthology.org/N19-1423.

[9] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. URL: https://arxiv.org/abs/2005.14165.

[10] Jared Kaplan et al. *Scaling Laws for Neural Language Models*. 2020. URL: https://arxiv.org/abs/2001.08361.

[11] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, et al. *Training Compute-Optimal Large Language Models*. 2022. URL: https://arxiv.org/abs/2203.15556.

[12] Long Ouyang, Jeff Wu, Xu Jiang, et al. *Training language models to follow instructions with human feedback*. 2022. URL: https://arxiv.org/abs/2203.02155.

[13] Haotian Liu et al. *Visual Instruction Tuning*. 2023. arXiv: 2304.08485. URL: https://arxiv.org/abs/2304.08485.

[14] Shunyu Yao et al. "ReAct: Synergizing Reasoning and Acting in Language Models". In: *International Conference on Learning Representations*. 2023. URL: https://arxiv.org/abs/2210.03629.

[15] DeepSeek-AI. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. 2025. URL: https://arxiv.org/abs/2501.12948v2 (visited on 09/08/2026).

[16] Jason Wei et al. *Emergent Abilities of Large Language Models*. 2022. arXiv: 2206.07682. URL: https://arxiv.org/abs/2206.07682v2 (visited on 09/11/2026).

[17] Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. *Are Emergent Abilities of Large Language Models a Mirage?* 2023. arXiv: 2304.15004. URL: https://arxiv.org/abs/2304.15004v2 (visited on 09/11/2026).

[18] Rico Sennrich, Barry Haddow, and Alexandra Birch. "Neural Machine Translation of Rare Words with Subword Units". In: *Proceedings of ACL*. 2016, pp. 1715–1725. DOI: 10.18653/v1/P16-1162. URL: https://aclanthology.org/P16-1162/.

[19] Google Research. *BERT: tokenization.py*. 2018. URL: https://github.com/google-research/bert/blob/master/tokenization.py (visited on 09/08/2026).

[20] Taku Kudo. "Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates". In: *Proceedings of ACL*. 2018, pp. 66–75. DOI: 10.18653/v1/P18-1007. URL: https://aclanthology.org/P18-1007/.

[21] Google. *SentencePiece: Unsupervised Text Tokenizer*. URL: https://github.com/google/sentencepiece (visited on 09/08/2026).

[22] Ofir Press and Lior Wolf. "Using the Output Embedding to Improve Language Models". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, 2017, pp. 157–163. URL: https://aclanthology.org/E17-2025/.

[23] Noam Shazeer. *Fast Transformer Decoding: One Write-Head is All You Need*. 2019. URL: https://arxiv.org/abs/1911.02150.

[24] Joshua Ainslie et al. "GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2023, pp. 4895–4901. DOI: 10.18653/v1/2023.emnlp-main.298. URL: https://aclanthology.org/2023.emnlp-main.298/.

[25] Dan Hendrycks and Kevin Gimpel. *Gaussian Error Linear Units (GELUs)*. 2016. URL: https://arxiv.org/abs/1606.08415.

[26] Noam Shazeer. *GLU Variants Improve Transformer*. 2020. URL: https://arxiv.org/abs/2002.05202.

[27] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. 2016. URL: https://arxiv.org/abs/1607.06450.

[28] Biao Zhang and Rico Sennrich. *Root Mean Square Layer Normalization*. 2019. URL: https://arxiv.org/abs/1910.07467.

[29] Ruibin Xiong et al. "On Layer Normalization in the Transformer Architecture". In: *Proceedings of the 37th International Conference on Machine Learning*. 2020, pp. 10524–10533. URL: https://proceedings.mlr.press/v119/xiong20b.html.

[30] Jianlin Su et al. *RoFormer: Enhanced Transformer with Rotary Position Embedding*. Apr. 20, 2021. arXiv: 2104.09864. URL: https://arxiv.org/abs/2104.09864.

[31] Samy Bengio et al. *Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks*. 2015. URL: https://arxiv.org/abs/1506.03099.

[32] Ziwei Ji et al. "Survey of Hallucination in Natural Language Generation". In: *ACM Computing Surveys* 55.12 (2023), pp. 1–38. DOI: 10.1145/3571730. URL: https://doi.org/10.1145/3571730.

[33] Ari Holtzman et al. “The Curious Case of Neural Text Degeneration”. In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=rygGQyrFvH.

[34] David Rein et al. *GPQA: A Graduate-Level Google-Proof Question and Answer Benchmark*. 2024. URL: https://github.com/idavidrein/gpqa (visited on 09/08/2026).

[35] Center for AI Safety, Scale AI, and HLE Contributors Consortium. *Humanity's Last Exam*. 2026. URL: https://lastexam.ai/ (visited on 09/08/2026).

[36] Epoch AI. *FrontierMath Tier 4 (v2)*. 2026. URL: https://epoch.ai/benchmarks/frontiermath-tier-4-v2 (visited on 09/08/2026).

[37] Dan Hendrycks et al. *Measuring Massive Multitask Language Understanding*. 2021. arXiv: 2009.03300. URL: https://arxiv.org/abs/2009.03300.

[38] Yubo Wang, Xueguang Ma, Ge Zhang, et al. *MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark*. 2024. arXiv: 2406.01574. URL: https://arxiv.org/abs/2406.01574.

[39] Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, et al. *C-Eval: A Multi-Level Multi-Discipline Chinese Evaluation Suite for Foundation Models*. 2023. arXiv: 2305.08322. URL: https://arxiv.org/abs/2305.08322.

