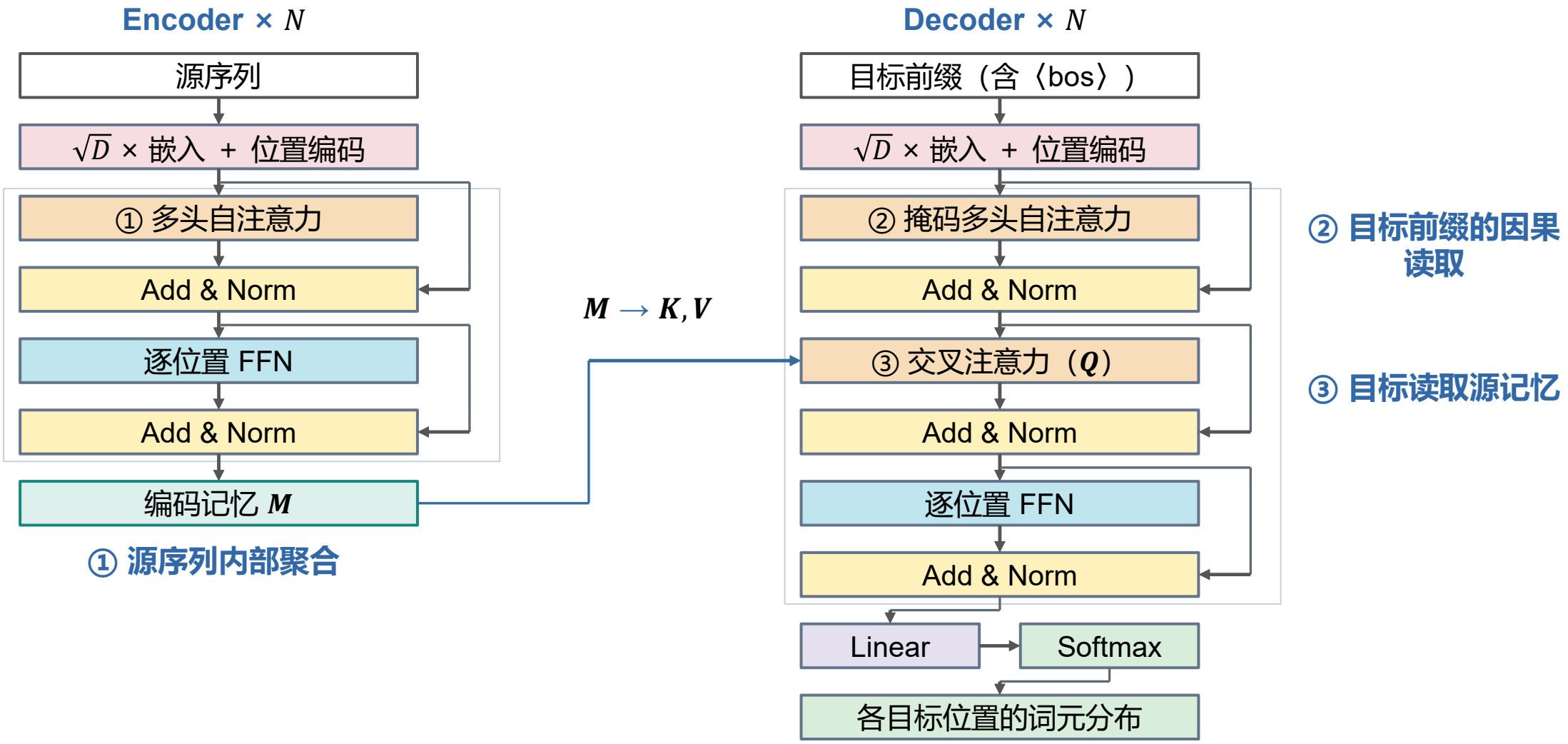
Transformer的结构与计算

机器学习 · 深度学习基础

学习目标

- 跟踪词元到输入表示，解释内容嵌入与位置表示的分工
- 说明编码器中注意力、前馈、残差和LayerNorm怎样组合
- 跟踪解码器读取目标前缀与源表示，并形成词表得分
- 判断填充与因果掩码允许的读取，区分键屏蔽和目标损失排除
- 沿同一短例解释目标移位、并行训练与逐步生成
- 按尺寸估算主要运算与存储，解释KV缓存的条件、步骤与代价

Transformer总体结构



编码器形成源记忆，解码器据前缀读取并预测；Add & Norm先残差相加，再归一化（**Post-LN**）。

1 有序序列的输入

建立词元、内容嵌入与位置表示。

词元索引经嵌入表变为逐位置向量

词元是序列中的离散单位。本例用词作单位；实际系统也可以使用子词等划分。

对象	本例内容	作用
源序列	我 / 爱 / 猫	待编码的输入
目标序列	I / love / cats / 〈eos〉	训练要预测的序列，含结束标记
〈bos〉、〈eos〉	开始、结束标记	指定生成起点和结束
〈pad〉	填充标记	对齐批量长度，后面单独屏蔽

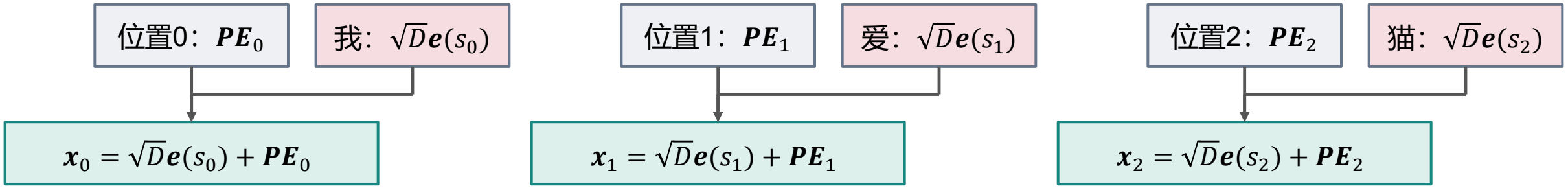
嵌入表为每个词表索引保存一个可学习向量。查表后，每个位置得到D维表示；索引只用于取行，不能直接当连续特征值使用。

源与目标可使用不同嵌入表。本讲给定划分与向量，不展开分词或词嵌入训练算法。

内容与位置共同形成输入

同一词元的嵌入查表结果不随位置改变。源端自注意力若只按内容匹配，交换词元顺序也会交换对应输出，无法据此表示词元的先后。

为给“我 爱 猫”与“猫 爱 我”提供不同的词序信息，每个位置加入同维的**位置编码**：



$$\boldsymbol{x}_p = \sqrt{D}\boldsymbol{e}(s_p) + \boldsymbol{PE}_p$$

其中，p从0开始，s_p为词元索引；内容嵌入和位置编码均为D维，相加后仍为D维。位置编码可固定或学习，本讲采用原始正弦 / 余弦构造。

同一词元在不同位置得到不同输入表示

同一内容嵌入加上不同位置编码，得到不同网络输入。 下表只核验这一点。

以下D为偶数，位置p从0开始，$r = 0, \ldots, D/2 - 1$：

$$PE_{p,2r} = \sin\left(p/10000^{2r/D}\right), PE_{p,2r+1} = \cos\left(p/10000^{2r/D}\right)$$

