

Transformer架构与训练笔记

第一版 · 2026-10-17（北京时间）

从“这一行到底在算什么”到“一次训练怎样改变整个模型”。依据你的 16 次原始提问重组，保留问题、定义、推导、可手算例子与适用边界。

阅读方式。 第 1—4 章建立完整流程和对象；第 5—8 章拆开注意力与完整架构；第 9—11 章严格推导训练，并把一次更新算到底；第 12—13 章解释生成与能力边界；第 14 章回查全部提问。第一次读不必先掌握全部偏导，可先读第 1—8 章和第 12 章，再回到训练推导。

论述约定。 “定义”规定对象；“推导”在列明条件后逐步论证；“直觉”只帮助理解；“示例”中的数值由本文构造，不来自真实语言模型。关于大型模型能力的说明属于经验与机制解释，不作为数学定理。

正文目录

第 1 章：先看完整任务，再看内部零件	3
第 2 章：一句话怎样成为矩阵	4
第 3 章：参数、向量、权重与维度	6
第 4 章：训练与生成共用哪一段计算	8
第 5 章：Q、K、V 为什么分开	10
第 6 章：从点积分数到凸组合	11
第 7 章：缩放与因果掩码的严格依据	13
第 8 章：从单头注意力到完整 Transformer	17
第 9 章：输出概率与统一训练目标	20
第 10 章：梯度怎样穿过整个计算图	22
第 11 章：一个小模型的完整前向与更新	26
第 12 章：模型如何逐个输出文字	31
第 13 章：简单运算为何能有复杂用途	33
第 14 章：原始提问回查、练习与来源	34

第 1 章：先看完整任务，再看内部零件

[返回目录](#)

1.1 你真正想接起来的是什么

原对话中，你依次问“为什么乘 V ”“ x 输出的 z 有什么意义”“输入的是一句话吧”“现在是在训练还是输出”“还得有一个预测函数吧”。这些问题共同要求一条完整的数据流。

本文先研究用于下一个 Token 预测的因果 Transformer 语言模型，即 GPT 类模型的基础计算形式。随后区分原始编码器—解码器 Transformer。不要把注意力公式、某一种架构和完整聊天产品当成同一个对象。

设词表是有限集合，大小为 N 。文本被编码成编号序列 $t_1, \dots, t_n$ ，其中 $t_i \in \{1, \dots, N\}$ 。模型要计算的是：

$$p_{\theta}(w \mid t_1, \dots, t_n), \quad w \in \{1, \dots, N\}.$$

它给所有候选下一个 Token 各分配一个非负概率，总和为 1。这里 θ 是模型全部可训练参数的集合。“输出一个词”之前，先发生“输出整个词表上的一组分数与概率”。

1.2 一条主线，两个用途

共同的前向计算： 文本 $\rightarrow$ Token 编号 $\rightarrow$ 嵌入向量与位置信息 $\rightarrow$ 多层 Transformer $\rightarrow$ 最后隐藏表示 $\rightarrow$ 词表分数 $\rightarrow$ 概率分布。

训练接着做： 读取数据提供的正确 Token $\rightarrow$ 计算损失 $\rightarrow$ 反向传播求梯度 $\rightarrow$ 优化器更新参数。

生成接着做： 按解码规则选一个 Token $\rightarrow$ 追加到前缀 $\rightarrow$ 再预测下一个 $\rightarrow$ 到停止条件时结束。

“训练”不是某一种特殊的 Attention 公式。Attention 是前向计算中的模块，训练和生成都会使用它。反向传播计算导数，优化器才负责实际改参数，两者也要分清。

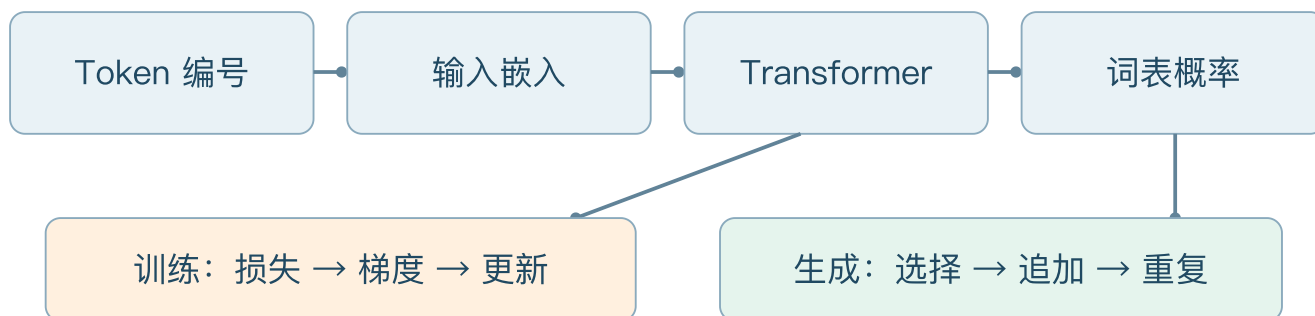


图 1: 共同的前向计算, 在概率之后分成训练与生成两条路径。

1.3 本文固定的语言示例

始终采用这句话：**患者服用华法林后出现黑便。** 为教学指定如下玩具分词：

$$(t_1, \dots, t_7) = (\text{患者}, \text{服用}, \text{华法林}, \text{后}, \text{出现}, \text{黑便}, \text{。})$$

真实 tokenizer 可能把它切成完全不同的片段。“黑便”是否恰好一个 Token，不由语义词语边界决定。本文没有调用真实 tokenizer；所有涉及这个切分的计算均属于教学示例。该句只用于学习模型结构，不推导临床判断。

要预测“黑便”，使用的是到“出现”为止的前缀，以及第 5 个位置的最终表示。若输入已经包含“黑便。”，接下来预测的就是句号之后的内容。

第 2 章：一句话怎样成为矩阵

[返回目录](#)

2.1 Token 编号不是语义坐标

Tokenizer 把文本变成离散编号。假设“患者”编号为 17，“服用”编号为 92，这两个数之差 75 不表示语义距离。编号用于索引，不直接拿来做语言意义上的加减。

输入嵌入矩阵定义为：

$$E \in \mathbb{R}^{N \times d}.$$

第 w 行是词表中第 w 个 Token 的可训练向量。第 i 个位置先查表得到行向量：

$$\mathbf{e}_i = \mathbf{E}_{t_i,:} \in \mathbb{R}^{1 \times d}.$$

若用 one-hot 行向量 $\mathbf{u}_i \in \mathbb{R}^{1 \times N}$ 表示编号，则 $\mathbf{e}_i = \mathbf{u}_i \mathbf{E}$ 。查表和这个乘法数学上等价；实现无需真的构造巨大 one-hot。

2.2 从一行到整句

采用最简单的加法位置编码时：

$$\mathbf{x}_i = \mathbf{e}_i + \mathbf{p}_i, \quad \mathbf{X}^{(0)} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix} \in \mathbb{R}^{n \times d}.$$

第 i 行对应一个位置，第 r 列对应表示空间的一维。模型接收的是整个序列矩阵；单独研究一行，只是把整体运算放大来看。

同一个 Token 在不同位置查出的 $\mathbf{e}_i$ 相同，但加上不同的 $\mathbf{p}_i$ 后， $\mathbf{x}_i$ 可以不同。采用 RoPE 时，位置作用的方式又不同，不能统一说所有模型都把位置向量加到嵌入上。