[40] Haonan Li et al. *CMMLU: Measuring massive multitask language understanding in Chinese*. 2023. arXiv: 2306.09212. URL: https://arxiv.org/abs/2306.09212.

[41] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. *Training Verifiers to Solve Math Word Problems*. 2021. arXiv: 2110.14168. URL: https://arxiv.org/abs/2110.14168.

[42] Dan Hendrycks et al. *Measuring Mathematical Problem Solving With the MATH Dataset*. 2021. arXiv: 2103.03874. URL: https://arxiv.org/abs/2103.03874.

[43] Mathematical Association of America. *MAA Invitational Competitions: AIME*. URL: https://maa.org/maa-invitational-competitions/ (visited on 09/08/2026).

[44] Mark Chen et al. *Evaluating Large Language Models Trained on Code*. July 7, 2021. arXiv: 2107.03374. URL: https://arxiv.org/abs/2107.03374 (visited on 09/08/2026).

[45] Naman Jain et al. *LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code*. 2024. arXiv: 2403.07974. URL: https://arxiv.org/abs/2403.07974.

[46] Xiang Yue, Yuansheng Ni, Kai Zhang, et al. *MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI*. 2024. arXiv: 2311.16502. URL: https://arxiv.org/abs/2311.16502.

[47] Yushi Bai, Xin Lv, Jiajie Zhang, et al. "LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2024. DOI: 10.18653/v1/2024.acl-long.172. URL: https://aclanthology.org/2024.acl-long.172/.

[48] Yushi Bai, Shangqing Tu, Jiajie Zhang, et al. *LongBench v2: Towards Deeper Understanding and Reasoning on Realistic Long-context Multitasks*. 2024. arXiv: 2412.15204. URL: https://arxiv.org/abs/2412.15204.

[49] OpenAI. *OpenAI MRCR: Long context multiple needle in a haystack benchmark*. URL: https://huggingface.co/datasets/openai/mrcr (visited on 09/08/2026).

[50] Jeffrey Zhou et al. *Instruction-Following Evaluation for Large Language Models*. 2023. arXiv: 2311.07911. URL: https://arxiv.org/abs/2311.07911.

[51] SWE-bench Team. *SWE-bench Verified*. URL: https://www.swebench.com/verified.html (visited on 09/08/2026).

[52] OpenAI. *GPT-6 Astra: A new generation of intelligence*. 2026. URL: https://openai.com/index/gpt-6-astra/ (visited on 09/08/2026).

[53] OpenAI. *BrowseComp: a benchmark for browsing agents*. 2025. URL: https://openai.com/index/browsecomp/ (visited on 09/08/2026).

[54] OSWorld Team. *OSWorld 2.0: Benchmarking computer-use agents on long-horizon real-world tasks*. 2026. URL: https://osworld-v2.xlang.ai/ (visited on 09/08/2026).

[55] ARC Prize Foundation. *ARC-AGI-3*. 2026. URL: https://arcprize.org/arc-agi/3 (visited on 09/08/2026).

[56] Anthropic. *Introducing Claude Fable 5.1 and Claude Mythos 5.1*. 2026. URL: https://www.anthropic.com/claude-fable-and-mythos-5-1 (visited on 09/08/2026).

[57] Anthropic. *Claude Fable 5.1 & Claude Mythos 5.1 System Card*. 2026. URL: https://www-cdn.anthropic.com/0339e6a7c5c7b87f5c07798616dc32c215d14235/Claude%20Fable%205.1%20%26%20Claude%20Mythos%205.1%20System%20Card.pdf (visited on 09/08/2026).

[58] Kishore Papineni et al. "Bleu: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://aclanthology.org/P02-1040/.

[59] Chin-Yew Lin. "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Text Summarization Branches Out*. 2004, pp. 74–81. URL: https://aclanthology.org/W04-1013/.

[60] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, et al. *Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena*. 2023. arXiv: 2306.05685. URL: https://arxiv.org/abs/2306.05685.

[61] Jacob Cohen. "A Coefficient of Agreement for Nominal Scales". In: *Educational and Psychological Measurement* 20.1 (1960), pp. 37–46. DOI: 10.1177/001316446002000104. URL: https://journals.sagepub.com/doi/abs/10.1177/001316446002000104.

[62] Edwin B. Wilson. "Probable Inference, the Law of Succession, and Statistical Inference". In: *Journal of the American Statistical Association* 22.158 (1927), pp. 209–212. DOI: 10.1080/01621459.1927.10502953. URL: https://doi.org/10.1080/01621459.1927.10502953.

[63] Bradley Efron. "Bootstrap Methods: Another Look at the Jackknife". In: *The Annals of Statistics* 7.1 (1979), pp. 1–26. DOI: 10.1214/aos/1176344552. URL: https://doi.org/10.1214/aos/1176344552.

[64] National Institute of Standards and Technology. *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. Tech. rep. NIST AI 100-1. National Institute of Standards and Technology, 2023. DOI: 10.6028/NIST.AI.100-1. URL: https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.100-1.pdf.

[65] Chloe Autio et al. *Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile*. Tech. rep. NIST AI 600-1. National Institute of Standards and Technology, 2024. DOI: 10.6028/NIST.AI.600-1. URL: https://www.nist.gov/publications/artificial-intelligence-risk-management-framework-generative-artificial-intelligence.

[66] OWASP Gen AI Security Project. *LLM01:2025 Prompt Injection*. 2025. URL: https://genai.owasp.org/llmrisk/llm01-prompt-injection/ (visited on 09/08/2026).

