arXiv:2603.11332v1 [cs.CC] 11 Mar 2026

# Transformer 的计算难度

Barna Saha*加州大学圣地亚哥分校 bsaha@ucsd.edu

Yinzhan Xu*加州大学圣地亚哥分校 xyzhan@ucsd.edu

Christopher Ye*加州大学圣地亚哥分校 czye@ucsd.edu

Hantao Yu†哥伦比亚大学 hantao.yu@columbia.edu

**摘要**

Transformer 架构彻底改变了现代人工智能在语言、视觉及其他领域的发展。它由 $L$ 层多头注意力组成，每一层都并行运行 $H$ 个注意力头，并将合并后的输出传递给下一层。在注意力机制中，长度为 $N$ 的输入中的每个词元都由一个维度为 $m$的嵌入向量表示。从计算角度看，注意力机制主要涉及三个 $N \times m$ 矩阵的乘法，同时对前两个矩阵的乘积所得到的中间结果应用 softmax 运算。大量研究致力于分析注意力机制的时间复杂度，并取得了若干近期进展。

另一方面，已知的 Transformer 算法会独立计算每个注意力头。这引出了一个基础性问题，该问题以“直接和”问题的形式贯穿理论计算机科学的发展历程：解决同一问题的多个实例，是否能比逐个单独解决每个实例更高效？通信复杂度和算法设计等领域针对这一问题，已有许多肯定和否定的答案。因此，要理解 Transformer 的计算困难性，一个关键挑战是确定其计算能否比 $LH$次独立的注意力计算更高效地完成。

本文对这一问题给出了否定的回答，并为多头多层Transformer给出了首个非平凡的计算下界。在小嵌入情形（$m = N^{o(1)}$）下，分别计算 $LH$个注意力头需要 $LHN^{2+o_{(1)}}$时间。我们证明，在强指数时间假设（SETH）下，这一复杂度基本上是最优的。在大嵌入情形（$m = N$）下，可以使用$LHN^{\omega+o(1)}$ 次算术运算（以及指数运算）分别计算 $LH$ 个注意力头，其中 $\omega$ 是矩阵乘法指数。我们通过证明当 $\omega > 2$时，计算至少需要 $LHN^{\omega-o(1)}$ 次算术运算，证明了这一界是最优的。我们在大嵌入情形下的下界依赖于对Baur-Strassen定理的一种新颖应用；该定理是一种强大的算法工具，也是著名的反向传播算法的理论基础。

*本研究获美国国家科学基金会 HDR TRIPODS 第二阶段项目资助（拨款编号 2217058；EnCORE研究所）。

†本研究获美国国家科学基金会拨款 CCF2008733、美国海军研究办公室拨款 N00014-22-1-2713、美国国家科学基金会拨款 CCF-2238221，以及哥伦比亚人工智能技术中心拨款资助。

# 1 引言

Transformer 架构 [VSP+17] 广泛应用于自然语言处理 [RWC+19] 和计算机视觉等诸多任务 [DBK+21, KMR+23]并在众多应用中取得了最先进的性能。Transformer的关键构建模块是注意力头，它能够捕捉输入词元之间的成对关系，因此适合处理语言输入。具体而言，给定输入 $X \in \mathbb{R}^{N\times m}$，注意力头由查询、键和值嵌入映射组成 $Q$， $K$， $V$： $\mathbb{R}^{N\times m} \to \mathbb{R}^{N\times m}$，其中 $m$ 称为嵌入维度，并计算

$$\mathrm{Att}_{Q,K,V}(X) = \mathrm{softmax}(Q(X)K(X)^\top)V(X) \in \mathbb{R}^{N\times m},$$

其中 softmax算子定义为

$$\mathrm{softmax}(v) = \frac{1}{\sum_{i=1}^{N}\exp(v_i)} \cdot (\exp(v_1), \ldots, \exp(v_N))$$

并按行应用于矩阵。Transformer 由 $L$ 个注意力层组成，其中每个注意力层包含 $H$ 个并行注意力头，它们将前一层的输出 $X$ 作为输入，并将其发送给所有注意力头，然后汇总所有注意力头的输出 [1] ，最后应用逐行多层感知机（MLP）函数 $X$。

Transformer 虽然无疑功能强大，但计算开销也很高，其注意力计算复杂度随输入长度 $N$ 呈二次增长。计算注意力的平凡算法需要 $O(N^2m)$ 时间，[2]而计算 Transformer 的平凡算法需要 $O(LHN^2m)$ 时间，尚未计入计算嵌入和多层感知机映射所需的时间。[3] 已有证明表明，若 $m = \Omega(\log N)$，则在强指数时间假设（SETH） [KWH23, AS24], 的条件下，计算注意力至少需要 $N^{2-o(1)}$ 时间；而在 OV 假设的一般化形式 [GHS+26]下，对于所有 $m$ ，显式计算矩阵乘积的平凡算法都是最优的。由于缺乏快速注意力算法，许多研究尝试通过提出注意力机制的次二次替代方案来解决这一瓶颈[KKL20, CLD+21, ZGD+20b, LYS+25]。然而，随着模型规模扩大，这些次二次替代方案都会导致精度下降；并且已有证明表明，不存在任何次二次替代方案能够在某种近似 [AY25]下捕获所有成对关系。因此，我们的工作主要关注以下问题。

**问题 1：** 标准*Transformer*模型的计算效率有多高？

虽然 [KWH23、AS24、GHS+26] 研究了单个注意力头的这一问题，但我们关注的是由许多层多头注意力组成的 Transformer。针对 Transformer 证明下界一直是一项重大挑战。事实上，Transformer 似乎比注意力头复杂得多。即使只考察表达能力（即可以计算哪些函数），证明下界的技术也相对丰富