2.3 输入向量也需要训练吗

需要。在通常的全参数预训练中， $\mathbf{E}$ 是可训练参数。编号与分词规则在这一训练阶段通常固定，梯度更新的是被查到的向量数值，不是把离散编号微分。

若损失对某位置向量的梯度为 $\mathbf{g}_i$ ，查表反向传播把 $\mathbf{g}_i$ 加到 $\mathbf{E}$ 对应行。一个 Token 在输入出现多次时，贡献相加。这一事实将在第 10 章给出公式。

不要预设“第 1 维表示药物、第 2 维表示症状”。模型学习的是能被后续运算利用的分布式表示，坐标并没有这样的人工语义标签。

第 3 章：参数、向量、权重与维度

[返回目录](#)

3.1 四种容易混淆的对象

对象	身份	是否直接被优化器更新
W_Q, W_K, W_V	可训练投影矩阵	全参数训练时是
Q, K, V	当前输入算出的中间结果	否；下次前向重新算
S	位置之间的匹配分数	否
A	当前输入的注意力混合系数	否
Z, H	中间或最终隐藏表示	否
E, U	输入嵌入、词表输出矩阵	全参数训练时是

中文“权重”有歧义：参数矩阵的权重与注意力权重 A 不是一回事。我们训练参数，参数改变后，下次前向得到的 A 才可能改变。对中间量求梯度也不等于把它当作独立参数更新。

3.2 维度表与行向量约定

全文向量按行排列；用 $x_i W$ ，不在同一条推导中切换为 $W x_i$ 。

符号	含义	形状
n	当前序列的位置数	正整数
N	词表大小	正整数
d	模型隐藏表示维度	正整数

符号	含义	形状
d_k, d_v	单头 Q/K 维度、V 维度	正整数
X	某一层输入	$n \times d$
W_Q, W_K	Q/K 投影参数	$d \times d_k$
W_V	V 投影参数	$d \times d_v$
Q, K	投影结果	$n \times d_k$
V	内容投影结果	$n \times d_v$
S, A	分数、混合系数	$n \times n$
Z	单头输出	$n \times d_v$
U	词表输出参数	$d \times N$

用 N 表词表大小，以免与 Value 矩阵 V 重名。 d_k 是一次点积有多少对坐标相乘，与句子有多少 Token 无关。 $n = 20, d_k = 64$ 时， QK^T 是 20×20 ，缩放因子是 8。

3.3 “这一步是在算 WQ 还是 Q”

严格拆开四步：

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V.$$

$$S = QK^T / \sqrt{d_k} + M.$$

$$A = \text{softmax}_{\text{row}}(S), \quad Z = AV.$$

第 1 步算 Q/K/V；第 2 步用它们算分数；第 3 步算混合系数；第 4 步算内容的加权结果。这里没有任何一步通过点积直接“解出最优 W_Q ”。训练在算完预测与损失以后，再计算参数梯度。

第 4 章：训练与生成共用哪一段计算

[返回目录](#)

4.1 目标错开一位

对序列 $t_1, \dots, t_7$ ，第 i 个位置的最终向量 h_i 用来预测 t_{i+1} 。这不是让它复述当前的 t_i 。

输入位置	允许使用的前缀	训练目标
1: 患者	患者	服用
2: 服用	患者、服用	华法林
3: 华法林	到华法林	后
4: 后	到后	出现
5: 出现	到出现	黑便
6: 黑便	到黑便	。

最简单的训练数组可以取 inputs 为 $t_1, \dots, t_6$ ，labels 为 $t_2, \dots, t_7$ 。也可以把七个 Token 都送入，再忽略没有已知下一目标的最后一行。两者不要混写。

句号不是天然的结束 Token。若训练数据显式追加 EOS，才会有“句号位置预测 EOS”的目标。若添加 BOS，则可训练首 Token 的预测；没有 BOS 时，上表从第二个 Token 的条件概率开始。

4.2 为什么整句输入仍然不是偷看答案

训练系统知道所有 labels，但计算 h_5 的前向路径只允许依赖 $t_1, \dots, t_5$ 。 t_6 可以存在于别的位置，不能流进这个位置。因果约束必须作用于每一层，不能只在最终输出时删掉未来词。

这使一层内部各位置能同时计算，下一层仍需等待上一层完成。所谓“并行训练”指位置维度的并行，不是层与层之间没有依赖，也不是多个优化步骤同时发生。

4.3 Teacher forcing 与真正生成

训练时，即使第 5 行概率最大的是错误 Token，后面位置仍用训练文本中的“黑便”作为输入。标准下一个 Token 交叉熵训练不需要先采样一个词，再把错误词塞回训练序列。

生成时没有预先提供的正确后续，选出的 Token 才会进入下一次上下文。选错后的后续前缀也跟着改变。

阶段	概率来自哪里	概率之后做什么
训练	当前参数下的前向计算	取真实目标的概率，算损失与梯度
生成	当前参数下的前向计算	采样或取最大值，追加 Token

相同参数、相同前缀、相同位置与掩码、关闭 dropout 且相同数值条件时，两者在同一位置的预测概率应相同。示意中的 30% 与 60% 不能被解释为“训练公式与生成公式不同”。

第 5 章：Q、K、V 为什么分开

[返回目录](#)

5.1 先有任务：当前位置需要读取别处的信息

如果仅对每个位置分别计算 $f(x_i)$ ，那么在 x_i 不变时，无论前面是什么，结果都不变。为了让一个位置受其他位置影响，我们引入：

$$z_i = \sum_{j \leq i} a_{ij}(X) v_j.$$

“从哪里读取多少”由 a_{ij} 决定；“读取后得到什么向量”由 v_j 决定。前者是混合规则，后者是被混合的内容。

这也回答为什么还要乘 V： A 只有位置间的比例，形状是 $n \times n$ ； AV 才把这些比例变成每个位置的新特征，形状是 $n \times d_v$ 。

5.2 为什么不直接计算 x 与 x 的点积

直接点积是可以设计的，但会限制匹配方式。学习两套投影后：

$$q_i k_j^T = x_i W_Q W_K^T x_j^T.$$

记 $B = W_Q W_K^T$ ，匹配分数就变成可学习的双线性形式 $x_i B x_j^T$ 。 B 的秩至多为 d_k ，也不必对称，因此“位置 i 对 j 的需求”不必等于“j 对 i 的需求”。如果强迫 $W_Q = W_K$ ，则 $B = W_Q W_Q^T$ 是对称半正定的，匹配表达受到额外约束。

手算例子。取 $x_a = (1, 0)$, $x_b = (0, 1)$ ，以及：

$$W_Q = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad W_K = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

则 $q_a = 1, k_b = 1$ ，有 $q_a k_b = 1$ ；但 $q_b = 0, k_a = 0$ ，有 $q_b k_a = 0$ 。原始 $x_a \cdot x_b = 0$ 并不妨碍投影后的特定方向匹配为 1。