[67] Kai Greshake et al. *Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection*. 2023. eprint: 2302.12173. URL: https://arxiv.org/abs/2302.12173.

[68] OWASP Gen AI Security Project. *LLM06:2025 Excessive Agency*. 2025. URL: https://genai.owasp.org/llmrisk/llm062025-excessive-agency/ (visited on 09/08/2026).

[69] OWASP Gen AI Security Project. *LLM03:2025 Supply Chain*. 2025. URL: https://genai.owasp.org/llmrisk/llm032025-supply-chain/ (visited on 09/08/2026).

[70] Safetensors Contributors. *Safetensors: Format and Rationale*. URL: https://github.com/safetensors/safetensors (visited on 09/08/2026).

[71] Nicholas Carlini et al. "Extracting Training Data from Large Language Models". In: *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 2021, pp. 2633–2650. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting (visited on 09/11/2026).

[72] Martín Abadi et al. *Deep Learning with Differential Privacy*. 引用 v2；首次提交 2016-07-01。Oct. 24, 2016. arXiv: 1607.00133v2. URL: https://arxiv.org/abs/1607.00133v2 (visited on 09/11/2026).

[73] Anthropic. *Detecting and Preventing Distillation Attacks*. 厂商公告；引用防护方向，不将归因与防护效果视为独立验证。Feb. 23, 2026. URL: https://www.anthropic.com/news/detecting-and-preventing-distillation-attacks (visited on 09/11/2026).

[74] Anthropic. *Thinking*. Claude Messages API 动态官方文档，按访问日核查；涉及 thinking、signature、display 与 redacted_thinking. 页面未标明固定发布日期；未核实底层密码算法。URL: https://platform.claude.com/docs/en/build-with-claude/thinking (visited on 09/11/2026).

[75] Alexander Nelson et al. *Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile*. Tech. rep. SP 800-61 Rev. 3. National Institute of Standards and Technology, 2025. URL: https://csrc.nist.gov/pubs/sp/800/61/r3/final.

[76] Yao Lu et al. *Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity*. 2022. URL: https://aclanthology.org/2022.acl-long.556/.

[77] Sewon Min et al. *Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?* 2022. URL: https://aclanthology.org/2022.emnlp-main.759/.

[78] Zihao Zhao et al. *Calibrate Before Use: Improving Few-shot Performance of Language Models*. 2021. URL: https://proceedings.mlr.press/v139/zhao21c.html.

[79] Jason Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2022. URL: https://arxiv.org/abs/2201.11903.

[80] Xuezhi Wang et al. *Self-Consistency Improves Chain of Thought Reasoning in Language Models*. Mar. 21, 2022. arXiv: 2203.11171. URL: https://arxiv.org/abs/2203.11171 (visited on 09/08/2026).

[81] Karl Cobbe et al. *Training Verifiers to Solve Math Word Problems*. 2021. URL: https://arxiv.org/abs/2110.14168.

[82] Stephen Robertson and Hugo Zaragoza. "The Probabilistic Relevance Framework: BM25 and Beyond". In: *Foundations and Trends in Information Retrieval* 3.4 (2009), pp. 333–389. DOI: 10.1561/1500000019. URL: https://www.staff.city.ac.uk/~sbrp622/papers/foundations_bm25_review.pdf.

[83] Vladimir Karpukhin et al. "Dense Passage Retrieval for Open-Domain Question Answering". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. 2020, pp. 6769–6781. DOI: 10.18653/v1/2020.emnlp-main.550. URL: https://aclanthology.org/2020.emnlp-main.550/.

[84] Yu. A. Malkov and D. A. Yashunin. *Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs*. 2018. arXiv: 1603.09320. URL: https://arxiv.org/abs/1603.09320v4.

[85] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. "Product Quantization for Nearest Neighbor Search". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.1 (2011), pp. 117–128. DOI: 10.1109/TPAMI.2010.57. URL: https://inria.hal.science/inria-00514462v2.

[86] Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. “Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods”. In: *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2009, pp. 758–759. DOI: 10.1145/1571941.1572114. URL: https://cormack.uwaterloo.ca/cormack/cormacksigir09-rrf.pdf.

[87] Kalervo Järvelin and Jaana Kekäläinen. “Cumulated Gain-based Evaluation of IR Techniques”. In: *ACM Transactions on Information Systems* 20.4 (2002), pp. 422–446. DOI: 10.1145/582415.582418. URL: https://researchportal.tuni.fi/en/publications/cumulated-gain-based-evaluation-of-ir-techniques.

[88] Patrick Lewis et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. 2020. eprint: 2005.11401. URL: https://arxiv.org/abs/2005.11401.

[89] Rodrigo Nogueira and Kyunghyun Cho. *Passage Re-ranking with BERT*. 2019. eprint: 1901.04085. URL: https://arxiv.org/abs/1901.04085.

[90] Xinbei Ma et al. *Query Rewriting for Retrieval-Augmented Large Language Models*. 2023. eprint: 2305.14283. URL: https://arxiv.org/abs/2305.14283.

[91] Zhengbao Jiang et al. *Active Retrieval Augmented Generation*. 2023. eprint: 2305.06983. URL: https://arxiv.org/abs/2305.06983.

[92] Tianyu Gao et al. *Enabling Large Language Models to Generate Text with Citations*. 2023. eprint: 2305.14627. URL: https://arxiv.org/abs/2305.14627.