取$D = 4$，给定同一词的嵌入$\boldsymbol{e} = (1,0,-1,0.5)^{\mathsf{T}}$：

位置p	位置编码PE_p	加上$2e$后的输入x_p
0	(0, 1, 0, 1)	(2, 1, −2, 2)
1	(0.8415, 0.5403, 0.0100, 1.0000)	(2.8415, 0.5403, −1.9900, 2.0000)

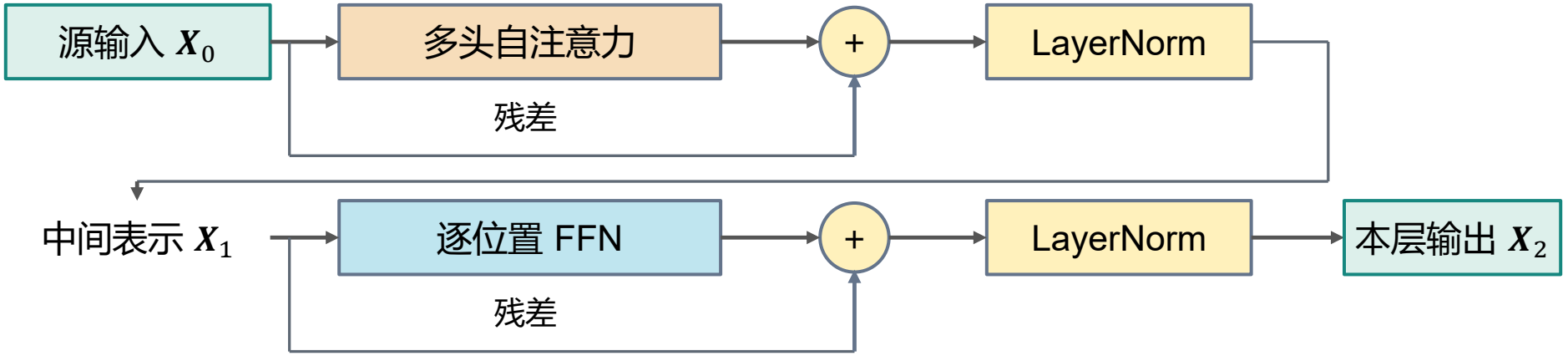
表内保留四位小数；第二组频率为$1/100$。内容向量相同、位置向量不同，因此网络输入随位置变化。编码数值提供信息，不能据此保证模型正确理解所有顺序关系。

2 编码器与解码器的组织

将注意力、前馈、残差与归一化组成完整网络。

编码器中的信息交互与表示变换

原始Post-LN编码层依次将$\boldsymbol{X}_0$更新为$\boldsymbol{X}_1$、$\boldsymbol{X}_2$：先聚合位置，再变换特征。

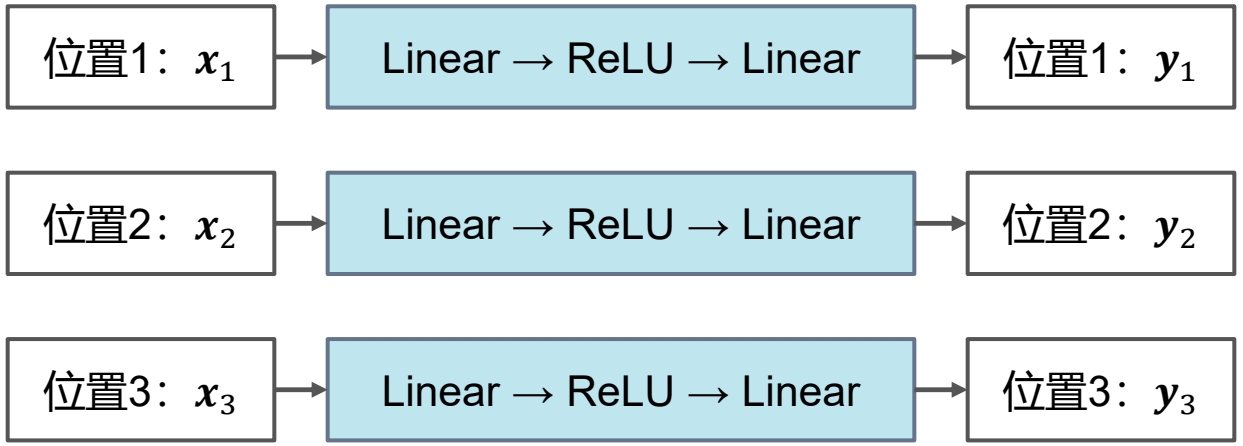


- 自注意力聚合源上下文；**逐位置前馈网络**（FFN）变换各位置的特征。
- 多层堆叠反复执行这两类计算，最终得到编码记忆$\boldsymbol{M}$。
  - 子层输出保留位置数和D维，与本子层输入逐元素相加。

逐位置FFN的参数共享

对逐行保存位置表示的矩阵$\boldsymbol{X}$，同一FFN作用于每一行：

$$FFN(\boldsymbol{X}) = ReLU(\boldsymbol{X}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2$$



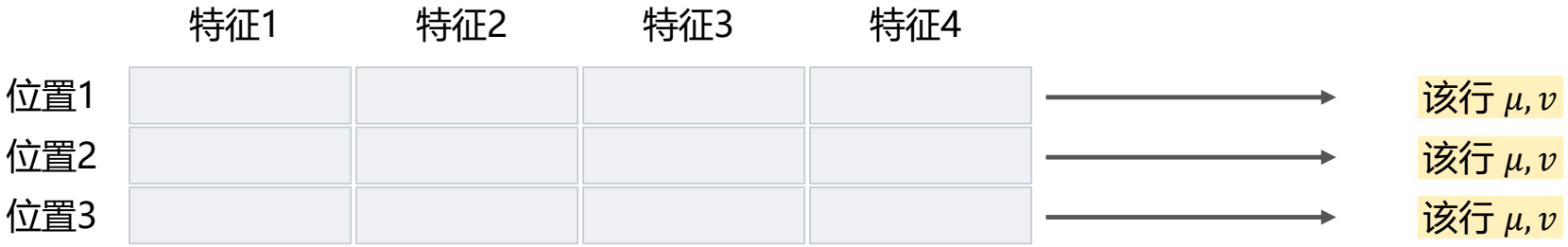
三条路径共享$\boldsymbol{W}_1, \boldsymbol{b}_1, \boldsymbol{W}_2, \boldsymbol{b}_2$：

- $\boldsymbol{W}_1: D \times d_{\text{ff}}$，$\boldsymbol{W}_2: d_{\text{ff}} \times D$；两组偏置按行广播。
- 中间扩展特征，输出恢复D维，位置数不变。