这说明学到的匹配不是“原始词向量相似度”的同义词。“语义相关就天然高分”也不是定理；训练目标会影响怎样的匹配有用。

5.3 为什么 K 和 V 不必一样

假设两个候选位置的权重已固定为 $(0.8, 0.2)$ ，内容为 $v_1 = (1, 0), v_2 = (0, 1)$ ，得到 $z = (0.8, 0.2)$ 。

保持 Q/K 不变，只把 W_V 改为原来的两倍，则 A 不变， z 变成 $(1.6, 0.4)$ 。反过来，只改变 Q/K 可能改变比例，却不直接改变候选内容向量。

所以独立的 W_K, W_V 允许模型分别学习“如何被匹配”和“传递什么”。使用同一个矩阵也不是数学错误，只是不同的模型约束。

“Q 提问，K 索引，V 内容”是检索类比。矩阵中没有真正的文字问题或数据库条目；每一维仍是数值特征，三个角色由计算位置定义。

第 6 章：从点积分数到凸组合

[返回目录](#)

6.1 一次点积究竟做了什么

对一行 $q_i = (q_{i1}, \dots, q_{id_k})$ 和一行 k_j ：

$$q_i k_j^\top = \sum_{r=1}^{d_k} q_{ir} k_{jr}.$$

它是一个实数分数，可以为负。只有当两个向量都非零时，才可写成模长乘积乘夹角余弦；不要求单位向量。分数为负不表示“不存在关联”，而表示在这一套参数与坐标表示下得到较低的相对分数。

6.2 Softmax 的定义与三项性质

对有限实数 $s_1, \dots, s_m$ 定义：

$$a_j = \frac{e^{s_j}}{\sum_{r=1}^m e^{s_r}}.$$

因为每项指数严格正，分母正，故 $a_j > 0$ ；直接相加得到 $\sum_j a_j = 1$ 。并且：

$$\frac{a_j}{a_k} = e^{s_j - s_k}.$$

因此分数更大者权重更大，相差 1 时权重比为 e 。所有分数同时加常数 c ，分子分母同乘 e^c ，概率不变。这也允许数值实现先减去最大分数，避免指数溢出。

Softmax 是一种平滑的相对分配规则。上述性质不能唯一推出“必须选指数”，其他注意力变体可以使用不同规则。

6.3 一个完整手算

取分数 $(\ln 2, 0, 0)$ ，则指数是 $(2, 1, 1)$ ，权重是 $(1/2, 1/4, 1/4)$ 。若：

$$v_1 = (0, 0), \quad v_2 = (4, 0), \quad v_3 = (0, 4),$$

则：

$$z = \frac{1}{2}v_1 + \frac{1}{4}v_2 + \frac{1}{4}v_3 = (1, 1).$$

依据凸包的定义，有限组点的一切非负、和为 1 的线性组合组成其凸包，因此这个 z 属于候选 Value 的凸包。这是严格结论。

但候选集合和系数都依赖输入与参数；凸包本身会改变。完整模型的多头输出投影、残差和 FFN 也不受同一个凸包约束。训练时若在注意力系数上使用未重新归一化的 dropout，其具体随机输出也不再直接适用这里的和为 1 结论。

6.4 矩阵乘法为什么恰好完成全部加权

A 的第 i 行是当前位置分配给各候选的比例， V 的第 j 行是候选内容。由矩阵乘法定义：

$$(AV)_{ir} = \sum_{j=1}^n a_{ij} V_{jr}.$$

固定 i ，让 r 遍历所有内容坐标，就得到 $z_i = \sum_j a_{ij} v_j$ 。不是“矩阵写法大概等价”，而是逐元素严格相同。

6.5 z 有意义吗

z_i 不是生成的文字，也不承诺某一坐标具有人类可读含义。它的功能是为后续计算提供依赖上下文的特征。在“我吃了苹果”和“我投资了苹果”中，“苹果”位置能读取的前文不同，因此其表示可以不同。

这里刻意把提示词放在“苹果”之前。对因果模型，不能用“苹果很甜”中后面的“甜”解释此前“苹果”位置已经获得的表示。

随机参数也能算出 z ，但算出来并不保证有用。功能上的用途由整个网络怎样利用它降低训练目标来建立，不能把“加权平均”本身当作理解语言的证明。

第 7 章：缩放与因果掩码的严格依据

[返回目录](#)

7.1 为什么会出现平方根

先讨论随机变量，再回到固定输入向量。假设 q_r, k_r 全部相互独立，各有均值 0、方差 1，定义 $Y_r = q_r k_r$ 。这是说明尺度的理想化假设，不是训练后 Q/K 的已证事实。

独立性与有限二阶矩给出：

$$\mathbb{E}Y_r = \mathbb{E}q_r \mathbb{E}k_r = 0,$$

$$\mathbb{E}Y_r^2 = \mathbb{E}q_r^2 \mathbb{E}k_r^2 = 1.$$

因此 $\text{Var}(Y_r) = 1$ 。不同 r 的乘积也独立，故：

$$\text{Var}\left(\sum_{r=1}^{d_k} Y_r\right) = d_k, \quad \text{SD}(q \cdot k) = \sqrt{d_k}.$$

又由 $\text{Var}(cZ) = c^2 \text{Var}(Z)$ ，除以 $\sqrt{d_k}$ 后方差是 1。除以 d_k 后方差却为 $1/d_k$ ，会随维度增加而缩小。

7.2 你追问过的方差等式

定义是：

$$\text{Var}(Z) = \mathbb{E}[(Z - \mathbb{E}Z)^2] = \mathbb{E}[Z^2] - (\mathbb{E}Z)^2.$$

当 $\mathbb{E}X = \mathbb{E}Y = 0$ 时，期望的线性性质给出 $\mathbb{E}(X + Y) = 0$ ，于是：

$$\text{Var}(X + Y) = \mathbb{E}[(X + Y)^2].$$

这一步不需要独立。继续展开：

$$\mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2].$$

要把交叉项消去，需要协方差为零；独立是充分条件。一般恒等式则是：

$$\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y) + 2 \text{Cov}(X, Y).$$

四个独立的等概率正负 1 相加，总和可能为 $-4, -2, 0, 2, 4$ ，概率分别为 $(1, 4, 6, 4, 1)/16$ 。均值是 0，二阶矩是 $(16 + 16 + 0 + 16 + 16)/16 = 4$ ，标准差是 2。最大可能值与典型波动尺度不同。

上述因子分解可从独立变量的乘积联合分布推得；本文以期望线性性、独立变量可积函数乘积的期望分解为概率基础定理，不把它们当作无条件代数变形。

7.3 为什么分数尺度影响学习

令 $a = \text{softmax}(s)$ 。直接对商求导可得：

$$\frac{\partial a_j}{\partial s_k} = a_j(\delta_{jk} - a_k).$$

δ_{jk} 是 Kronecker 符号，同下标为 1，否则为 0。当某项趋近 1、其余趋近 0，所有这些导数都趋近 0。因而过尖锐的注意力 Softmax 可能使经过它传回的梯度变小。缩放帮助控制进入 Softmax 的分数尺度。

这是机制解释，不保证实际模型方差恰好为 1，也不表示一切梯度问题都被解决。输出 Softmax 与交叉熵合并后的梯度还有简化式，不能直接用“Softmax 饱和”推出错误且自信的输出一定学不动。