[93] Joon Sung Park et al. *Generative Agents: Interactive Simulacra of Human Behavior*. 2023. eprint: 2304.03442. URL: https://arxiv.org/abs/2304.03442.

[94] Qingyun Wu et al. *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation*. 2023. eprint: 2308.08155. URL: https://arxiv.org/abs/2308.08155.

[95] Model Context Protocol. *Architecture: Specification 2025-06-18*. June 18, 2025. URL: https://modelcontextprotocol.io/specification/2025-06-18/architecture (visited on 09/08/2026).

[96] Amazon Web Services. *Transactional outbox pattern*. URL: https://docs.aws.amazon.com/prescriptive-guidance/latest/cloud-design-patterns/transactional-outbox.html (visited on 09/08/2026).

[97] Hector Garcia-Molina and Kenneth Salem. “Sagas”. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 1987, pp. 249–259. URL: https://www.cs.cornell.edu/andru/cs711/2002fa/reading/sagas.pdf.

[98] OpenAI. *Unlocking the Codex harness: how we built the App Server*. URL: https://openai.com/index/unlocking-the-codex-harness/ (visited on 09/08/2026).

[99] DeepSeek. *DeepSeek Harness developer preview*. URL: https://www.deepseek.com/harness/en/ (visited on 09/08/2026).

[100] OpenClaw. *Gateway architecture*. URL: https://docs.openclaw.ai/concepts/architecture (visited on 09/08/2026).

[101] Nous Research. *Hermes Agent*. URL: https://github.com/NousResearch/hermes-agent (visited on 09/08/2026).

[102] OpenAI. *Computer use*. URL: https://developers.openai.com/api/docs/guides/tools-computer-use (visited on 09/08/2026).

[103] Chrome DevTools Team. *Chrome DevTools Protocol*. URL: https://chromedevtools.github.io/devtools-protocol/ (visited on 09/08/2026).

[104] Microsoft. *Playwright BrowserType API*. URL: https://playwright.dev/docs/api/class-browsertype (visited on 09/08/2026).

[105] W3C. *WebDriver BiDi*. URL: https://www.w3.org/TR/webdriver-bidi/ (visited on 09/08/2026).

[106] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Vol. 9. Proceedings of Machine Learning Research. 2010, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html.

[107] Kaiming He et al. “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1026–1034. URL: https://www.cv-foundation.org/openaccess/content_iccv_2015/html/He_Delving_Deep_into_ICCV_2015_paper.html.

[108] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: https://jmlr.org/papers/v15/srivastava14a.html.

[109] Andrei Z. Broder. “On the resemblance and containment of documents”. In: *Proceedings of Compression and Complexity of SEQUENCES 1997*. 1997, pp. 21–29. DOI: 10.1109/SEQUEN.1997.666900. URL: https://www.cs.princeton.edu/courses/archive/spr05/cos598E/bib/broder97resemblance.pdf.

[110] Katherine Lee et al. “Deduplicating Training Data Makes Language Models Better”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022, pp. 8424–8445. DOI: 10.18653/v1/2022.acl-long.577. URL: https://aclanthology.org/2022.acl-long.577/.

[111] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations*. 2015. URL: https://arxiv.org/abs/1412.6980.

[112] Ilya Loshchilov and Frank Hutter. “Decoupled Weight Decay Regularization”. In: *International Conference on Learning Representations*. 2019. URL: https://arxiv.org/abs/1711.05101.

[113] Yizhong Wang et al. “Self-Instruct: Aligning Language Models with Self-Generated Instructions”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023, pp. 13484–13508. DOI: 10.18653/v1/2023.acl-long.754. URL: https://aclanthology.org/2023.acl-long.754/.

[114] Chunting Zhou et al. *LIMA: Less Is More for Alignment*. 2023. URL: https://arxiv.org/abs/2305.11206.

[115] Edward J. Hu et al. “LoRA: Low-Rank Adaptation of Large Language Models”. In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=nZeVKeeFYf9.

[116] Qingru Zhang et al. *AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning*. 2023. URL: https://arxiv.org/abs/2303.10512.

[117] Shih-Yang Liu et al. *DoRA: Weight-Decomposed Low-Rank Adaptation*. 2024. URL: https://proceedings.mlr.press/v235/liu24bn.html.

[118] Tim Dettmers et al. *QLoRA: Efficient Finetuning of Quantized LLMs*. 2023. URL: https://arxiv.org/abs/2305.14314.

[119] Neil Houlsby et al. *Parameter-Efficient Transfer Learning for NLP*. 2019. URL: https://proceedings.mlr.press/v97/houlsby19a.html.

[120] Haokun Liu et al. *Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning*. 2022. URL: https://arxiv.org/abs/2205.05638.

[121] Brian Lester, Rami Al-Rfou, and Noah Constant. *The Power of Scale for Parameter-Efficient Prompt Tuning*. 2021. URL: https://aclanthology.org/2021.emnlp-main.243/.

[122] Xiang Lisa Li and Percy Liang. *Prefix-Tuning: Optimizing Continuous Prompts for Generation*. 2021. URL: https://aclanthology.org/2021.acl-long.353/.

[123] Xiao Liu et al. *GPT Understands, Too*. 2021. URL: https://arxiv.org/abs/2103.10385.

[124] Xiao Liu et al. *P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks*. 2022. URL: https://aclanthology.org/2022.acl-short.8/.

[125] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. *BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models*. 2022. URL: https://aclanthology.org/2022.acl-short.1/.