参数跨位置共享，但该子层不直接混合不同位置。输入已经包含注意力聚合的上下文；不同编码层通常使用不同FFN参数。

LayerNorm的统计范围

子层变换与残差相加会改变表示的数值尺度。**层归一化**（LayerNorm）先对每个位置的特征做标准化，再通过可学习的缩放和平移调整表示。



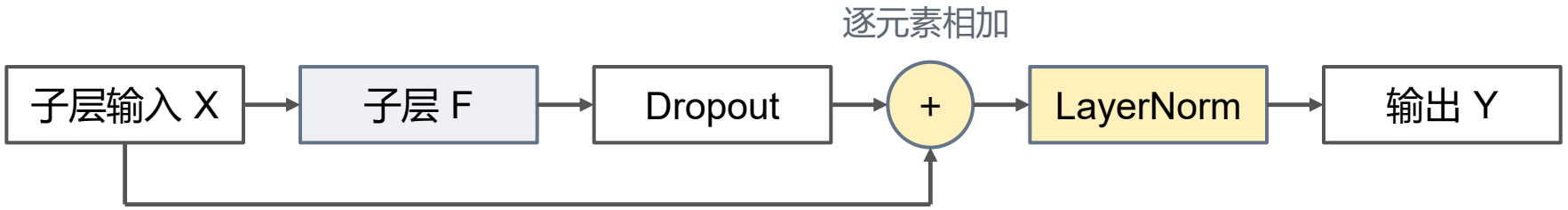
本配置分别对每个样本、每个位置的D个特征求均值和方差：

$$\mu = \frac{1}{D}\sum_j x_j\,, v = \frac{1}{D}\sum_j (x_j - \mu)^2, y_j = \gamma_j \frac{x_j - \mu}{\sqrt{v+\epsilon}} + \beta_j$$

- $\boldsymbol{\gamma}, \boldsymbol{\beta}$各为$D$维可学习参数，跨位置共享；$\epsilon > 0$。
- 均值和方差按当前行计算，不跨位置或样本；训练和评价均使用当前统计量。

Post-LN先完成残差相加，再做归一化

对注意力或前馈子层F，本讲采用后置层归一化（Post-LN），先残差相加，再归一化：



$$\boldsymbol{Y} = LN(\boldsymbol{X} + Dropout(F(\boldsymbol{X})))$$

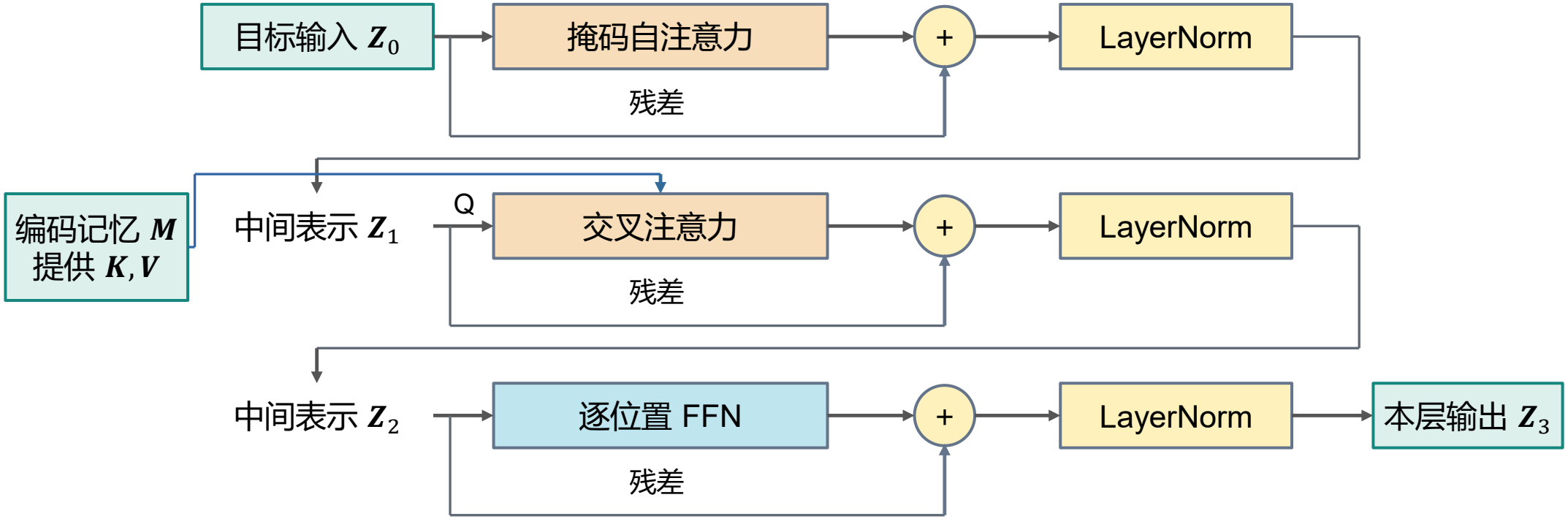
图中$\boldsymbol{X} \to \boldsymbol{Y}$泛指当前子层，可对应编码层的$\boldsymbol{X}_0 \to \boldsymbol{X}_1$或$\boldsymbol{X}_1 \to \boldsymbol{X}_2$。残差始终来自当前输入。