7.4 掩码是一条依赖限制

为避免对无穷量求导的歧义，先直接定义允许集合：

$$J_i = \{1, \dots, i\}, \quad a_{ij} = \begin{cases} \frac{e^{q_i k_j^\top / \sqrt{d_k}}}{\sum_{r \in J_i} e^{q_i k_r^\top / \sqrt{d_k}}}, & j \in J_i, \\ 0, & j \notin J_i. \end{cases}$$

常见实现把不允许的位置分数加 $-\infty$ ，是上述定义的简写；未来位置权重为 0。每行至少需要一个允许位置，否则全被屏蔽的 Softmax 没有这个归一化意义。本文因允许关注自身，不会出现空集合。

没有因果掩码、没有其他屏蔽且分数有限时，各候选权重严格正，但可能小到几乎无贡献。浮点数实际计算还可能下溢为 0，不能把数学上的正数与机器表示混为一谈。

	患者	服用	华法林	后	出现	黑便	。
患者	可见	屏蔽	屏蔽	屏蔽	屏蔽	屏蔽	屏蔽
服用	可见	可见	屏蔽	屏蔽	屏蔽	屏蔽	屏蔽
华法林	可见	可见	可见	屏蔽	屏蔽	屏蔽	屏蔽
后	可见	可见	可见	可见	屏蔽	屏蔽	屏蔽
出现	可见	可见	可见	可见	可见	屏蔽	屏蔽
黑便	可见	可见	可见	可见	可见	可见	屏蔽
。	可见	可见	可见	可见	可见	可见	可见

图 2：每行是读取信息的位置。橙框“出现”行不能读取“黑便”；图中表示可见性，不是注意力大小。

7.5 多层之后会不会绕路偷看未来

归纳证明。 初始第 i 行只由 t_i 与位置决定，故依赖前缀 $t_{\leq i}$ 。假设第 ℓ 层每个位置 j 只依赖 $t_{\leq j}$ 。下一层位置 i 只读取 $j \leq i$ 的 K/V，所有这些输入都只依赖 $t_{\leq i}$ ；Q 也由本位置产生。逐位置 FFN、归一化和残差不会引入新位置。因此下一层仍只依赖前缀。归纳完成。

条件是所有跨位置操作遵守该限制。若某一步在位置之间做全局汇总，或错误地使用未来信息，则不能直接套用证明。常见 LayerNorm 沿每个位置的特征维归一化，满足此条件。

第 8 章：从单头注意力到完整 Transformer

[返回目录](#)

8.1 多头怎样接回模型维度

设有 H 个头，每头分别计算 $Z^{(r)} \in \mathbb{R}^{n \times d_v}$ 。拼接与投影是：

$$O = \text{Concat}(Z^{(1)}, \dots, Z^{(H)})W_O, \quad W_O \in \mathbb{R}^{Hd_v \times d}.$$

拼接后是 $n \times Hd_v$ ，乘 W_O 回到 $n \times d$ ，才能与原输入相加。多头允许多个混合分布同时存在，不要求“药物头”“语法头”这样人为固定的语义分工。

8.2 FFN 为什么逐位置却仍有用

一个两层前馈网络按行写为：

$$\text{FFN}(x) = \sigma(xW_1 + b_1)W_2 + b_2,$$

其中 $W_1 \in \mathbb{R}^{d \times m}$, $W_2 \in \mathbb{R}^{m \times d}$ 。对同层每个位置使用同一组参数。Attention 混合位置，FFN 则在当前位置变换特征；输入已经包含上下文，所以逐位置处理并不等于脱离上下文。

以 ReLU 为例， $\sigma(u) = \max(0, u)$ 逐坐标作用。若删除激活，两次仿射变换能合并成一次仿射变换；有激活则一般不能。但是 Attention 自身通常已经非线性，不能因此声称“没有 FFN，整个网络就是线性的”。

ReLU 在 0 不可微，实际自动微分采用约定值；使用平滑激活时按其导数计算。现代模型还可能用门控 FFN，本文不把这个简单形式当作所有模型的精确配置。

8.3 残差不是一句“保留信息”就完了

残差形式是 $y = x + F(x)$ 。相加要求形状相同。它提供一条不必经过 F 的直接路径，而不是保证原信息永远不丢失，例如 $F(x) = -x$ 会得到零。

在列向量的 Jacobian 记号下：

$$\mathbf{J}_y = \mathbf{I} + \mathbf{J}_F.$$

所以梯度有一条恒等路径。这有助于解释深层优化，但不保证 $\mathbf{I} + \mathbf{J}_F$ 永远良好，也不构成“绝不会梯度消失”的证明。

8.4 LayerNorm 实际算什么

对一个位置的行向量 $\mathbf{x} = (x_1, \dots, x_d)$ ，定义：

$$\mu = \frac{1}{d} \sum_{r=1}^d x_r, \quad v = \frac{1}{d} \sum_{r=1}^d (x_r - \mu)^2,$$

$$\text{LN}(\mathbf{x})_r = \gamma_r \frac{x_r - \mu}{\sqrt{v + \epsilon}} + \beta_r.$$

$\epsilon > 0$ 防止分母为零， γ, β 通常可训练。它沿同一位置的特征坐标运算，不用全句未来位置的均值。这里 v 是归一化定义中的总体平方偏差，不使用统计样本方差的 $d - 1$ 分母。

直觉是调整特征尺度，使后续模块面对的数值更稳定；加上 ϵ 和可训练缩放后，不应声称最终输出严格均值 0、方差 1。RMSNorm 是另一种定义，不能与这里的 LayerNorm 不加区分地互换。

8.5 一种明确的完整 Block

本文选 Pre-LN 作为完整架构示范，省略 dropout 与各投影偏置：

$$\mathbf{T}^{(\ell)} = \mathbf{X}^{(\ell)} + \text{MHA}_{\ell}(\text{LN}_{\ell,1}(\mathbf{X}^{(\ell)})),$$

$$\mathbf{X}^{(\ell+1)} = \mathbf{T}^{(\ell)} + \text{FFN}_{\ell}(\text{LN}_{\ell,2}(\mathbf{T}^{(\ell)})).$$

多层后可再用最终归一化得到 $H = \text{LN}_f(X^{(L)})$ 。每层参数通常独立；同一层参数在不同序列位置共享。原始 2017 年论文使用 Post-LN，即子层相加后归一化，顺序不同。不能把本文的 Pre-LN 公式标成原论文逐项原样结构。[来源 R1、R3]

8.6 位置信息为何需要单独交代

若没有位置编码也没有位置相关掩码，置换矩阵 P 作用于 X 后，Q/K/V 都左乘 P ，分数变成 PSP^T 。按行 Softmax 随着行列置换一起重排，因此输出变为 PZ 。这种性质称为置换等变：单纯交换输入位置会相应交换输出，结构本身不提供绝对顺序标签。