[126] Hugging Face. *Transformers*. Hugging Face. URL: https://github.com/huggingface/transformers (visited on 09/08/2026).

[127] Hugging Face. *PEFT*. Version 0.20.0. URL: https://github.com/huggingface/peft/tree/v0.20.0 (visited on 09/11/2026).

[128] Hugging Face. *TRL: SFT Trainer*. Version 1.13.0. URL: https://github.com/huggingface/trl/blob/v1.13.0/docs/source/sft_trainer.md (visited on 09/11/2026).

[129] LLaMA-Factory contributors. *LLaMA-Factory*. URL: https://github.com/hiyouga/LlamaFactory (visited on 09/11/2026).

[130] Axolotl maintainers and contributors. *Axolotl: Open Source LLM Post-Training*. URL: https://github.com/axolotl-ai-cloud/axolotl (visited on 09/11/2026).

[131] Tianqi Chen et al. *Training Deep Nets with Sublinear Memory Cost*. 2016. URL: https://arxiv.org/abs/1604.06174.

[132] Colin Raffel et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67. URL: https://www.jmlr.org/papers/v21/20-074.html.

[133] Mohammad Bavarian et al. *Efficient Training of Language Models to Fill in the Middle*. 2022. URL: https://arxiv.org/abs/2207.14255.

[134] Ronald J. Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine Learning* 8 (1992), pp. 229–256. DOI: 10.1007/BF00992696. URL: https://link.springer.com/article/10.1007/BF00992696.

[135] John Schulman et al. *High-Dimensional Continuous Control Using Generalized Advantage Estimation*. 2015. URL: https://arxiv.org/abs/1506.02438.

[136] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. URL: https://arxiv.org/abs/1707.06347.

[137] Ralph Allan Bradley and Milton E. Terry. "Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons". In: *Biometrika* 39.3-4 (1952), pp. 324–345. DOI: 10.1093/biomet/39.3-4.324. URL: https://academic.oup.com/biomet/article-abstract/39/3-4/324/326091.

[138] Rafael Rafailov et al. *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*. May 29, 2023. arXiv: 2305.18290. URL: https://arxiv.org/abs/2305.18290.

[139] Zhihong Shao et al. *DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models*. Feb. 5, 2024. arXiv: 2402.03300. URL: https://arxiv.org/abs/2402.03300.

[140] Hunter Lightman et al. *Let's Verify Step by Step*. May 31, 2023. arXiv: 2305.20050. URL: https://arxiv.org/abs/2305.20050 (visited on 09/08/2026).

[141] Zichen Liu et al. *Understanding R1-Zero-Like Training: A Critical Perspective*. Mar. 26, 2025. arXiv: 2503.20783. URL: https://arxiv.org/abs/2503.20783 (visited on 09/08/2026).

[142] OpenAI. *Model guidance*. OpenAI API Documentation. 2026. URL: https://developers.openai.com/api/docs/guides/latest-model (visited on 09/11/2026).

[143] Tri Dao et al. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*. May 27, 2022. arXiv: 2205.14135. URL: https://arxiv.org/abs/2205.14135.

[144] Samyam Rajbhandari et al. *ZeRO: Memory Optimizations Toward Training Trillion Parameter Models*. Oct. 4, 2019. arXiv: 1910.02054. URL: https://arxiv.org/abs/1910.02054.

[145] NVIDIA. *CUDA Programming Guide: Writing SIMT Kernels*. URL: https://docs.nvidia.com/cuda/cuda-programming-guide/02-basics/writing-cuda-kernels.html (visited on 09/08/2026).

[146] Samuel Williams, Andrew Waterman, and David Patterson. "Roofline: An Insightful Visual Performance Model for Multicore Architectures". In: *Communications of the ACM* 52.4 (2009), pp. 65–76. DOI: 10.1145/1498765.1498785. URL: https://amcr.lbl.gov/departments/computer-science-department/ppan/roofline-performance-model/ppan-roofline-publications/ (visited on 09/08/2026).

[147] The Linux Kernel Documentation. *NUMA Memory Policy*. URL: https://kernel.org/doc/html/v6.7/admin-guide/mm/numa_memory_policy.html (visited on 09/08/2026).

[148] NVIDIA. *GPUDirect RDMA: Overview*. URL: https://docs.nvidia.com/cuda/gpudirect-rdma/index.html (visited on 09/08/2026).

[149] NVIDIA. *RDMA over Converged Ethernet: RoCE*. URL: https://docs.nvidia.com/networking-ethernet-software/cumulus-linux/Layer-1-and-Switch-Ports/Quality-of-Service/RDMA-over-Converged-Ethernet-RoCE/ (visited on 09/08/2026).

[150] NVIDIA. *NCCL User Guide: Collective Operations*. URL: https://docs.nvidia.com/deeplearning/nccl/user-guide/docs/usage/collectives.html (visited on 09/08/2026).

[151] NVM Express. *Specifications*. URL: https://nvmexpress.org/specifications/ (visited on 09/08/2026).

[152] Amazon Web Services. *What Is Amazon S3? Amazon S3 Data Consistency Model*. URL: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html (visited on 09/08/2026).

[153] Ascend. *Ascend Extension for PyTorch: v2.5.1-7.1.0 compatibility matrix*. 2025. URL: https://github.com/Ascend/pytorch/tree/v2.5.1-7.1.0 (visited on 09/08/2026).

[154] Volcano Authors. *Gang*. URL: https://volcano.sh/docs/scheduler/plugins/gang/ (visited on 09/08/2026).