[1]注意力输出可以通过求和或拼接进行聚合。在第 C 节中，我们将求和聚合归约为拼接聚合。因此，不失一般性，我们可以考虑求和聚合。 [2]按照实践中的取值，我们始终假设 $m \le O(N)$。 [3]计算多层感知机所需的时间最终取决于其隐藏维度 $d_{\mathrm{hid}}$。对于 $\mathbb{R}^m$ 中的输入，实践中计算多层感知机所需的时间大致应为 $O(md_{\mathrm{hid}})$。本文为不含多层感知机、且其嵌入映射可在线性时间内计算的 Transformer 给出了下界 $O(Nm)$。在第 B 节中，我们将介绍如何把这些下界推广到采用常见多层感知机的 Transformer。请注意，这只会加强下界，因为这些是 Transformer 的特定实例。

针对注意力机制或单层 Transformer 的下界（例如，单向通信 [SHT23] 以及拆分 VC 维数 [KUJ+25]），但直到最近才有一项突破性成果，为具有多层结构的仅解码器 Transformer 证明了强下界 [CPW25]。

研究 Transformer模型的复杂度，进一步受到一些特定任务已知下界的启发；我们希望 Transformer 能够处理这些任务。许多简单而基础的问题都需要大型 Transformer。例如，有人猜想，任何能够计算 3-SUM 的 Transformer 都需要多项式规模（即 $mHL = N^{\Omega(1)}$） [SHT23, SHT24b]。此外，浅层解码器 Transformer 若没有多项式宽度，就无法计算函数复合 [CPW25]。尽管如此，如果我们仍想用 Transformer 计算此类函数（例如，3-SUM 自然地捕捉了输入词元三元组之间的交互），那么尽可能高效地计算大型 Transformer 就至关重要。

除了显而易见的实际动机之外，研究 Transformer模型的复杂度本身也是一个有趣的理论问题。理解 Transformer模型的复杂度，本质上就是要弄清楚，我们能否比逐个计算更快地计算多个注意力头。我们还可以更一般地提出这样的问题：

> **问题 2：** 同时求解同一问题的多个副本，其成本能否通过摊销，比单独求解每个副本更高效地降低？

这是理论计算机科学中一个反复出现的问题，通常称为“直接和”问题，针对这一问题有许多正面和反面例子。

从正面来看，通信复杂度中的一个著名结果给出了一个部分函数：其复杂度为 $\Theta(\log n)$，但同时计算该函数的许多实例时，摊销复杂度为常数 $O(1)$ [FKNN95]。近期，一项突破性成果针对全函数也得到了类似结果~total functions [MS25]。一个次数为 $n$的多项式可在 $n$ 个点上用 $O(n)$ 时间求值 [Fid72]，而朴素算法需要 $\tilde{O}(n^2)$ 时间。类似地，一个 $n \times n$ 矩阵与 $n$ 个不同向量的乘积可在 $O(n^{2.372})$ 时间内计算 [ADV+25]，而分别计算所有乘积则需要 $O(n^3)$ 时间。此外，计算 Transformer 时，我们不必显式计算每个注意力头。例如，一层的输出是 $H$ 个注意力头的和。对于矩阵乘积，给定 $n$ 对 $n \times n$ 矩阵 $\{A_i, B_i\}$，其乘积之和 $\sum_i A_i B_i$ 可在 $O(n^{3.251})$ 时间内计算，比逐个计算每个乘积快得多；逐个计算需要 $O(n^{3.372})$ 时间 [ADV+25]。

在否定方向上，同样有大量研究成果。例如，对于输入相互独立的单调函数，计算效率无法高于分别计算每个函数 [GF81]。类似地， [Bsh89] 针对某些二次系统类别给出了直和定理。[BB19] 还为随机化查询复杂度证明了强直接和定理。 [MWY13]利用强直接和定理，为各种草图化和估计任务得到了分布式计算和流式下界。与矩阵向量乘法问题不同，人们普遍猜测，无法在强次立方时间内计算一个(最小值 $+$)-乘积，即一个 $n \times n$ 矩阵与 $n$ 多个向量的乘积 [VW18]。问题 2 自然属于这一长期研究脉络：探究 Transformer 是否满足直和定理。

在本工作中，我们从否定的角度基本解决了问题 1（或在注意力与 Transformer 情形下的问题 2）：我们给出了多头多层 Transformer 计算的首个非平凡下界，并证明分别计算每个注意力头的朴素算法本质上是最优的。

### 1.1 我们的贡献

我们对小嵌入维度（$m = N^{o(1)}$）情形和大嵌入维度（$m = N$）情形都给出了更强的下界。在小嵌入维度情形下，朴素算法的运行时间为 $LHN^{2+o(1)}$，而此前已知的关于注意力的条件下界 [AS24, GHS+26] 排除了运行时间为 $LHN^{1-o(1)} + N^{2-o_{(1)}}$ 的算法。我们给出了显著改进的条件下界，与朴素算法相匹配。

**定理 1.1**（非正式定理3.1）。 在 3-OV 假设或强指数时间假设下，任何计算 $L$ 层、$H$ 个注意力头、采用求和聚合且嵌入维度为 $m = \Omega(\log N)$ 的 *Transformer* 都需要

$$LHN^{2-o(1)}$$

时间。即使 *Transformer* 不含多层感知机，该下界仍然成立。

3-OV 假设指出，在由 $\{0,1\}^{\Theta(\log n)}$ 个向量组成的集合中，从 $n$ 个向量里找出一组三个两两正交的向量需要 $n^{3-o(1)}$ 时间。3-OV 假设是 $k$-OV 假设的一个特例；后者已被用于证明许多下界，例如 [BRS+18、Li21、Bon22、ATYZ24、DLV25]。此外，它还是强指数时间假设（SETH）的推论 [Wil04], 。SETH 是 P $\neq$ NP 的强化版本，也是细粒度复杂性领域中一项广为人知的猜想；它指出，对于任意常数 $\varepsilon > 0$，不存在针对 $n$ 个变量的可满足性问题的 $2^{(1-\varepsilon)n}$ 算法 [IP01] 。SETH 为包括编辑距离 [BI15], 、图直径[RV13], 、动态规划问题 [ATYZ24], 、近似最近点对问题 [ARW17],、子集和 [ABHS19] 在内的许多问题带来了几乎最优的下界。