因果掩码已经引入方向与可见范围，所以不能把上述无掩码结论原封不动套给带因果掩码的网络。显式位置信息进一步提供距离或位置结构。

加法位置编码是前述 $e_i + p_i$ 。RoPE 则对 Q/K 的二维坐标块做与位置相关的旋转。用列向量说明其核心恒等式：令旋转后 $\tilde{q}_i = R_i q_i, \tilde{k}_j = R_j k_j$ ，则：

$$\tilde{q}_i^T \tilde{k}_j = q_i^T R_i^T R_j k_j.$$

同一二维块若 $R_i = R(i\omega)$ ，则 $R_i^T R_j = R((j - i)\omega)$ ，点积引入相对位置差。本段为了清楚写旋转恒等式显式采用列向量；换回前文行向量时相应转置。[来源 R4]

8.7 三类架构分别是什么

架构	主要信息流	常见任务形式
Encoder-only	输入位置双向互读	表示、分类、掩码预测等
Decoder-only	因果自注意力	自回归续写
Encoder-Decoder	编码器双向；解码器因果并读取编码结果	条件序列生成，如翻译

编码器—解码器的交叉注意力中，Q 来自解码端，K/V 来自编码端，长度可以不同，注意力矩阵是目标长度乘源长度。它不是把同一个 X 投影三次的 self-attention。原始 Transformer 属于第三类；本笔记训练与生成主线研究第二类。[来源 R1、R5]

第 9 章：输出概率与统一训练目标

[返回目录](#)

9.1 缺少的预测函数

最终隐藏矩阵 $H \in \mathbb{R}^{n \times d}$ 的每行 h_i 经过输出矩阵：

$$\ell_i = h_i U + b, \quad U \in \mathbb{R}^{d \times N}, \quad b \in \mathbb{R}^{1 \times N}.$$

ℓ_i 是 logits，不是概率。词表中第 w 个 Token 的分数为隐藏向量与 U 第 w 列的点积，加上可选偏置。

$$p_{i,w} = \frac{e^{\ell_{i,w}}}{\sum_{r=1}^N e^{\ell_{i,r}}}.$$

Attention 中的 Softmax 沿“可见输入位置”归一化；这里沿“整个候选词表”归一化。两个函数形式相同，索引集合和用途不同。输入中没出现过的候选 Token，也有自己的输出列与概率。

“向量变回文字”并不是对输入嵌入做数学求逆。它是依据隐藏特征计算候选分数，再进行离散选择。

9.2 为什么用负对数损失

若目标编号是 y ，用 one-hot 目标分布 $r_w = \mathbf{1}_{w=y}$ ，交叉熵为：

$$L_i = - \sum_w r_w \ln p_{i,w} = - \ln p_{i,y}.$$

正确目标概率提高，损失严格下降。但仅凭“概率低时罚得多”并不能唯一确定对数形式。

更系统的依据是条件最大似然。对一段已知文本，条件概率链式分解给出：

$$p_{\theta}(t_2, \dots, t_m \mid t_1) = \prod_{i=1}^{m-1} p_{\theta}(t_{i+1} \mid t_{\leq i}).$$

这来自概率链式法则，不要求 Token 独立。最大化这个乘积等价于最小化其负对数，即各位置负对数之和。多个训练样本则按给定训练采样与目标求和或平均。[来源 R2 提供神经网络语言模型背景]

定义有效目标集合 I ，平均损失写为：

$$L = \frac{1}{|I|} \sum_{i \in I} -\ln p_{i, t_{i+1}}.$$

padding 与不计分位置应排除。某些对话微调仅在助手回答位置计损失；这是损失掩码，作用不同于因果注意力掩码。后者控制信息流，前者控制哪些位置计入目标。

9.3 输入层、内部层、输出层怎样一起学

把全部参数收集为 $\theta = \{E, \text{各层参数}, U, b\}$ 。先用同一组当前参数算 L ，再一次反向传播获得各参数梯度，然后更新。通常的全参数预训练不需要分别先训练输入层、再训练输出层。

“联合训练”不意味着每个参数每一步梯度都非零。没有出现在输入中的嵌入行、被屏蔽的路径或恰好抵消的贡献，都可能梯度为零。冻结参数或只训练适配器的微调又是不同设置。

9.4 权重共享不等于逆变换

当 $E \in \mathbb{R}^{N \times d}$ ，可以设置 $U = E^T$ 。此时第 w 个 logit 是 $h_i E_{w,:}^T + b_w$ ，即与词表各嵌入的点积。它不是对 E 求逆，也不要求 E 是方阵。

同一个参数 E 被用于输入查表和输出打分，两条路径梯度相加。没有共享时，两组参数各自更新。第 11 章为了看清两种角色，刻意不共享。[权重共享设计见 R1 §3.4、R6]

第 10 章：梯度怎样穿过整个计算图

[返回目录](#)

10.1 梯度为什么给出下降方向

对可微标量损失 $L(\theta)$ ，把所有参数展开成向量。由一阶展开：

$$L(\theta - \eta g) = L(\theta) - \eta \|g\|^2 + o(\eta), \quad g = \nabla L(\theta).$$

若 $g \neq 0$ ，充分小的正 η 使损失下降。有限步长不自动满足条件；批次损失下降也不保证所有样本或测试集表现改善。

反向传播是多元链式法则在计算图上的高效应用，求的是当前参数处的局部导数。梯度下降是更新规则；Adam 类方法再使用梯度的历史统计量调整更新。本文数值例子只用最简单的梯度下降。

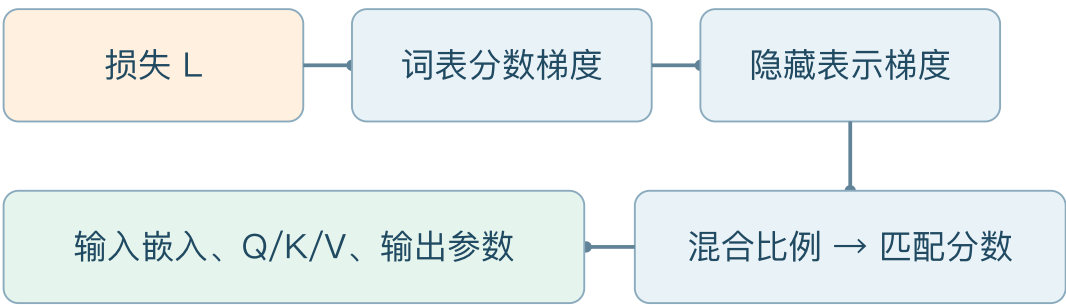


图 3：梯度沿依赖关系反向传递；分支贡献相加，最终共同更新参数。

10.2 记号与矩阵梯度的定义

用 $\bar{X} = \partial L / \partial X$ 表示与 X 同形状的梯度，满足：

$$dL = \sum_{ir} \bar{X}_{ir} dX_{ir}.$$

它不是把两个矩阵相除。若一个变量被多处分支使用，来自所有分支的梯度相加。以下推导先针对一个位置、一个损失；多位置平均只需累加并乘平均系数。

10.3 从损失回到 logits

对目标 y ：

$$L = -\ell_y + \ln \sum_r e^{\ell_r}.$$

逐坐标求导：

$$\bar{\ell}_j = p_j - \mathbf{1}_{j=y}.$$

错误候选的导数是正数，目标的导数是 $p_y - 1 < 0$ 。如果 logits 本身是独立参数，梯度下降会降低错误分数、提高目标分数。但在真实网络中 logits 共享上游参数，更新的效果需要考虑完整依赖，不能逐项照搬为语义结论。