原始结构使用Dropout；本讲数值与依赖核验设其为0，以隔离确定性计算。

Pre-LN将归一化放在子层输入侧。本讲图与实现均采用Post-LN顺序。

解码器的三类子层与残差路径

目标输入Z_0先读取前缀，再读取源记忆，最后逐位置变换；Z_1, Z_2, Z_3为三次Post-LN输出：

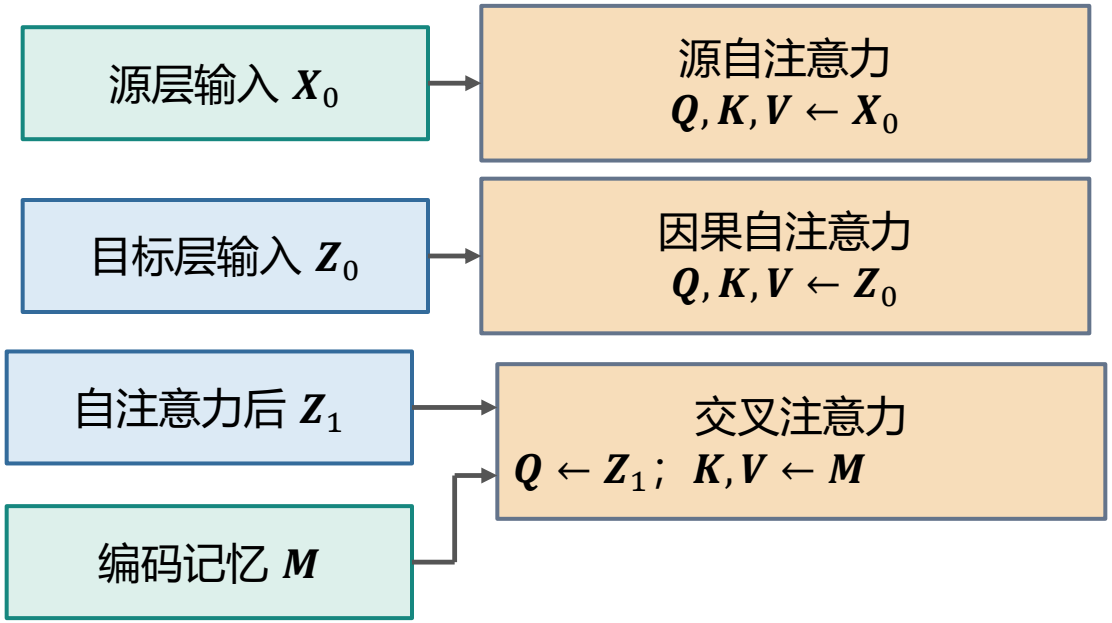


交叉注意力的Q来自前一子层输出，K, V来自编码记忆M。各子层分别残差相加后做LayerNorm。

Decoder堆叠N层后，才接词表映射与softmax；源位置数和目标位置数可以不同。

三种注意力的信息来源

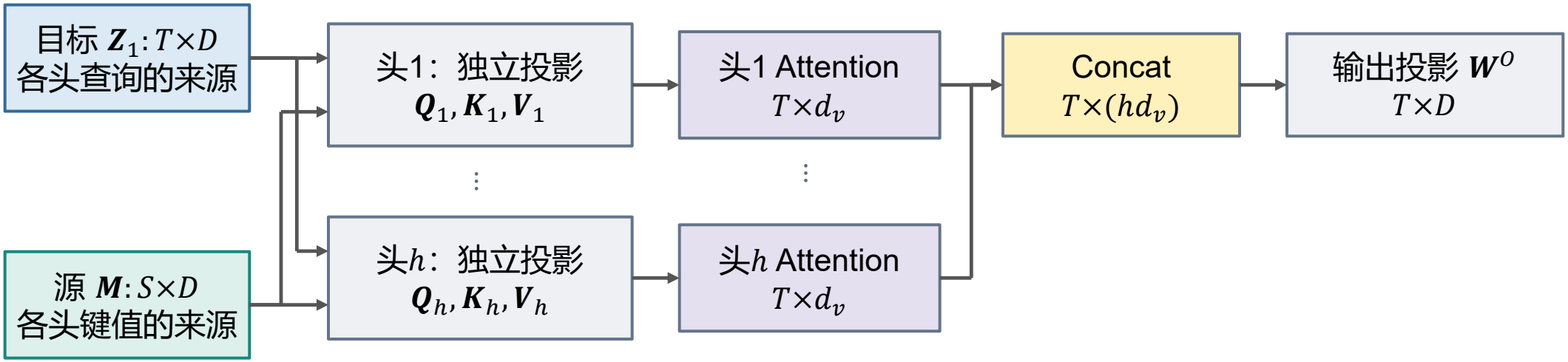
沿用前页状态：源层输入$\boldsymbol{X}_0$、目标层输入$\boldsymbol{Z}_0$，目标自注意力经Post-LN后为$\boldsymbol{Z}_1$。三种读取分别为：



源自注意力排除源填充键；目标自注意力同时施加因果与目标填充键约束；交叉注意力排除源填充键。$\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}$均由图示来源作各自可学习投影得到。

多头注意力的投影与合并

以交叉注意力为例：每头把目标中间表示$\boldsymbol{Z}_1$投影为$\boldsymbol{Q}_i$，把源记忆投影为$\boldsymbol{K}_i, \boldsymbol{V}_i$；各头独立计算后拼接并输出投影。

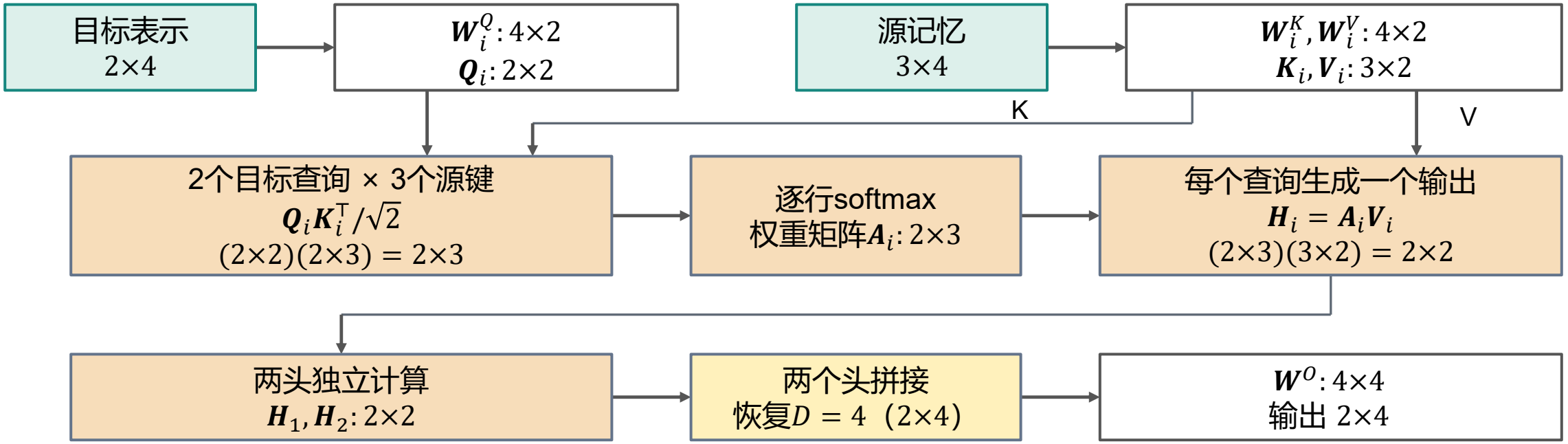


$Q_i: T\times d_k, K_i: S\times d_k, V_i: S\times d_v, Q_i K_i^{\top}: T\times S$

- 逐行softmax的权重乘$\boldsymbol{V}_i$得$T\times d_v$；合头为$T\times(hd_v)$，经$\boldsymbol{W}^O$映回$T\times D$。
- 标准等宽设置采用$d_k = d_v = D/h$；源长S与目标长T无须相等。