尽管许多早期理论研究将 Transformer 的嵌入维度建模为大约 $O(\log N)$，近期研究也考虑了嵌入维度接近 $O(n)$ 的情形。 [YSBS+25] 发现，对于许多实际的图数据集，图的数量（输入长度）可能远小于实际使用的 Transformer 的嵌入维度。对于更大的嵌入维度，$m$定理1.1中的下界并不会更好。例如，当嵌入维度 $m$ 与输入长度相同（即 $m = N$）时，最佳运行时间反而更差：朴素上界为 $O(LHN^2m) = O(LHN^3)$，而使用快速矩阵乘法后，上界可改进为 $LHN^{\omega+o(1)}$，其中 $\omega$ 表示矩阵乘法指数。另一种下界可通过采用已有的单次注意力计算的条件下界 $N^{\omega-o(1)}$ [GHS+26] 得到。然而，最佳下界 $\max\{LHN^{2-o_{(1)}}\, N^{\omega-o(1)}\}$ 与最佳上界 $LHN^{\omega+o(1)}$之间仍存在很大差距，而且在这种设定下，Transformer 的时间复杂度仍未得到充分理解。

在这项工作中，我们证明，只要满足 $\omega > 2$，计算 Transformer 的难度就不亚于计算 $\Theta(LH)$ 个矩阵乘法实例的难度，从而在无需任何额外假设的情况下得到相匹配的 $LHN^{\omega-o(1)}$ 下界。

我们的结果是在算术电路模型的扩展版本中证明的，该模型是对研究充分的算术电路模型的推广（例如，参见 [BCS13]，其中有关于算术电路的介绍）。在标准算术电路模型中，算法只使用标准算术运算 $\{+, -, \times, /\}$。

由于注意力计算涉及取指数，目前尚不清楚如何用标准算术电路来计算注意力，也就是说，标准算术电路无法执行注意力计算，因为模拟指数函数本身就可能已经足够困难。因此，在我们所考虑的算术电路中同时允许指数门和对数门是很自然的，我们称之为扩展算术电路 （eAC）。我们指出，针对扩展算术电路的下界强于针对算术电路的下界，因为任何算术电路

都是一种扩展算术电路。此外，所有已知的 Transformer 计算算法都存在于扩展算术电路模型中。

借助快速矩阵乘法，我们很容易得到一个规模为$LHN^{\omega+o(1)}$的扩展算术电路，用于模拟嵌入维度为 $N$ 的 $H$ 头、 $L$ 层 Transformer 推理。我们证明，这基本上是最优的。具体而言，任何计算 Transformer 的扩展算术电路都需要规模 $LHN^{\omega-o(1)}$。

**定理 1.2** (非正式定理5.1)。 任何计算 $L$ 层 $H$ 头、采用求和聚合且嵌入维度为 $m = \Omega(N)$ 的 *Transformer* 的扩展算术电路，其规模至少为

$$LHN^{\omega-o(1)}$$

当 $\omega > 2$ 时。即使 *Transformer* 不含多层感知机，该下界依然成立。

我们指出，当 $\omega = 2$(即，具有 $m = O(N)$ 的 Transformer 可在 $LHN^{2+o(1)}$ 时间内计算时)，我们也可以改用定理 1.1 来得到匹配的 $LHN^{2-o(1)}$ 下界。关于将我们的结果推广到拼接聚合，请参见第 C 节。

我们关于 Transformer 的下界表明，当嵌入维度较大时，在扩展算术电路模型中计算 Transformer，除了快速矩阵乘法之外，不可能有显著的改进。我们提出了一种证明多层 Transformer 下界的新方法，而这一直是该领域文献中颇具挑战的研究方向。

## 1.2 技术概述

本节概述我们所采用的技术。

**小嵌入维度**。 我们先介绍小嵌入维度情形下的下界 $m = N^o$ 3 $A = \{a_1, \ldots, a_N\}$ $B = \{b_1, \ldots, b_N\}$ $C =$(1)。考虑 $\{c_1, \ldots, c_{LH}\} \subset \{0,1\}^d$ $d = \Theta$ $n$ 一个 -OV 实例 ,,, 其中某些参数为 (log )。注意，这三个集合的规模各不相同，我们称之为非平衡 3-OV 实例。已知，在标准 3-OV 假设成立的情况下，这类实例需要 $LHN^{2-o(1)}$ 时间。我们构造了一个 Transformer $T_C$，它具有 $L$ 层，每层有 $H$ 个注意力头，嵌入维度为 $O(d)$，使得 $T_C(A, B)$（经过一些简单的后处理）可以判定 $(A, B, C)$ 是否包含一个正交三元组。首先，我们说明 Transformer 的输入：

$$X = \begin{pmatrix} a_1^\top & b_1^\top & 1 \\ \vdots & \vdots & \vdots \\ a_N^\top & b_N^\top & 1 \\ 0_{1\times d} & 0_{1\times d} & 2 \end{pmatrix}.$$