[155] U.S. Department of Energy. *Best Practices Guide for Energy-Efficient Data Center Design*. 2024. URL: https://www.energy.gov/sites/default/files/2024-07/best-practice-guide-data-center-design.pdf (visited on 09/08/2026).

[156] Yanli Zhao, Andrew Gu, Rohan Varma, et al. *PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel*. 2023. arXiv: 2304.11277. URL: https://arxiv.org/abs/2304.11277.

[157] Mohammad Shoeybi et al. *Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism*. 2020. URL: https://arxiv.org/abs/1909.08053v4.

[158] Yanping Huang et al. *GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism*. 2019. URL: https://arxiv.org/abs/1811.06965v5.

[159] Vijay Korthikanti et al. *Reducing Activation Recomputation in Large Transformer Models*. 2022. arXiv: 2205.05198. URL: https://arxiv.org/abs/2205.05198.

[160] NVIDIA. *Megatron Core: Context Parallel Package*. URL: https://docs.nvidia.com/megatron-core/developer-guide/latest/user-guide/features/context_parallel.html (visited on 09/08/2026).

[161] PyTorch contributors. *PyTorch: FSDP2 fully_shard Implementation*. Version 2.11.0. URL: https://github.com/pytorch/pytorch/blob/v2.11.0/torch/distributed/fsdp/_fully_shard/_fully_shard.py (visited on 09/11/2026).

[162] DeepSpeed contributors. *DeepSpeed*. URL: https://github.com/deepspeedai/DeepSpeed (visited on 09/11/2026).

[163] NVIDIA. *Megatron-LM*. URL: https://github.com/NVIDIA/Megatron-LM (visited on 09/11/2026).

[164] Hugging Face. *Accelerate: DeepSpeed Integration*. Version 1.15.0. URL: https://github.com/huggingface/accelerate/blob/v1.15.0/docs/source/usage_guides/deepspeed.md (visited on 09/11/2026).

[165] Noam Shazeer. *Fast Transformer Decoding: One Write-Head is All You Need*. 2019. URL: https://arxiv.org/abs/1911.02150.

[166] DeepSeek-AI. *DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model*. 2024. URL: https://arxiv.org/abs/2405.04434v5 (visited on 09/08/2026).

[167] Woosuk Kwon et al. *Efficient Memory Management for Large Language Model Serving with PagedAttention*. 2023. URL: https://arxiv.org/abs/2309.06180.

[168] Gyeong-In Yu et al. “Orca: A Distributed Serving System for Transformer-Based Generative Models”. In: *16th USENIX Symposium on Operating Systems Design and Implementation*. USENIX Association, 2022, pp. 521–538. URL: https://www.usenix.org/conference/osdi22/presentation/yu.

[169] Amey Agrawal et al. *SARATHI: Efficient LLM Inference by Piggybacking Decodes with Chunked Prefills*. 2023. URL: https://arxiv.org/abs/2308.16369.

[170] John D. C. Little. “A Proof for the Queuing Formula: $L = \lambda W$”. In: *Operations Research* 9.3 (1961), pp. 383–387. DOI: 10.1287/opre.9.3.383. URL: https://pubsonline.informs.org/doi/abs/10.1287/opre.9.3.383.

[171] WHATWG. *HTML Standard: Server-sent events*. 2026. URL: https://html.spec.whatwg.org/multipage/server-sent-events.html (visited on 09/08/2026).

[172] Roy T. Fielding, Mark Nottingham, and Julian Reschke. *RFC 9110: HTTP Semantics*. 2022. URL: https://www.rfc-editor.org/rfc/rfc9110.html.

[173] Kubernetes Authors. *Liveness, Readiness, and Startup Probes*. 2026. URL: https://kubernetes.io/docs/concepts/workloads/pods/probes/ (visited on 09/08/2026).

[174] Yinmin Zhong et al. *DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving*. 2024. URL: https://arxiv.org/abs/2401.09670v3.

[175] Aosong Feng et al. *IPR: Intelligent Prompt Routing with User-Controlled Quality-Cost Trade-offs*. Version 4. Oct. 9, 2025. arXiv: 2509.06274. URL: https://arxiv.org/abs/2509.06274v4 (visited on 09/11/2026).

[176] OpenTelemetry Authors. *OpenTelemetry: Signals*. 2026. URL: https://opentelemetry.io/docs/concepts/signals/ (visited on 09/08/2026).

[177] vLLM contributors. *vLLM: Quickstart*. Version 0.29.0. URL: https://docs.vllm.ai/en/v0.29.0/getting_started/quickstart/ (visited on 09/11/2026).

[178] SGLang contributors. *SGLang: Inference Framework*. Version 0.5.19. URL: https://github.com/sgl-project/sglang/tree/v0.5.19 (visited on 09/11/2026).

[179] NVIDIA. *TensorRT-LLM*. URL: https://github.com/NVIDIA/TensorRT-LLM (visited on 09/11/2026).

[180] vLLM contributors. *vLLM: Architecture Overview*. v0.29.0. URL: https://docs.vllm.ai/en/v0.29.0/design/arch_overview/ (visited on 09/11/2026).

[181] Maxim Milakov and Natalia Gimelshein. *Online normalizer calculation for softmax*. 2018. URL: https://arxiv.org/abs/1805.02867.

[182] Yaniv Leviathan, Matan Kalman, and Yossi Matias. *Fast Inference from Transformers via Speculative Decoding*. Nov. 30, 2022. arXiv: 2211.17192. URL: https://arxiv.org/abs/2211.17192.