交叉注意力：两个目标位置怎样读取三个源位置

省略批量维，固定$S=3, T=2, D=4, h=2$，每头$d_k=d_v=2$。沿交叉注意力跟踪两个目标位置怎样读取三个源位置：



两个头拼接后恢复$D=4$；残差、LN和FFN保持2×4。

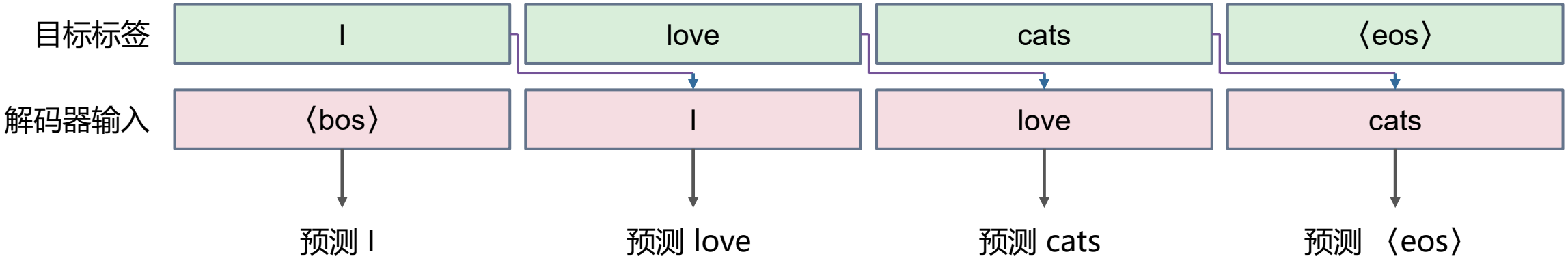
词表映射：$(2\times4)(4\times6)=2\times6$，逐行softmax得到两行分布。恢复批量维为$B\times T\times D \to B\times T\times V_t$；$V_t$为词表大小。训练用有效位置求损失，生成用末位置选词。

3 可读范围与并行训练

连接掩码、目标移位、有效损失和并行条件。

目标移位与预测对齐

另取$T = 4$的训练例，目标为 I / love / cats / 〈eos〉。前补〈bos〉、去掉末尾〈eos〉，得到解码器输入：



真实目标右移一位：前补〈bos〉，末尾〈eos〉不送入

输入给出前序词元，标签指定下一词元。训练中使用真实前缀称为**教师强制**（teacher forcing）。

训练要学习根据已有前缀预测下一词元，生成时才能按同一规则使用。**移位确定预测对象，因果掩码限制可读输入**；对角线读取已移位输入，不会泄漏当前标签。

掩码在归一化前排除不应读取的键

注意力掩码规定当前查询可读取哪些键。先修改缩放评分：允许位置加0，禁止位置加−∞；再做softmax。

$$\boldsymbol{A} = softmax_{\mathrm{row}}(\boldsymbol{S}+\boldsymbol{M}_{\mathrm{mask}})$$

其中，$\boldsymbol{S}$为缩放评分矩阵，$\boldsymbol{M}_{\mathrm{mask}}$为加性掩码。源序列填充到“我 / 爱 / 猫 / 〈pad〉”，给定四个评分均为0：

设置	softmax的输入评分	对四个源键的权重
未屏蔽填充键	(0, 0, 0, 0)	(1/4, 1/4, 1/4, 1/4)
屏蔽填充键	(0, 0, 0, −∞)	(1/3, 1/3, 1/3, 0)

填充位置只为对齐长度，不应提供内容。掩码改变归一化范围；不能只在softmax后随意置零而仍声称权重和为1。

因果掩码的可读范围

因果掩码使目标自注意力位置i只读取输入位置$j \le i$。目标输入已按前述规则右移，图中行对应查询，列对应键：

	输入键 j			
	〈bos〉	I	love	cats
查询 1	可读	×	×	×
查询 2	可读	可读	×	×
查询 3	可读	可读	可读	×
查询 4	可读	可读	可读	可读

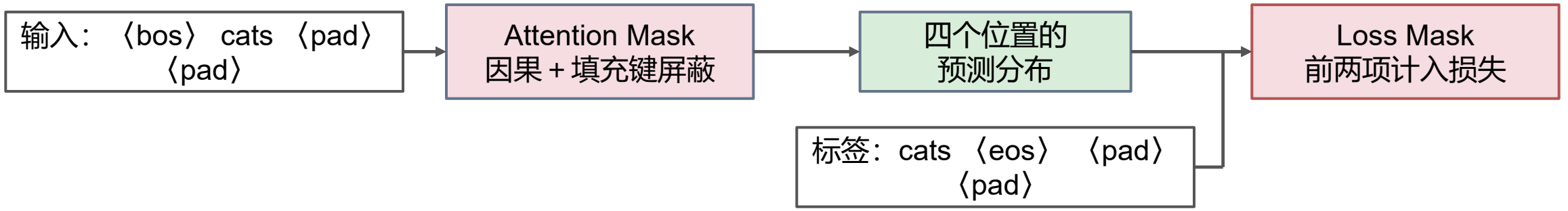
行：查询i　列：输入键j　有色可读（$j \le i$），灰色屏蔽

位置2可读〈bos〉和I并预测love；输入位置3中的love已被屏蔽。此处对角线读取的是已移位输入I，待预测的love不能被读取。

图中有色区域表示可读、灰色区域表示屏蔽；对应加性掩码分别为0与$-\infty$。

读取约束与损失约束

- 注意力掩码（Attention Mask）规定从哪里读取信息。
- **损失掩码**（Loss Mask）规定哪些预测参与训练。



- 屏蔽填充键后，填充查询仍可能产生输出；这些无效目标位置还需从损失中排除。
- 排除填充标签只影响损失，还须另行屏蔽注意力中的填充键。

编码器与交叉注意力使用源填充键约束；目标自注意力使用目标填充键及因果约束。本例源至少有一个有效位置，目标至少有开始标记。

交叉熵只对有效目标位置求平均

一个批量中还有短目标“cats / 〈eos〉”，它的输入为“〈bos〉 / cats / 〈pad〉 / 〈pad〉”，标签为“cats / 〈eos〉 / 〈pad〉 / 〈pad〉”。两个样本共6个有效目标位置。

设位置(b,t)对真实目标词元给出的概率为$p_{b,t}$，有效位置集合为$\mathcal{V}$：

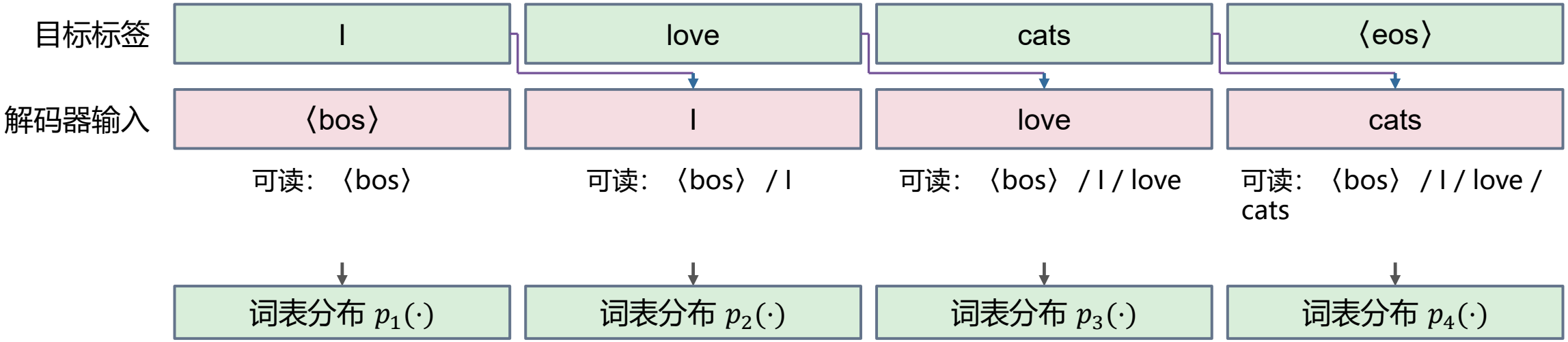
$$L=-\frac{1}{|\mathcal{V}|}\sum_{(b,t)\in\mathcal{V}}\log p_{b,t}$$