除最后一个词元外，每个词元都由一个来自 $A$ 的向量、一个来自 $B$的向量以及常数 1 组成。现在我们描述输入维度为 $2d+1$ 的 $T_C$。 $T_C$ 将由 $L$ 层组成，每层有 $H$ 个注意力头，嵌入维度为 $m = 2d+2$。对于每个注意力头，我们将前 $2d+1$ 列中的 $V$ 设为 0，使得第 $\ell$ 层的输出（记作 $X^{(\ell)}$）在前 $2d+1$ 列中包含 $X = X^{(0)}$。特别地，我们可以假设每个注意力头的输入在前 $2d+1$ 列中包含 $X$ 。

我们现在描述每个注意力头的嵌入映射。由于 $|C| = LH$，我们可以对 $C = \{c_{h,\ell}\}$进行索引，并将 $C$ 中的每个向量与Transformer中的一个注意力头对应起来。考虑第 $\ell$层中的第 $h$个注意力头。我们按如下方式定义嵌入映射$q_{h,\ell}, k_{h,\ell}, v_{h,\ell}$： $\mathbb{R}^m \to \mathbb{R}^m$ 。请注意，由于我们的映射

不依赖于第 $m$个坐标，因此以下映射对于第一层仍然是良定义的，而第一层是一个映射 $\mathbb{R}^{m-1} \to \mathbb{R}^m$。

$$
\begin{aligned}
&q_{h,\ell} : u \mapsto -(u_1, \ldots, u_d, 0, \ldots, 0),\\
&k_{h,\ell} : u \mapsto (u_{d+1} \cdot c_{h,\ell}[1], \ldots, u_{2d} \cdot c_{h,\ell}[d], 0, \ldots, 0),\\
&v_{h,\ell} : u \mapsto (0, \ldots, 0, u_{2d+1}).
\end{aligned}
$$

注意，我们可以用适当的权重矩阵$W_{Q,h,\ell}$,$W_{K,h,\ell}$,$W_{V,h,\ell} \in \mathbb{R}^{m\times m}$ 对所有 $h, \ell$实现上述映射。将上述映射应用于所有 $N$ 词元，我们得到

$$
Q_{h,\ell}(X) = \begin{pmatrix} -a_1^\top & 0_{1\times(d+2)} \\ \vdots & \vdots \\ -a_N^\top & 0_{1\times(d+2)} \\ 0_{1\times d} & 0_{1\times(d+2)} \end{pmatrix}, \quad K_{h,\ell}(X) = \begin{pmatrix} b_1^\top \odot c_{h,\ell} & 0_{1\times(d+2)} \\ \vdots & \vdots \\ b_N^\top \odot c_{h,\ell} & 0_{1\times(d+2)} \\ 0_{1\times d} & 0_{1\times(d+2)} \end{pmatrix}, \quad V_{h,\ell}(X) = \begin{pmatrix} 0_{1\times(2d+1)} & 1 \\ \vdots & \vdots \\ 0_{1\times(2d+1)} & 1 \\ 0_{1\times(2d+1)} & 2 \end{pmatrix}
$$

其中 $\odot$ 表示两个向量的逐元素乘积，而 $0_{a\times b}$ （相应地， $1_{a\times b}$）表示一个以常数0（相应地，1）填充的 $a \times b$子矩阵。因此，第 $\ell$层中第 $h$个注意力头的输出为softmax($Q_{h,\ell}(X)K_{h,\ell}(X)^\top$)$V_{h,\ell}(X)$。为略微简化论述，我们转而考虑hardmax 注意力。hardmax 函数将向量 $v$ 映射到其最大元素上的均匀分布。也就是说，对于向量 $v$ ，其 $I_{\max}(v) = \{i$ 使得. $v_i = \max_j v_j\}$，hardmax($v$)$_i = \frac{1}{|I_{\max}(v)|}$若 $i \in I_{\max}(v)$，否则hardmax($v$)$_i = 0$ 。于是，一个hardmax 注意力头输出$Z_{h,\ell}$ := hardmax($Q_{h,\ell}(X)K_{h,\ell}(X)^\top$)$V_{h,\ell}(X)$，其中hardmax按行应用。已知hardmax 注意力可以用 softmax 注意力 [SHT24b], 来近似，因此考虑hardmax 注意力本质上不失一般性。事实上，通过将$Q_{h,\ell}$ 与$K_{h,\ell}$ 的元素按量级为 $O$(log $NHL$)的适当因子进行缩放，我们可以确保每个注意力头输出$Y_{h,\ell}$ ，使得

$$
\left\| Y_{h,\ell} - \mathrm{hardmax}(Q_{h,\ell}(X)K_{h,\ell}(X)^\top)V_{h,\ell}(X) \right\|_\infty \ll \frac{1}{\mathrm{poly}(NHL)}. \tag{1}
$$

然后，将 $Z_{h,\ell}$ 记为 hardmax($Q_{h,\ell}(X)K_{h,\ell}(X)^\top$)$V_{h,\ell}(X)$ 的最后一列，我们断言 $Z_{h,\ell}$ 是一个向量：若 $a_i$ 和 $c_{h,\ell}$ 不属于任何正交三元组（即不存在 $b_j$ 使得 $a_i, b_j, c_{h,\ell}$ 彼此正交），则其第 $i$ 个元素为 2；若 $a_i$,$c_{h,\ell}$参与了一个正交三元组，则其值至多为 3 2。这是因为，在前一种情况下，第 $i$ 行中最大值对应的索引集合只有 $N+1$ ，其中 $V_{h,\ell}(X)$ 包含 2；在后一种情况下，最大值对应的索引集合至少还包含另一个元素，因此 hardmax 会对至少一个 $V$ 包含 1 的行取平均。对所有 $h, \ell$, $i$求和后，若不存在正交三元组，则有$\sum_{h,\ell}\sum_i Z_{h,\ell}[i] = 2NHL$；若存在，则至多为 $2NHL - \frac{1}{2}$。根据 (1)，我们可以利用 $\sum_{h,\ell}\sum_i Y_{h,\ell}[i, m]$ 来判断 $A, B, C$是否包含正交三元组。