10.4 从输出层回到隐藏向量

$\ell = hU + b$ 时，按乘积微分或逐元素求和得到：

$$\bar{U} = h^T \bar{\ell}, \quad \bar{h} = \bar{\ell} U^T, \quad \bar{b} = \bar{\ell}.$$

例如 $\ell_j = \sum_r h_r U_{rj} + b_j$ ，故 $\partial L / \partial U_{rj} = h_r \bar{\ell}_j$ ，这就是第一式的外积。多个位置时， $\bar{U} = H^T \bar{\Lambda}$ ， $\bar{b}$ 对位置求和。

10.5 穿过残差和内容混合

先考虑单头简化模块：

$$R = X + ZW_O, \quad Z = AV.$$

已知 $\bar{R}$, 则:

$$\bar{W}_O = Z^\top \bar{R}, \quad \bar{Z} = \bar{R} W_O^\top.$$

同时 X 从残差直连收到一份 $\bar{R}$ 。再由 $Z = AV$:

$$\bar{A} = \bar{Z} V^\top, \quad \bar{V} = A^\top \bar{Z}.$$

为什么 $\bar{A}_{ij}$ 会涉及 V? 因为改变 a_{ij} , 会沿 v_j 方向改变 z_i , 故 $\bar{A}_{ij} = \bar{z}_i \cdot v_j$ 。这说明匹配应不应该加强, 要看所读取的内容如何影响最终损失。

10.6 穿过注意力 Softmax

只对允许位置求导。由第 7 章 Jacobian:

$$\bar{S}_{ij} = a_{ij} \left(\bar{A}_{ij} - \sum_{r \in J_i} a_{ir} \bar{A}_{ir} \right), \quad j \in J_i.$$

禁止位置的 $\bar{S}_{ij} = 0$ 。每个允许位置的梯度由“它自己的影响”减去“当前混合影响的加权平均”决定, 再乘其权重。导数不是只看这一条边, 还受到同一行其他候选的竞争影响。

10.7 回到 Q、K、V 的参数

对掩码前的有限分数 $S_0 = QK^\top / \sqrt{d_k}$:

$$\bar{Q} = \bar{S} K / \sqrt{d_k}, \quad \bar{K} = \bar{S}^\top Q / \sqrt{d_k}.$$

再由 $Q = XW_Q, K = XW_K, V = XW_V$:

$$\bar{W}_Q = X^\top \bar{Q}, \quad \bar{W}_K = X^\top \bar{K}, \quad \bar{W}_V = X^\top \bar{V}.$$

X 收到各分支合计：

$$\bar{X} = \bar{R} + \bar{Q}W_Q^T + \bar{K}W_K^T + \bar{V}W_V^T.$$

此式针对本节定义的简化残差模块，完整 Pre-LN Block 还要穿过对应归一化与 FFN，不能把它冒充整个完整层的梯度。

10.8 FFN 与 LayerNorm 如何接上

令 $B = XW_1 + b_1, C = \sigma(B), Y = CW_2 + b_2$ ，给定 $\bar{Y}$ ，则：

$$\bar{W}_2 = C^T \bar{Y}, \quad \bar{C} = \bar{Y}W_2^T, \quad \bar{B} = \bar{C} \odot \sigma'(B),$$

$$\bar{W}_1 = X^T \bar{B}, \quad \bar{X} = \bar{B}W_1^T.$$

偏置梯度对位置求和， $\odot$ 表示逐元素乘法。残差外包一层时，再加直连梯度。

对单行 LayerNorm，令 $s = \sqrt{v + \epsilon}, \hat{x} = (x - \mu)/s$ ，已知输出梯度 g ，并令 $u = g \odot \gamma$ 。对均值、方差与商逐项使用链式法则，可以合并为：

$$\bar{x} = \frac{1}{s} [u - \text{mean}(u)\mathbf{1} - \hat{x} \text{mean}(u \odot \hat{x})].$$

这里 mean 沿特征维计算；同一行有 $\bar{\gamma} = g \odot \hat{x}, \bar{\beta} = g$ ，多行时相加。该式保留了 ϵ ，不需要假设 $\text{mean}(\hat{x}^2) = 1$ 。它可由 $d\mu = \text{mean}(dx)$ 、 $dv = 2 \text{mean}((x - \mu) \odot dx)$ 、 $ds = dv/(2s)$ 代入 $d\hat{x}$ 推得。

这些局部规则按逆序组合，就能覆盖第 8 章选定的整个完整网络。标准自动微分框架实现的正是这种规则组合，不是对每个参数反复试错猜方向。

10.9 最后回到输入嵌入

若 $x_i = E_{t_i,:} + p_i$ ，则：

$$\bar{E}_{w,:} = \sum_{i:t_i=w} \bar{X}_{i,:}$$

可训练的位置向量同样收到对应梯度；固定位置编码不更新。若 $U = E^T$ ，还需加来自输出路径的 $\bar{U}^T$ 。训练更新可以同时影响“输入如何表示”和“输出如何打分”，无需分开进行两个训练过程。

第 11 章：一个小模型的完整前向与更新

[返回目录](#)

11.1 边界与全部设定

为了把数值算到底，本章使用单头因果注意力 + 输出投影 + 残差 + 词表输出的小模型。它省略位置编码、归一化和 FFN，因而不是完整 GPT。第 8 章提供完整结构，本章只核验核心计算链，不能从两位置实验推断大模型能力。

玩具词表只有 A、B、C，编号为 1、2、3。它们是抽象符号，不声称分别等同于真实医学 Token。训练样本的输入前缀是 A、B，正确下一个 Token 为 C。只对最后位置计损失。

取 $N = 3, n = 2, d = d_v = 2, d_k = 1$ 。所有可训练矩阵初值完整指定为：

$$E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}, \quad W_Q = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad W_K = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

$$W_V = W_O = I_2, \quad U = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}.$$

没有偏置，不共享 E 与 U。共有 $6 + 2 + 2 + 4 + 4 + 6 = 24$ 个参数元素。C 的嵌入行暂时未被输入查到。

11.2 查表、投影与掩码

输入为：

$$X = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad Q = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad K = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad V = X.$$

因为 $\sqrt{d_k} = 1$ ，点积分数与加掩码结果分别为：

$$S_0 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad S = \begin{bmatrix} 0 & -\infty \\ 1 & 0 \end{bmatrix}.$$

记 $a = e/(e+1) \approx 0.731059$ ， $c = 1/(e+1) \approx 0.268941$ ，则：

$$A = \begin{bmatrix} 1 & 0 \\ a & c \end{bmatrix}, \quad Z = AV = \begin{bmatrix} 1 & 0 \\ a & c \end{bmatrix}.$$