给定6项真实目标概率为(1/2, 1/4, 1/2, 1/8, 1/2, 1/2)，负对数之和为$9\ln 2$，平均损失约为1.0397。两个填充目标不进入分子或分母。

反向传播通过词表映射、解码器、交叉注意力及编码器更新相应参数。这里的概率是教学给定值，不是未训练模型的真实翻译表现。

因果约束下的并行训练

训练时，已移位的真实输入〈bos〉 / I / love / cats全部给定。因此可在同一次前向中计算各位置的受约束预测：



图中$p_i(\cdot)$是位置i对完整目标词表的分布，每一行只使用允许的前缀。有效标签选取对应概率计算损失，再汇总反向传播；所有位置共享同一网络参数。

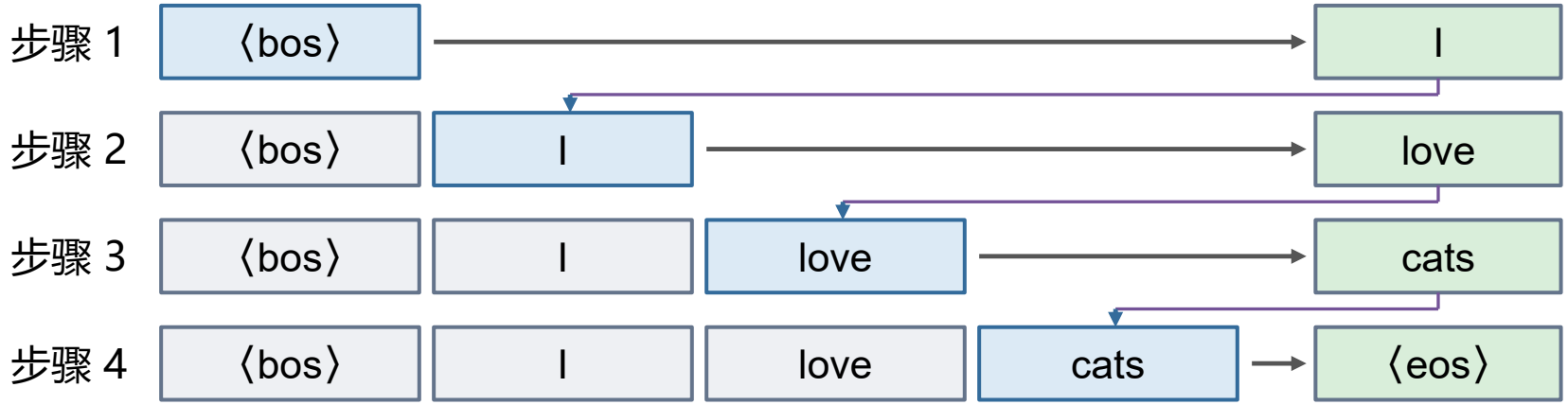
并行成立依赖输入已经可得；生成中的后续输入还需等待选词，不能据此一次得到完整未知序列。

4 逐步生成与任务接口

跟踪前缀增长并说明三种结构的使用方式。

自回归生成的前缀增长

源记忆M保持不变。从〈bos〉开始，每步用末位置分布选词，再追加为下步输入。



图中选词是给定的教学轨迹。源记忆和模型参数可复用，但下一步必须等待本步选词。遇到〈eos〉或达到长度上限时停止；实际系统还需指定贪心等选词规则。

训练与生成的计算组织

两种情形使用相同网络结构，关键差异在目标前缀从何处得到：

训练：真实前缀已知

移位后的真实词元同时输入

各有效位置同时求预测与损失

反向传播更新参数

生成：前缀随选词增长

〈bos〉及此前选中的词元

只用末位置分布选择下一词元

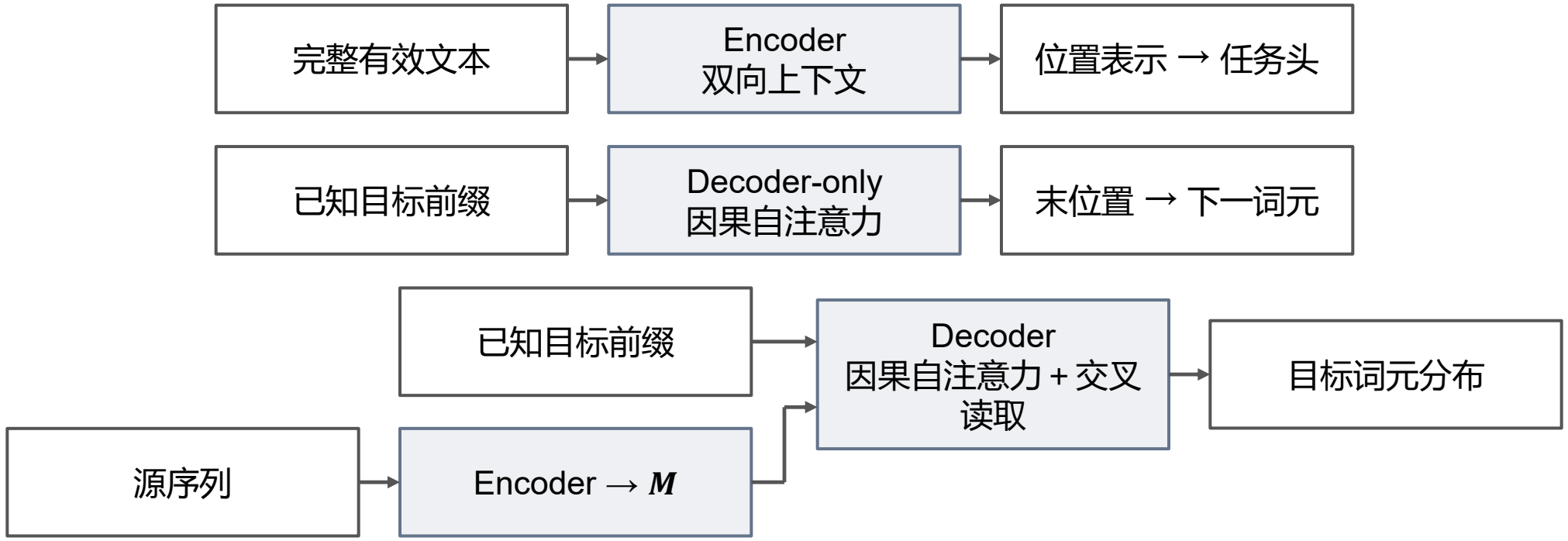
通常固定参数，选词后再前向

共同约束：当前预测只读取允许前缀，并通过交叉注意力读取有效源记忆

训练位置的并行来自真实输入已给定，生成步骤的顺序来自新输入尚未产生。早期生成结果会影响后续前缀，因此训练损失不能直接替代真实使用效果评价。

三种Transformer结构

三种典型结构保留不同的信息通路与输出接口：



第三行的Decoder同时接收目标前缀与源记忆$\boldsymbol{M}$，以前缀的中间表示查询源记忆。结构名称描述信息通路；分类、掩码预测或自回归训练还须规定具体输入、输出和学习目标。