**大嵌入维度**。 现在，我们将介绍针对计算大嵌入维度 Transformer（$m = N$）的扩展算术电路的下界。回顾一下，计算单个注意力头需要 $N^{\omega+o(1)}$ 时间，因为我们需要进行两次矩阵乘积：一次用于计算 $QK^\top$，另一次用于计算 Softmax($QK^\top$) 与 $V$ 的乘积。由此很容易看出，我们可以使用 $O(LH)$ 个矩阵乘法实例来计算具有 $L$ 层、 $H$ 个注意力头和嵌入维度为 $N$ 的 Transformer。

对应的下界要困难得多。虽然证明可以将矩阵乘法归约为一个嵌入维度为$N$ 的注意力头并不太难（例如，可以设置 $Q \leftarrow A$、 $K \leftarrow B^\top$ 和 $V \leftarrow I$ ，使注意力头输出

softmax$(QK^\top)$。我们可以使用简单的去归一化技巧来获得$\exp(QK^\top)$（见第A节），进而计算 $AB^T$；但如何用单个 Transformer 同时计算多个相互独立的矩阵乘法实例，并不明确。以一个只有一层、具有 $H$ 个注意力头的 Transformer 为例。该 Transformer 的输出是这一层所有注意力头输出的总和。如前所述，计算矩阵乘积之和比逐个计算每个矩阵乘积更高效（例如， $N$个矩阵乘积之和可以在 $O\left(N^{3.251}\right)$ 时间内计算，而分别计算每个乘积则需要 $O\left(N^{3.372}\right)$时间。因此，如果我们直接将单个矩阵乘积归约为单个注意力头的方法应用于 Transformer 中的所有 $LH$ 个注意力头，那么 Transformer 的输出只能计算 $LH$个注意力头输出的总和，其维度为 $\Theta(N^2)$，而表达 $\Theta(LHN^2)$ 维度所需的 $LH$ 个不同矩阵乘积则需要更高维度。

我们所需的关键工具是 Baur-Strassen定理 [BS83], ，它最初是为在算术电路模型中证明下界而设计的。Baur-Strassen定理最著名的应用或许是反向传播算法，它是高效训练神经网络的基础构件 [RHW86]。Baur-Strassen定理也应用于算法设计 [CGS15、Fis24]。我们提出 Baur-Strassen定理的一种新应用：为 Transformer 推导下界。Baur-Strassen定理指出，给定一个规模为 $s$ 、计算函数 $f:\mathbb{R}^n\to\mathbb{R}$ 的算术电路，就存在一个规模为 $O(s)$ 的算术电路，能够计算 $f$ 关于其 $n$ 个输入的所有偏导数。特别地，Baur-Strassen定理为我们提供了一种强大的工具，可以从算法中的部分计算中提取更多信息，从而为获得所需的 $\Omega(LHN^2)$ 维输出提供了途径。

在标准设定中，算术电路（定义 2.1）由 $+$、$-$、$\times$、$/$ 个门组成，电路的规模是门的总数。如前所述，由于注意力机制中的关键非线性是需要进行幂运算的 Softmax 函数，因此我们为算术电路配备 指数函数 和 自然对数函数 门，并称其为 扩展算术电路。我们的第一步是扩展 Baur-Strassen定理：如果存在一个规模为 $s$ 、计算函数 $f:\mathbb{R}^n\to\mathbb{R}$ 的扩展算术电路，那么就存在一个规模为 $O(s)$ 的扩展算术电路，能够计算 $f$的所有偏导数。

有了这一工具，我们现在来描述下界构造。为简单起见，我们用所谓的去归一化注意力头来描述该构造；它用逐元素exp替代按行进行的 Softmax。具体而言，给定 $Q(X)$ $K(X)$ $V(X)$，一个去归一化注意力头的输出为 $\exp(Q(X)K(X)^\top)V(X)$（关于去归一化注意力头的更详细讨论，见附录 A）。在定理 1.2 的证明中（见定理 5.1），我们会修改下文给出的构造，以确保能够从标准注意力头（带归一化）中提取矩阵乘积。借助去归一化注意力头，我们可以构造一个 Transformer $T$，其输出为向量 $T(X)\in\mathbb{R}^N$，其中各元素为 $T(X)_i=\sum_{k=1}^{LH}\sum_{j=1}^{N}\exp((A_kB_k^\top)_{ij})$，其中 $A_k$、$B_k\in\mathbb{R}^{N\times m}$ 是任意的 $N\times m$ 矩阵，适用于 $k\in[LH]$。我们的 Transformer 每层仅由 $H$ 个去归一化注意力头组成，其中第 $\ell$层的第 $h$个注意力头设置$Q_h$、$\ell(X)\leftarrow A_{(\ell-1)H+h}$、 $K_h$、$\ell(X)\leftarrow B_{(\ell-1)H+h}$，并且$V_h$、$\ell(X)\leftarrow 1_{N\times 1}$ 是全 1 向量。（严格来说，$V_h$、$\ell(X)$ 应该是一个 $N\times m$ 矩阵，但这里我们不需要其余的列，因此可以直接用 0 补齐。）于是，$\exp(Q_h$、$\ell(X)K_h$、$\ell(X)^\top)V_h$、$\ell(X)=\exp(A_{(\ell-1)H+h}B_{(\ell-1)H+h}^\top)\cdot 1_{N\times 1}$ 是由 $\exp(A_{(\ell-1)H+h}B_{(\ell-1)H+h}^\top)$ 的行和构成的向量。最后，我们将每层所有注意力头的输出相加，并在所有 $L$ 层上进行归纳。