第 2 个位置实际读取了 A 的内容和自己的内容。这里的偏好只是指定参数产生的结果，尚未代表任何语言知识。

11.3 残差、词表分数、损失

由 $R = X + ZW_O$ ，取最后一行：

$$h = (a, 1 + c) \approx (0.731059, 1.268941).$$

词表分数与概率为：

$$\ell = hU = (a, 1 + c, -a),$$

$$p \approx (0.339660, 0.581626, 0.078714).$$

概率最大的是 B，但目标是 C，因此：

$$L = -\ln p_C \approx 2.541928.$$

没有跳过“预测函数”：概率由 h 、 U 、Softmax 明确算出，而不是凭空给定 30%。训练也不需要真的选出 B；它直接使用目标 C 的概率计算损失。

11.4 从损失反传到注意力分数

先求：

$$\bar{\ell} = p - (0, 0, 1) \approx (0.339660, 0.581626, -0.921286).$$

再求：

$$\bar{h} = \bar{\ell} U^T \approx (1.260945, 0.581626).$$

只有最后位置计损失，故 $\bar{R}$ 第一行为零，第二行为 $\bar{h}$ 。因为 $W_O = I$ ， $\bar{Z} = \bar{R}$ 。又因 V 为单位矩阵， $\bar{A} = \bar{Z} V^T = \bar{Z}$ 。

第二行 Softmax 的分数梯度第 1 项是：

$$\bar{S}_{21} = a(1.260945 - a \cdot 1.260945 - c \cdot 0.581626) \approx 0.133562.$$

另一项约为 -0.133562 。第一行只有一个允许位置，其 Softmax 恒为 1，不随唯一有限分数变化，所以第一行分数梯度为零。

于是：

$$\bar{S} \approx \begin{bmatrix} 0 & 0 \\ 0.133562 & -0.133562 \end{bmatrix}.$$

这正是原问答中缺少的中间桥梁：目标错误先影响词表分数，再影响 h ，再影响内容混合比例，最后才给出匹配分数应怎样局部调整。

11.5 回到各个参数，连输入层也算出

由第 10 章公式得到：

$$\bar{W}_Q \approx \begin{bmatrix} 0 \\ 0.133562 \end{bmatrix}, \quad \bar{W}_K \approx \begin{bmatrix} 0.133562 \\ -0.133562 \end{bmatrix}.$$

$$\bar{W}_V = \bar{W}_O \approx \begin{bmatrix} 0.921825 & 0.425203 \\ 0.339120 & 0.156423 \end{bmatrix}.$$

本例两矩阵梯度相等是初值特意简单造成的，不是普遍恒等式。

$$\bar{U} \approx \begin{bmatrix} 0.248311 & 0.425203 & -0.673514 \\ 0.431008 & 0.738049 & -1.169057 \end{bmatrix}.$$

把残差、Q、K、V 路径梯度加到输入，得到：

$$\bar{E} \approx \begin{bmatrix} 1.055387 & 0.425203 \\ 1.466503 & 0.871611 \\ 0 & 0 \end{bmatrix}.$$

C 没有作为输入出现，且本例不共享权重，因此 C 的输入嵌入行梯度为零；但 U 中 C 对应的输出列有非零梯度。这清楚地区分了“同一个 Token 的输入角色与输出角色”。

11.6 一次联合更新，重新计算

学习率取 $\eta = 0.1$ ，对所有矩阵同时使用旧参数处算出的梯度：

$$\theta_{\text{new}} = \theta_{\text{old}} - 0.1 \nabla L(\theta_{\text{old}}).$$

例如 W_Q 第二元素从 1 变成约 0.986644，U 第二行 C 列从 0 变成约 0.116906。不能先改完 U 再用新 U 给旧前向算梯度，那会混合不同计算点。

用全部更新后的参数重新前向，而非只修改最终概率，得到：

量	更新前	更新后
a_{21}	0.731059	0.713748
p_A	0.339660	0.271374
p_B	0.581626	0.556819
p_C	0.078714	0.171807
目标损失 L	2.541928	1.761382

正确目标概率提高，但仍不是最大项；一次更新不必立刻预测正确。注意力给第一个位置的比例反而下降，说明“学习进步”等于“所有看起来相关的注意力都应上升”是不成立的。更新同时改变内容、匹配、输入表示和输出打分。

11.7 怎样核验推导不是手写错了

配套程序独立按矩阵规则计算梯度，并对全部 24 个参数元素使用中心差分：

$$\frac{L(\theta + \epsilon e_r) - L(\theta - \epsilon e_r)}{2\epsilon}, \quad \epsilon = 10^{-5}.$$

与解析梯度的最大绝对差小于 3×10^{-11} 。程序同时重新前向验证上述损失下降。差分是有舍入和截断误差的数值核验，不能替代第 10 章一般公式的推导，也不能验证真实大型模型效果。

可以在配套资料中查看完整参数、梯度与更新结果和可运行核验程序。本章显示六位小数，程序全程使用未四舍五入值。

第 12 章：模型如何逐个输出文字

[返回目录](#)

12.1 为什么只用最后一行来预测下一项

每一行 h_i 受训练目标约束，用于预测当前位置之后的 Token。前面各行预测的是已经出现在输入里的后续位置；最后一行对应尚未知的下一位置。因此标准续写取最后一行词表分布。

它能依赖整个允许前缀，不意味着它无损保存了全部原文，也不是把句子压缩成一个独立、永久不变的句子向量。下次生成时，新位置仍可读取历史各层 K/V 。

12.2 从概率到 Token 还有一步选择

贪心解码： 取概率最大编号。它最大化当前一步的概率，不保证整个输出序列的联合概率最大。

采样： 按概率随机选。即使概率较小也可能被选中。Softmax 本身只给分布，不执行抽样。

温度： 对 $T > 0$ 使用 $\text{softmax}(\ell/T)$ 。较小 T 通常使分布更尖锐，较大 T 更平缓； $T = 0$ 不直接代入除法，而可讨论趋零极限。Top-k 或 top-p 还会截断候选并重新归一化。这些是解码规则，不是本次把模型训练了一遍。

12.3 用固定医学句子走一轮

已知输入为“患者服用华法林后出现”，本文玩具切分下有五个位置。完整模型算出 h_5 和词表分布，假设解码选中“黑便”。新前缀有六个位置，再用 h_6 预测下一项，假设选择“。”。

这只是可能的选择路径；没有给真实模型运行结果。若用户输入已是完整句子，则“黑便”属于已知上下文，不再作为下一步待填目标。

聊天模型还把角色标记、系统与历史消息等序列化上下文，在助手回答开始位置续写。学会遵循指令和对话格式还涉及相应数据与后续训练；仅讲基础下一个 Token 预训练并没有解释聊天产品的所有行为。

12.4 KV Cache 为什么能复用旧结果

在因果网络中，追加新 Token 不改变旧位置允许看到的前缀。参数固定、位置定义与计算配置不变时，旧位置各层 K/V 无需重新计算。保存它们，就能让新位置的 Q 读取旧 K/V 加上新位置自己的 K/V。

这依赖第 7 章的因果归纳性质。缓存要逐层保存，不是只保存最终向量。旧位置的中间表示在多层网络中早已参与构造后续所需的 K/V，并没有“只用最后一行所以前面都没用”。

窗口截断、位置重映射、参数改变等会影响缓存可复用条件。缓存通常以额外内存换取减少重复计算，也不消除新 Token 读取历史内容的代价。

12.5 到底什么时候停止

可因选中 EOS、达到长度上限或触发应用停止规则而结束。句号不自动等于结束。输出 Token 序列还要由 tokenizer 的解码规则变回可见文本，Token 与完整汉字、完整词语并非一一对应。