[183] Benoit Jacob et al. "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2704–2713. URL: https://openaccess.thecvf.com/content_cvpr_2018/papers/Jacob_Quantization_and_Training_CVPR_2018_paper.pdf.

[184] Elias Frantar et al. *GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers*. Oct. 31, 2022. arXiv: 2210.17323. URL: https://arxiv.org/abs/2210.17323.

[185] Ji Lin et al. *AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration*. June 1, 2023. arXiv: 2306.00978. URL: https://arxiv.org/abs/2306.00978.

[186] Guangxuan Xiao et al. *SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models*. Nov. 18, 2022. arXiv: 2211.10438. URL: https://arxiv.org/abs/2211.10438.

[187] Paulius Micikevicius et al. *FP8 Formats for Deep Learning*. 2022. URL: https://arxiv.org/abs/2209.05433.

[188] Yann LeCun, John S. Denker, and Sara A. Solla. "Optimal Brain Damage". In: *Advances in Neural Information Processing Systems*. Vol. 2. 1989. URL: https://proceedings.neurips.cc/paper/1989/hash/6c9882bbac1c7093bd25041881277658-Abstract.html (visited on 09/08/2026).

[189] Babak Hassibi and David G. Stork. "Second Order Derivatives for Network Pruning: Optimal Brain Surgeon". In: *Advances in Neural Information Processing Systems*. Vol. 5. 1992. URL: https://proceedings.neurips.cc/paper/1992/hash/303ed4c69846ab36c2904d3ba8573050-Abstract.html (visited on 09/08/2026).

[190] Elias Frantar and Dan Alistarh. *SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot*. Jan. 2, 2023. arXiv: 2301.00774. URL: https://arxiv.org/abs/2301.00774 (visited on 09/08/2026).

[191] NVIDIA. *cuSPARSELt: A High-Performance CUDA Library for Sparse Matrix-Matrix Multiplication*. URL: https://docs.nvidia.com/cuda/cusparselt/index.html (visited on 09/08/2026).

[192] Carl Eckart and Gale Young. "The Approximation of One Matrix by Another of Lower Rank". In: *Psychometrika* 1.3 (1936), pp. 211–218. DOI: 10.1007/BF02288367. URL: https://link.springer.com/article/10.1007/BF02288367 (visited on 09/08/2026).

[193] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. Mar. 9, 2015. arXiv: 1503.02531. URL: https://arxiv.org/abs/1503.02531 (visited on 09/08/2026).

[194] Yoon Kim and Alexander M. Rush. *Sequence-Level Knowledge Distillation*. June 25, 2016. arXiv: 1606.07947. URL: https://arxiv.org/abs/1606.07947 (visited on 09/08/2026).

[195] Noam Shazeer et al. *Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer*. Jan. 23, 2017. arXiv: 1701.06538. URL: https://arxiv.org/abs/1701.06538.

[196] William Fedus, Barret Zoph, and Noam Shazeer. "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity". In: *Journal of Machine Learning Research* 23.120 (2022), pp. 1–39. URL: https://jmlr.org/papers/v23/21-0998.html.

[197] Damai Dai, Chengqi Deng, Chenggang Zhao, et al. *DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models*. Jan. 11, 2024. arXiv: 2401.06066. URL: https://arxiv.org/abs/2401.06066.

[198] Joan Puigcerver et al. *From Sparse to Soft Mixtures of Experts*. ICLR 2024; arXiv 初稿发表于 2023 年. 2024. arXiv: 2308.00951. URL: https://arxiv.org/abs/2308.00951.

[199] Shouyuan Chen et al. *Extending Context Window of Large Language Models via Positional Interpolation*. 2023. URL: https://arxiv.org/abs/2306.15595.

[200] Bowen Peng et al. *YaRN: Efficient Context Window Extension of Large Language Models*. Version 2; 本章借鉴分频调整与注意力尺度思想，不引用跨版本性能数值. 2023. URL: https://arxiv.org/abs/2309.00071v2.

[201] Iz Beltagy, Matthew E. Peters, and Arman Cohan. *Longformer: The Long-Document Transformer*. 2020. URL: https://arxiv.org/abs/2004.05150.

[202] Nelson F. Liu et al. “Lost in the Middle: How Language Models Use Long Contexts”. In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 157–173. DOI: 10.1162/tacl_a_00638. URL: https://aclanthology.org/2024.tacl-1.9/.

[203] Cheng-Ping Hsieh et al. *RULER: What's the Real Context Size of Your Long-Context Language Models?* 2024. URL: https://arxiv.org/abs/2404.06654.

[204] Team Olmo. *Olmo 3*. 2025. URL: https://arxiv.org/abs/2512.13961v2 (visited on 09/12/2026).

[205] OpenAI. *gpt-oss-120b & gpt-oss-20b Model Card*. 2025. URL: https://arxiv.org/abs/2508.10925v1 (visited on 09/12/2026).

[206] An Yang et al. *Qwen3 Technical Report*. 2025. URL: https://arxiv.org/abs/2505.09388v1 (visited on 09/08/2026).

[207] Gemma Team. *Gemma 4 Technical Report*. 2026. URL: https://arxiv.org/abs/2607.02770v2 (visited on 09/12/2026).