假设存在一个规模为 $s$、用于计算 Transformer$T$ 的扩展算术电路（eAC）。需要指出的是，尽管我们的 Transformer $T$ 具有固定的嵌入和多层感知机映射，因此可以由固定电路计算，但嵌入和多层感知机映射的参数最多可包含 $O(m)$ 个权重，并将这些权重作为额外输入提供给电路。有关我们模型的更多详细信息，请参见第 2.2 节。

关于我们模型的讨论。给定上述 Transformer，我们现在的目标是应用扩展 Baur-Strassen 定理来提取有用的导数。我们先对输入稍作修改。引入辅助输入变量 $C_{kij}$，用于表示 $k \in [LH]$ 和 $i$、$j \in [N]$，我们将 $C_k$ 定义为一个元素为 $C_k[i,j] = C_{kij}$ 的 $N \times N$矩阵。然后，对于所有 $k \in [LH]$ 矩阵 $\tilde{A}_k$，$\tilde{B}_k \in \mathbb{R}^{N \times 2N}$，我们定义

$$\tilde{A}_k = \begin{pmatrix} A_k & C_k \end{pmatrix}, \quad \tilde{B}_k = \begin{pmatrix} B & I \end{pmatrix}.$$

首先，我们观察到 $\tilde{A}_k$，$\tilde{B}_k$ 可以在 $O\left(N^2\right)$ 时间内高效计算。

现在，我们描述如何使用 Transformer $T$ 计算 $A_k B_k^\top$ 的矩阵乘积 $k \in [LH]$。运行 Transformer $T$ 并对输出向量的所有条目求和，我们得到

$$F \leftarrow \sum_{k=1}^{LH} \sum_{i,j=1}^{N} \exp\left( \left( \tilde{A}_k \tilde{B}_k^\top \right)_{ij} \right) = \sum_{k=1}^{LH} \sum_{i,j=1}^{N} \exp\left( \left( A_k B_k^\top \right)_{ij} + C_{kij} \right).$$

现在，考虑 $F$ 关于 $C_{kij}$的导数。

$$\begin{aligned} \frac{\partial F}{\partial C_{kij}} &= \frac{\partial}{\partial C_{kij}} \exp\left( \left( A_k B_k^\top \right)_{ij} + C_{kij} \right) \\ &= \exp\left( \left( A_k B_k^\top \right)_{ij} + C_{kij} \right). \end{aligned}$$

现在，如果我们对所有 $k \in [LH]$ 和 $i$设置 $C_{kij} \leftarrow 0$，$j \in [N]$，

$$\frac{\partial F}{\partial C_{kij}} = \exp\left( \left( A_k B_k^\top \right)_{ij} \right).$$

因此，通过应用扩展 Baur-Strassen 定理，我们得到 $\exp\left(A_k B_k^\top\right)$的各个条目。在此基础上，应用 $O\left(LHN^2\right)$ 个 ln门，我们得到所需的 $LH$ 个矩阵乘积。特别地，我们得到一个规模为 $O\left(s + LHN^2\right)$ 的 eAC，它可以计算 $LH$ 个独立矩阵乘积。

我们所需的最后一个要素，是任何计算 $LH$个独立矩阵乘积的eAC在规模上的下界。众所周知，标准算术电路需要$LHN^{\omega - o(1)}$ 的规模才能计算 $LH$ 个独立矩阵乘积（见引理4.5），而扩展算术电路的下界则更为微妙。乍看之下，扩展算术电路可能比标准算术电路强大得多，并且可能能够用少于 $N^\omega$ 个门来计算矩阵乘积。我们证明，当限制于低次函数时，任何规模为 $s$的扩展算术电路都可以由不含指数函数与自然对数函数门、规模为 $O(s)$的标准算术电路来模拟。也就是说，扩展算术电路在计算低次函数时并不具有优势。我们的结果推广了Strassen [Str73]的结果（阐述见 [Blä13]），该结果指出除法门对于计算低次函数并无用处，并且可能具有独立的意义。

## 1.3 进一步的相关工作

我们回顾 Transformer 理论研究中的两条脉络：(1) Transformer 的表达能力，即 Transformer 能计算和学习哪些函数；(2) Transformer 模型的复杂度，即计算 Transformer（或 Transformer 的近似）有多高效?

**Transformer 的表达能力。** 关于 Transformer 表达能力的研究由来已久。若干研究由 [HHG+20] 开创，[YPPN21], 证明了 Transformer 相比循环架构，在层次结构解析 [YPPN21], 稀疏平均 [SHT23], 上下文学习 [WDL25], 复制 [JBKM24], 以及文档相似度 [AY25]等任务中的优势。

尽管 Transformer 具有强大的表达能力，但针对 Transformer 的下界表明，许多函数无法由小型 Transformer 计算。针对 Transformer 的下界通常依赖于通信复杂度 [SHT23、PNP24、SHT24b]以证明深度为一的下界，或归约到其他计算模型 [MS23、PNP24、SHT24b]。值得注意的是，[SHT24b] 通过证明在特定参数下 Transformer 与 MPC 协议之间的等价性，将 Transformer 模型与大规模并行计算模型联系起来，从而能够˜利用在 MPC 中具有困难性的问题给出 Transformer 的下界。此外，$O(1)$深度的 Transformer 存在一个困难函数 [CPW25]。 [Hah20] 证明了 hardmax Transformer 无法计算奇偶校验或 Dyck 语言。

**Transformer 计算的算法与困难性。**注意力的精确计算已得到较为充分的理解，在小嵌入情形下已经建立了二次下界 [KWH23] ，并推广到了更一般的设置 [AS24、AY25、GHS+26]。鉴于这些下界，研究人员在小数值项[AS24]和常数嵌入维度 [GHS+26]等特殊设置下开发了次二次算法。