BERT、GPT的预训练细节留给语言模型专题。

任务输入、可读范围与输出接口

沿前页三类结构，核对任务输入、可读范围和输出用途：

任务示例	输入与可读范围	输出怎样使用
文本分类	编码器读取完整有效文本	取指定分类位置或对有效位置汇聚，再接分类映射
按提示续写	提示作为解码器的已知前缀，按因果范围读取	末位置分布选下一词元，追加后继续
源到目标翻译	源由编码器表示，目标前缀经自注意力及交叉注意力	每个目标位置形成词表分布；训练和生成按前述流程使用

这些是任务接口示例，并非仅凭网络名称即可完成的能力。输入标记、任务输出映射、训练目标和参数是否适配，均需与实际模型核对。

本讲只定位接口，预训练和微调留待专题；下面回到这些通路的计算代价。

5 计算量、存储与增量生成

前缀逐步增长：哪些计算在重算，哪些结果可复用？

投影、注意力与前馈的计算量

投影与FFN逐位置计算；标准密集注意力还要处理位置对。

取单层、单样本，长度n、总宽D、h个头，每头宽D/h，FFN中间宽d_{ff}。按**乘加次数（MAC）**统计主要矩阵乘法，一次乘法加累加计1次。

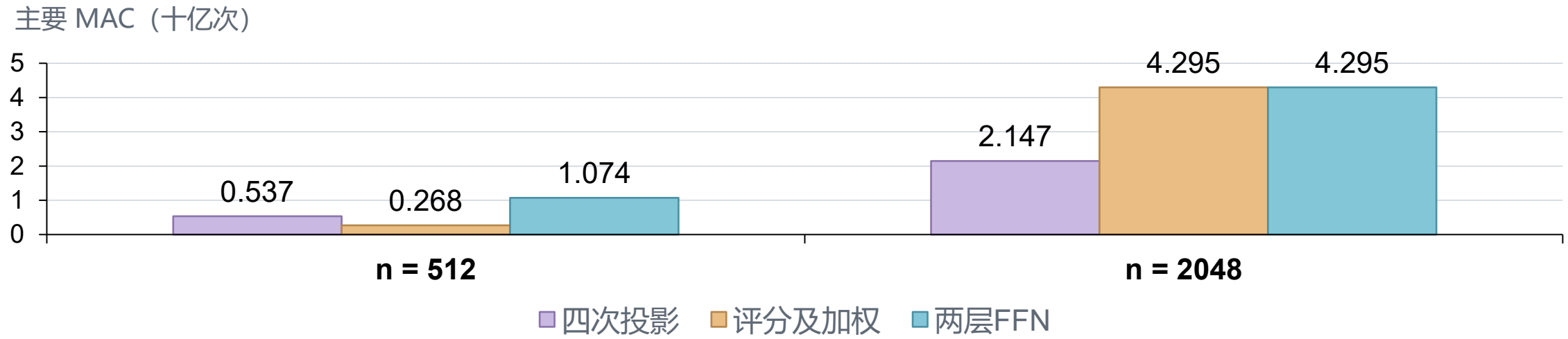
计算对象	对应矩阵乘法	MAC	增长规律
Q/K/V及输出投影	$3(n\times D)(D\times D)$；$(n\times D)(D\times D)$	$4nD^2$	**逐位置，线性**
评分$\boldsymbol{QK}^{\top}$及加权$\boldsymbol{AV}$	每头$(n\times D/h)(D/h\times n)$及$(n\times n)(n\times D/h)$	$h\cdot 2n^2(D/h)=2n^2D$	**位置对，二次**
两层FFN	$(n\times D)(D\times d_{\text{ff}})$；$(n\times d_{\text{ff}})(d_{\text{ff}}\times D)$	$2nDd_{\text{ff}}$	**逐位置，线性**

主要MAC总量：$4nD^2+2n^2D+2nDd_{\text{ff}}$。固定$D$时，各头宽度$D/h$抵消头数$h$，不能再乘一次头数。

未计偏置、激活、softmax、LN、残差、反向、词表输出及交叉注意力；不是完整训练运算量或实际耗时。1 MAC约为2 FLOP。

长度增长与主要计算量

单层、单样本，固定$D = 512$、$d_{\mathrm{ff}} = 2048$，按前页主要矩阵乘法统计：



长度扩大4倍，投影与FFN扩大4倍，注意力评分及加权扩大16倍。短序列时FFN仍可能占更多计算，整层不能只用注意力二次项概括。

显式保存hn^2个注意力元素，$h = 8$且每元素2字节时，两种长度的一份矩阵约为4 MiB与64 MiB；层数、批量及其他运行空间另计。

并行性、计算量与实际速度

所问问题	Transformer的相关特点	必要条件或代价
位置能否同时计算	已知输入时，同层位置可用矩阵运算并行	深层仍按依赖执行；自回归输出步骤仍顺序发生
远处信息经过多长路径	全局自注意力可直接联系允许的位置	仍需学习有效权重，且计算所有位置对有开销
实际要耗时多久	取决于运算、访存及设备利用率	不能仅由复杂度或理论并行推得加速倍数

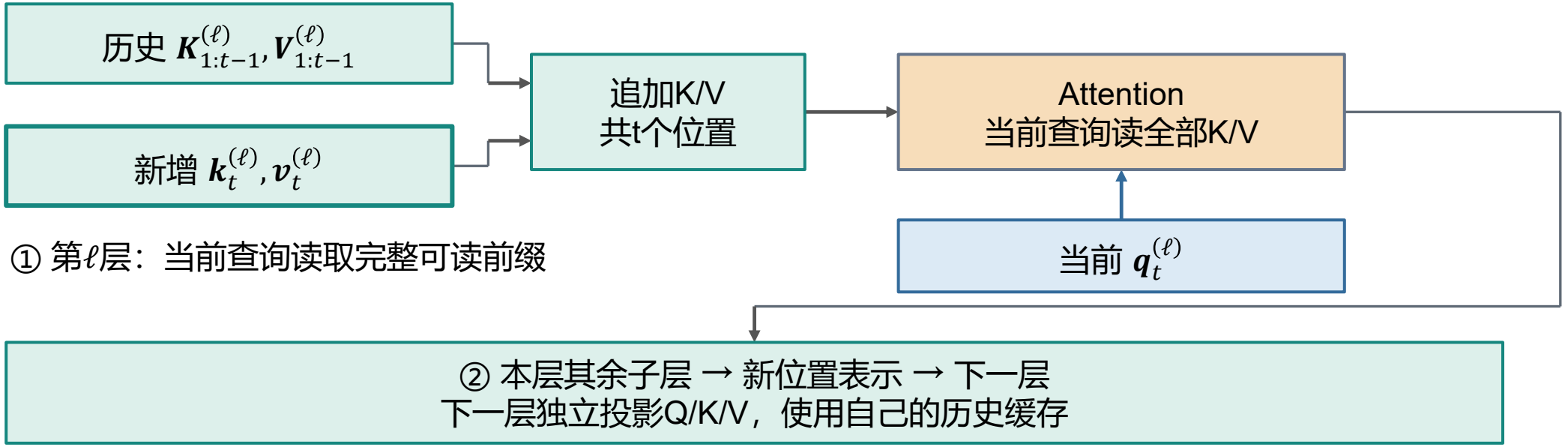
RNN沿位置递推状态来传播信息；Transformer用自注意力让允许的位置直接交互。输入已给定时，同层位置可并行，但标准密集注意力仍需处理大量位置对。