[208] Meta. *Llama 4 Model Card*. Model Card. 覆盖 Scout 与 Maverick；官方在线模型卡，访问日期 2026-09-09。未取得可核读的固定 commit，因此链接为动态页面，不作为固定版本 API 行为证据。Meta. URL: https://github.com/meta-llama/llama-models/blob/main/models/llama4/MODEL_CARD.md (visited on 09/09/2026).

[209] DeepSeek-AI. *DeepSeek-V3 Technical Report*. 2024. URL: https://arxiv.org/abs/2412.19437v2 (visited on 09/08/2026).

[210] DeepSeek-AI. *DeepSeek-V4.1-Flash: Pushing the Limits of KV Cache Compression*. 官方仓库技术报告；日期按发布日记录；参数和资源结果为发布者报告。DeepSeek-AI, Sept. 10, 2026. URL: https://huggingface.co/deepseek-ai/DeepSeek-V4.1-Flash/blob/df42c109f1defefcbfcedbe7d905718a12266e40/DeepSeek_V41_Tech_Report.pdf (visited on 09/10/2026).

[211] Xin Cheng et al. *Conditional Memory via Scalable Lookup: A New Axis of Sparsity for Large Language Models*. 首次提交 2026-01-12；此处引用 v2。July 12, 2026. DOI: 10.48550/arXiv.2601.07372. arXiv: 2601.07372v2. URL: https://arxiv.org/abs/2601.07372v2 (visited on 09/10/2026).

[212] Shuai Bai et al. *Qwen2.5-VL Technical Report*. 2025. URL: https://arxiv.org/abs/2502.13923v1 (visited on 09/08/2026).

[213] Albert Q. Jiang et al. *Mistral 7B*. 2023. URL: https://arxiv.org/abs/2310.06825v1 (visited on 09/12/2026).

[214] Albert Q. Jiang et al. *Mixtral of Experts*. 2024. URL: https://arxiv.org/abs/2401.04088v1 (visited on 09/12/2026).

[215] Zhengxiao Du et al. *GLM: General Language Model Pretraining with Autoregressive Blank Infilling*. 2021. URL: https://arxiv.org/abs/2103.10360v2 (visited on 09/08/2026).

[216] GLM-4.5 Team. *GLM-4.5: Agentic, Reasoning, and Coding (ARC) Foundation Models*. 2025. URL: https://arxiv.org/abs/2508.06471v1 (visited on 09/08/2026).

[217] Kimi Team. *Kimi K2: Open Agentic Intelligence*. 2025. URL: https://arxiv.org/abs/2507.20534v2 (visited on 09/08/2026).

[218] MiniMax. *MiniMax-01: Scaling Foundation Models with Lightning Attention*. 2025. URL: https://arxiv.org/abs/2501.08313v1 (visited on 09/08/2026).

[219] Marah Abdin et al. *Phi-4 Technical Report*. 2024. URL: https://arxiv.org/abs/2412.08905v1 (visited on 09/12/2026).

[220] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[221] Hugo Touvron et al. "Training data-efficient image transformers & distillation through attention". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Proceedings of Machine Learning Research. 2021, pp. 10347–10357. URL: https://proceedings.mlr.press/v139/touvron21a.html.

[222] Hongyi Zhang et al. "mixup: Beyond Empirical Risk Minimization". In: *International Conference on Learning Representations*. 2018. arXiv: 1710.09412. URL: https://arxiv.org/abs/1710.09412.

[223] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020. URL: https://arxiv.org/abs/2103.00020.

[224] Junnan Li et al. *BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models*. 2023. arXiv: 2301.12597. URL: https://arxiv.org/abs/2301.12597.

[225] Jean-Baptiste Alayrac et al. *Flamingo: a Visual Language Model for Few-Shot Learning*. 2022. arXiv: 2204.14198. URL: https://arxiv.org/abs/2204.14198.

[226] Alec Radford et al. *Robust Speech Recognition via Large-Scale Weak Supervision*. 2022. arXiv: 2212.04356. URL: https://arxiv.org/abs/2212.04356.

[227] Jonathan Ho, Ajay Jain, and Pieter Abbeel. *Denoising Diffusion Probabilistic Models*. June 19, 2020. arXiv: 2006.11239. URL: https://arxiv.org/abs/2006.11239.

[228] Tim Salimans and Jonathan Ho. *Progressive Distillation for Fast Sampling of Diffusion Models*. 2022. arXiv: 2202.00512. URL: https://arxiv.org/abs/2202.00512.

[229] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising Diffusion Implicit Models". In: *International Conference on Learning Representations*. 2021. URL: https://arxiv.org/abs/2010.02502.

[230] William Peebles and Saining Xie. *Scalable Diffusion Models with Transformers*. 2023. URL: https://arxiv.org/abs/2212.09748.

[231] Jonathan Ho and Tim Salimans. *Classifier-Free Diffusion Guidance*. 2022. URL: https://arxiv.org/abs/2207.12598.

[232] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. URL: https://arxiv.org/abs/2112.10752.

[233] Yaron Lipman et al. *Flow Matching for Generative Modeling*. Oct. 6, 2022. arXiv: 2210.02747. URL: https://arxiv.org/abs/2210.02747.

LLM from Scratch

Zero to Hero

从文本表示到注意力计算，从训练目标到推理执行，本书以中文串联大语言模型的核心原理与工程实践.

正文提供概念、推导与系统关系，例题与系统案例说明计算过程及工程取舍. 学习围绕可解释的机制展开，结论由可检查的证据支撑.

基础与表示　模型与学习　评估与运行

应用与模型扩展　规模与效率

持续修订版