鉴于精确注意力计算的困难性，已有大量工作研究注意力的高效近似，包括 [BMR+20]、Reformer [KKL20], [KVPF20], Big Bird [ZGD+20a],Scatterbrain [CDW+21]、Performer [CLD+21]、KDEFormer [ZHDK23], HyperAttention [HJK+24] 以及 Polysketchformer [KMZ24]。最近，[YA25] 提供了一个新方向，即用较小输入长度的 Transformer 来计算较大输入长度的 Transformer。

提高 Transformer 性能的实际尝试通常转向硬件优化，包括 FlashAttention 等优化输入/输出复杂度的技术 [DFE+22，FEN+23，SY24] 以及利用并行性的推测解码 [LKM23]。

## 2 预备知识

**记号。** 令 $[n]$ 表示集合 $\{1,\ldots,n\}$。令 $\mathbb{R}_{\geq 0}$ 表示非负实数集，$\mathbb{R}_{>0}$ 表示正实数集。 $\mathbb{R}_{\leq 0}$, $\mathbb{R}_{<0}$ 的定义类似。除非另有说明，我们的对数函数 log 是以 2 为底的对数。

对于环 $R$，令 $R[x_1,\ldots,x_n]$ 表示以 $n$ 为变量、系数属于 $R$的多项式集合，并令 $R[[z]] = \{\sum_{n=0}^{\infty} a_n z^n | a_n \in R\}$ 表示以 $z$ 为变量、系数属于 $R$的形式幂级数集合。

令 $e_i$ 表示第 $i$ 个基向量，即除第 $i$ 个坐标为 1 外，其余坐标均为 0。令 $I_n$ 表示维度为 $n$ 的单位矩阵。对于任意矩阵 $A$，用 $A_{ij}$ 或 $A[i,j]$ 表示第 $(i,j)$ 个元素；如果矩阵名称本身带有下标，我们通常使用后者。令 $A[i;\ ]$ 表示 $A$ 的第 $i$ 行，$A[;\ j]$ 表示 $A$ 的第 $j$ 列。对于向量 $v \in \mathbb{R}^n$，令 $\mathrm{diag}(v)$ 表示对角矩阵，其元素为 $\mathrm{diag}(v)_{ii} = v_i$。对于矩阵 $A$，令 $A^\top$ 表示其转置，$A^{-1}$ 表示其逆元。对于任意函数 $f$，令 $f(A)$ 表示将 $f$ 应用于 $A$ 的每个元素后得到的矩阵。类似地，令 $A \leq B$ 表示对于 $A$ 的每个元素，$A_{ij} \leq B_{ij}$ $B$。令 $A \odot B$ 表示两个矩阵之间的逐元素乘积。令 $0_{a\times b}$、$1_{a\times b}$ 分别表示全为零和全为一的 $a \times b$ 矩阵。令 $I_n$ 表示 $n \times n$ 单位矩阵。

softmax算子定义为

$$\mathrm{softmax}(v) = \frac{1}{\sum_{j=1}^{N}\exp(v_j)} \cdot (\exp(v_1), \ldots, \exp(v_N)) \in \mathbb{R}^N$$

对于 $v \in \mathbb{R}^N$。我们还将 hardmax 算子定义为

$$\mathrm{hardmax}(v)[i] = \frac{\mathbf{1}\left[i \in I_{\max}(v)\right]}{|I_{\max}(v)|}$$

其中 $I_{\max}(v) = \{j : v_j = \max v\}$。给定一个矩阵 $A$，我们滥用记法，用 softmax($A$) 和 hardmax($A$) 分别表示对 $A$ 的每一行应用 softmax 和 hardmax($A$) 后得到的矩阵。

## 2.1 算术电路

**定义 2.1**。算术电路（AC）是一个操作序列 $\{g_1, \ldots, g_s\}$，满足每个$g_j \leftarrow y \circ z$对于 $\circ \in \{+, -, \times, /\}$ 和 $y,\ z \in \{g_i\}_{i<j} \cup \{x_i\} \cup \mathbb{R}$，其中门的一个子集 $\{g_{i_k}\}$ 称为输出门。给定输入 $x$，将输出门上的值记为 $\mathcal{C}(x)$。算术电路$\mathcal{C}$ 计算 $F : \mathbb{R}^n \to \mathbb{R}^m$，如果对于所有 $x \in \mathbb{R}^n$，都有 $\mathcal{C}(x) = F(x)$。

矩阵乘法指数 $\omega$ 定义为最小常数，使得对任意 $\varepsilon > 0$，两个 $n \times n$ 矩阵的乘积都可由规模为 $O(n^{\omega+\varepsilon})$的算术电路计算。

我们还将考虑一类更一般的电路，其中包含指数门和对数门。

**定义2.2.** 计算 $F : \mathbb{R}^n \to \mathbb{R}$的扩展算术电路 （eAC）是一个运算序列 $\{g_1, \ldots, g_s\}$，使得每个$g_j$ 为下列形式之一：

1. $g_j \leftarrow y \circ z$ for $\circ \in \{+, -, \times, /\}$ and $y, z \in \{g_i\}_{i<j} \cup \{x_i\} \cup \mathbb{R}$.
2. $g_j \leftarrow \exp(y)$ for $y \in \{g_i\}_{i<j} \cup \{x_i\} \cup \mathbb{R}$.
3. $g_j \leftarrow \ln(y)$ for $y \in \{g_i\}_{i<j} \cup \{x_i\} \cup \mathbb{R}_{>0}$.