当显式注意力矩阵占用过大时，FlashAttention等精确注意力实现通过分块和重计算减少显存读写，避免完整矩阵驻留；密集注意力的主要乘法量仍随n^2增长。

因此比较效率要同时说清长度、批量、模型尺寸、精度、硬件及实现。这里给的是计算与存储估算，不是速度测试。

因果依赖与逐层KV复用

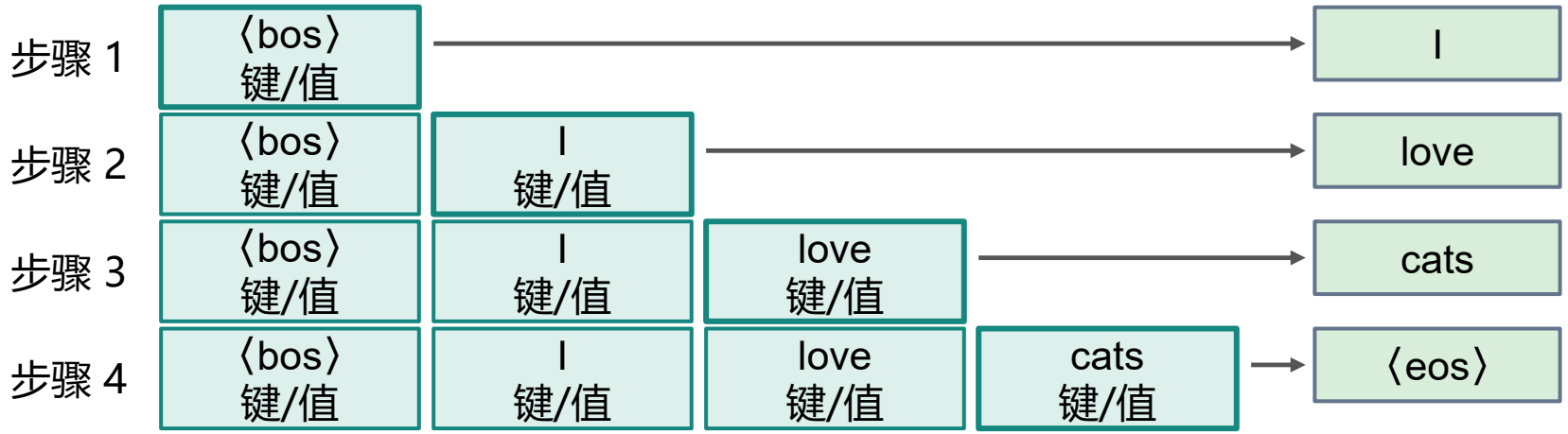
前提：旧前缀、参数、位置处理和源记忆固定，关闭推理随机性。因果约束下，旧位置不读新增位置，因此旧表示及其K/V可复用。



当前q_t仍读取历史与新增K/V；缓存不保存旧注意力输出。层间只传递新位置表示，各层缓存独立；位置编号连续，前提改变须重验。

增量生成中的缓存增长

下图展示同一生成轨迹中单层缓存的增长，各层分别维护相同前缀长度的历史。



每格代表该层一个位置的键值对；浅青绿为复用，加粗青绿框为本步新增

本步输入先写入K/V，再预测下一词元；预测结果到下一步才入缓存。第4步输入cats，预测结束标记后停止；结束标记未入缓存。

以完整前缀重算作对照：有缓存时只新增一个位置K/V，当前Q仍读取全部历史与新增键值。缓存不会消除生成步骤依赖。

已知长提示可一次预填充（prefill），再逐词解码（decode）；固定源记忆在各交叉注意力层的键值投影也可复用。

KV缓存减少多少重复计算

无缓存重算多个查询；有缓存只计算一个新查询，仍读取全部历史键值。

单层、单样本，固定D；第t步输入t个位置，累计T步（含〈bos〉）。只数评分与加权乘法：

计数范围	无缓存：密集重算完整前缀	有缓存：只算当前查询
第t步MAC	$2t^2D$：t个查询各读t个键值	$2tD$：1个查询读t个键值
累计T步MAC	$2D\sum_{t=1}^{T} t^2 = \frac{DT(T+1)(2T+1)}{3}$	$2D\sum_{t=1}^{T} t = DT(T+1)$
增长阶	$O(T^3D)$	$O(T^2D)$

上表采用每步密集重算完整前缀的基准，不代表所有无缓存实现。未计投影与FFN；缓存也省去它们的旧位置重算。三角计算可改变常数。

计算减少的代价，是在各层保存随前缀增长的历史K/V；下一页核算这部分存储。

KV缓存增加多少存储

每层保存历史K和V，位置越多，缓存越大；当前查询仍要读取它们。

$$\boxed{2} \times \boxed{B} \times \boxed{L} \times \boxed{T} \times \boxed{D}$$

2	B	L	T	D
K与V 两类张量	批量大小	层数	已缓存 位置数	每层K或V 的总宽度

$$N_{\text{elements}} = 2BLTD$$

标准多头注意力中，K和V各以总宽D保存，已包含全部头，不再乘头数。

取$B = 1, L = 24, T = 4096, D = 1024$，每元素2字节： $2\times1\times24\times4096\times1024\times2 = 402653184$ 单位为字节，除以2^{20}得 **384 MiB**（目标侧自注意力KV缓存）。

上式为目标自注意力KV；不含交叉缓存、参数或其他运行空间。缓存仍需读历史、等待上步选词；不能直接套用到双向编码器。

本讲总结

- **输入**：词元经嵌入查表，再与同维位置表示组合
- **编码**：注意力跨位置聚合，前馈逐位置变换；残差与LN按Post-LN顺序组织
- **解码**：目标自注意力读取前缀，交叉注意力读取源记忆，输出映射形成词表得分
- **约束**：填充及因果掩码限制读取，填充目标另从损失中排除
- **训练与生成**：移位明确预测对象；已知真实前缀支持并行训练，生成时逐步追加
- **计算与效率**：分别核算投影、注意力与FFN；因果解码可缓存逐层键值，节省重算并增加存储