普通生成过程保持参数不变；上下文和缓存改变。把用户信息写进本次上下文，可以改变后续输出，并不等于本次进行了梯度训练。

第 13 章：简单运算为何能有复杂用途

[返回目录](#)

13.1 动态混合已经不是固定线性映射

单头无位置旋转时，可以写出：

$$Z = \text{softmax}_{\text{row}} \left(\frac{XW_Q W_K^T X^T}{\sqrt{d_k}} + M \right) XW_V.$$

虽然参数在推理中固定，混合矩阵 $A(X)$ 仍随输入改变。整体一般不满足可加性与齐次性，因此不是普通固定矩阵线性变换。每层重新对已上下文化的表示计算 Q/K/V，后层能使用前层构造的特征。

这解释“表达形式更灵活”，没有证明任何随机参数都会理解语言，也没有证明增加层数一定改善效果。

13.2 有用的语义从哪里来

训练目标奖励能预测文本的数据处理方式。有些预测需要句法、指代、事实关联或多步依赖；模型若学到可复用特征，可以在多个样本上减少损失。输入嵌入、内容混合和输出层共同适应，使隐藏向量成为后续计算可利用的表示。

这是训练机制与经验结果之间的解释，不能由矩阵公式单独推出“必然具有可靠推理能力”。表达能力、优化是否找到好参数、数据是否提供信号、泛化是否可靠，是不同问题。

13.3 注意力图不是现成的因果解释

较大注意力系数表示该头对某个 Value 使用了较大系数；它不直接量化最终答案受该词影响多少。Value 的方向与大小、其他头、投影、残差和后续非线性都会改变最终效果。

“华法林与黑便有关”可以作为教学情境，但不能在未测真实模型时声称某个头已经学到这个关系。第 11 章甚至展示了损失降低时某条注意力比例下降的情况。

13.4 学到训练目标，也可能学到捷径

对数损失以数据实际提供的后续为目标，不直接判断命题真伪。一段错误文本也能成为训练样本；降低该样本损失不等于提高事实可靠性。模型有能力拟合复杂关系，也可能依赖表面共现、偏差或记忆。

所以本文能严格建立的是“定义后的模型怎样计算、梯度怎样产生、给定例子怎样更新”；对现实模型理解、推理、可靠性的判断，需要额外实验与证据。

第 14 章：原始提问回查、练习与来源

[返回目录](#)

14.1 你的完整提问链与阅读位置

以下保留提问的先后顺序，选中文段只摘要，不重复整段旧回答。

原轮次与问题	对应补足
1 用我会的知识解释 Transformer 架构	完整任务、完整模块
2 点积变权重这里太快，不好理解	逐项点积与 Softmax
3 为什么 64 项方差是 64，标准差是 8	概率假设与推导
4 为什么方差可写成平方的期望	均值为零的条件
5 等等 dk 是什么来着	统一维度表
6 这一步是在算 WQ 还是 q	参数与中间结果
7 为什么又乘上了 V	混合规则与内容、矩阵乘法
8 没理解这个东西有什么高级的	输入决定混合
9 x 输出的 z 有什么意义	上下文文化表示及边界

原轮次与问题	对应补足
10 输入是一句话吧，到底训练还是输出	序列矩阵、共用前向
11 继续：最后一个位置为什么够用	最后位置与缓存
12 没有 M，每个词会联系其他词吧	允许集合与多层因果证明
13 用医学句子展示训练和输出	固定目标对齐、逐个生成
14 解释损失、梯度、参数更新	一般反向推导、完整数值
15 还得有个预测函数吧	词表输出层
16 输出层单独训练还是一起训练	统一目标与共享参数、嵌入反传

14.2 三种容易混淆的“学习”

参数训练：损失反传后更新参数。上下文适应：参数固定，用本次上下文改变激活与预测。工程记忆：应用存储信息并在未来检索或插入上下文。三者可配合，但不能互相替代为同一个概念。

14.3 自检问题与解答

练习 1。 $n = 5, d = 12, d_k = 3, d_v = 4, H = 2, N = 100$ ，Q、A、拼接结果、 W_O 、U 各是什么形状？

解： 5×3 、 5×5 、 5×8 、 8×12 、 12×100 。每个头的缩放因子为 $\sqrt{3}$ ，与 N 或 n 无关。

练习 2。 Softmax 分数 $(2, 1, 0)$ 全部加 100，会改变权重吗？全部乘 100 呢？

解：加常数不变，因为公共指数因子约去；乘 100 会把分数差扩大 100 倍，通常使权重更尖锐。相同最大值并列时，趋大尺度极限在并列最大项之间均分。

练习 3。 训练输入含“黑便”，为什么用“出现”预测它仍可行？

解：每一层都禁止“出现”位置读取后续，且其他模块逐位置运算。归纳保证未来信息不能绕路进入。只有数据数组里出现目标，并不等于计算该预测时能使用它。

练习 4。 为何第 11 章输入 C 行梯度为零，但输出 C 列非零？

解：输入只查 A/B 行，C 行未被使用；输出需要对整个词表打分，C 列直接参与目标概率。若共享 E 与 U，这两条路径的梯度会合并，结论随之改变。

练习 5。 知道 A 和 V 就能知道最终文字吗？

解：只能算出该头 Z；还需多头投影、残差、FFN、多层运算、最终输出打分与解码。即使知道最终分布，随机采样也不唯一确定文字。

练习 6。 为什么最小化 L 不等于每次都能提高准确率？

解：L 是平滑的概率目标，准确率通常是离散最大项是否正确。第 11 章目标概率提高但仍不是最大项；过大步长还可能增加损失，批次改善也不保证其他样本改善。

14.4 来源、依赖与本版范围

对话来源：解释 Transformer 架构。本地保留原始对话与读取核对报告。共 16 次用户提问；另有一条仅显示插件输出被隐去的消息，其原始内容无法恢复。本版重组全部可读问答，而非声称复原被隐去部分。

- R1 Vaswani 等，2017，[Attention Is All You Need](#)。架构、缩放点积、多头、前馈、位置与输出层的原始来源。本文自行展开计算，不沿用原文整段表述。
- R2 Bengio 等，2003，[A Neural Probabilistic Language Model](#)。神经概率语言模型与学习分布式表示的背景。
- R3 Ba 等，2016，[Layer Normalization](#)。归一化定义与方法来源。
- R4 Su 等，[RoFormer: Enhanced Transformer with Rotary Position Embedding](#)。旋转位置编码来源。
- R5 Devlin 等，[BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding](#)。双向编码器与不同预训练目标的对照。

- [R6 Press 与 Wolf, Using the Output Embedding to Improve Language Models](#)。输入输出权重共享的讨论。

数学基础依赖：有限维矩阵乘法、指数与对数求导、多元链式法则、一阶可微展开、概率链式法则，以及具有有限二阶矩时的期望与方差性质。本文展开与学习主线直接相关的推导，不从实数构造或测度论重建这些基础。

本版不包含真实模型训练、真实 tokenizer 验证或真实医学注意力测量。小模型数值核验与完整架构说明明确分开。后续可在这一基础上扩展真实小型 Transformer 实验，但不会把本版的数值示例标为已经训练出的语言能力。