## 2.2 Transformer

我们首先定义自注意力头，即Transformer的核心原语。令 $N$ 表示上下文（输入）长度，$m$ 表示嵌入（隐藏）维度，$d_{\text{in}}$ 表示输入维度。按照实际取值，我们假设 $m = O(N)$。对于输入 $X \in \mathbb{R}^{N \times d_{\text{in}}}$，我们将 $X$ 的每一行称为一个词元。

**定义2.3**（注意力）. 一个自注意力头 是由下式定义的映射$f_{Q,K,V} : \mathbb{R}^{N \times d_{\text{in}}} \to \mathbb{R}^{N \times m}$

$$f_{Q,K,V}(X) = \mathrm{softmax}(Q(X)K(X)^\top)V(X)$$

并由按行参数化的查询、键和值嵌入 $Q$、$K$、$V$：$\mathbb{R}^{N \times d_{\text{in}}} \to \mathbb{R}^{N \times m}$（例如，$Q(X) = (Q(X_1)\ldots\ldots, Q(X_N))$）。令 $\mathrm{Att}^N_{m,\ d_{\text{in}}}$表示所有输入维度为 $d_{\text{in}}$、嵌入维度为 $m$ 且输入长度为 $N$的自注意力头的集合。当 $m = d_{\text{in}}$ 时，我们将其记为 $\mathrm{Att}^N_m$。

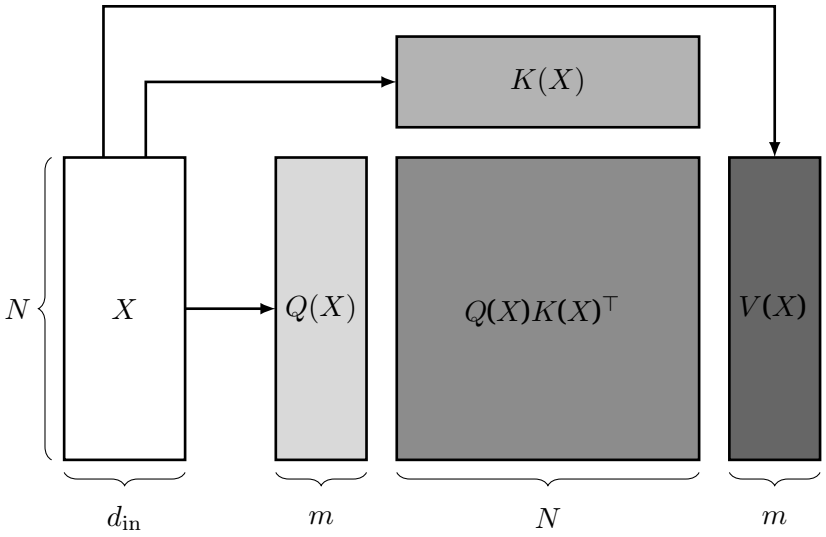


图1：输入为 $X \in \mathbb{R}^{N\times d_{\text{in}}}$、输入长度为 $N$ 、嵌入维度为 $m$ 的注意力头。 $Q(X)$ $K(X)$ $V(X)$ 通过对 $X$ 逐行应用嵌入映射得到。然后，通过 $\text{softmax}(Q(X)K(X)^\top)V(X)$ 计算输出。

图1 展示了一个注意力头。在实践中，以及在我们的构造中，嵌入函数通常采用以下形式： $Q(X) = XW_Q$（类似地， $K(X) = XW_K$， $V(X) = XW_V$），其中的权重矩阵为 $W_Q$、$W_K$、$W_V \in \mathbb{R}^{d_{\text{in}}\times m}$。

Hardmax 注意力在文献中也得到了广泛研究 [MSS22、SHT24b、JSLC25]，尤其是在理论研究中。对于每个查询，Hardmax 注意力只允许其关注能够取得最大内积（最高相关性）的键。

**定义 2.4** （Hardmax 注意力）：A 个*hardmax*注意力头 是一个映射 $f^{\text{hard}}_{Q,K,V}: \mathbb{R}^{N\times d_{\text{in}}} \to \mathbb{R}^{N\times m}$，定义为

$$f^{\text{hard}}_{Q,K,V}(X) = \text{hardmax}(Q(X)K(X)^\top)V(X).$$

一般来说，除非另有说明，我们使用“（自）注意力”来指代 softmax 注意力（定义2.3）。最后，我们定义多层感知机 （MLP），它是一种浅层神经网络。通用逼近定理指出，任何连续函数都可以由一个 MLP（具有足够多的神经元）近似，因此我们将 MLP 建模为定义在向量上的连续函数。

**Defi义 2.5.** 多层感知机是一个连续函数 $\Psi: \mathbb{R}^{m_1} \to \mathbb{R}^{m_2}$，对于 $m_1, m_2 > 0$.

给定一个矩阵 $A$，我们滥用符号，将 $\Psi(A)$ 记为对矩阵 $A$逐行应用 $\Psi$ 的结果。现在我们正式定义 Transformer。

**定义 2.6** （Transformer）。一个*Transformer* 是一个映射 $T: \mathbb{R}^{N\times d_{\text{in}}} \to \mathbb{R}^{N\times d_{\text{out}}}$，由注意力头 $\{f_{\ell,h}\}^{L,H}_{\ell,h=1}$ 以及逐元素多层感知机层 $\Psi_\ell: \mathbb{R}^m \to \mathbb{R}^m$ （对于 $\ell \in [L]$）指定。输入 $X \in \mathbb{R}^{N\times d_{\text{in}}}$后，Transformer 计算中间嵌入 $X^{(1)}, \ldots, X^{(L)}$，其中

$$X^{(\ell)} = \Psi_\ell\left(X^{(\ell-1)} + \sum_{h=1}^{H} f_{\ell,h}\left(X^{(\ell-1)}\right)\right) \quad \forall \ell \in [L],$$

并为输出多层感知机 $\Psi_O$ 返回 $T(X) = \Psi_O(X^{(L)})$: $\mathbb{R}^m \to \mathbb{R}^{d_{\text{out}}}$ （为方便记号，令 $X^{(0)} := X$ ）。

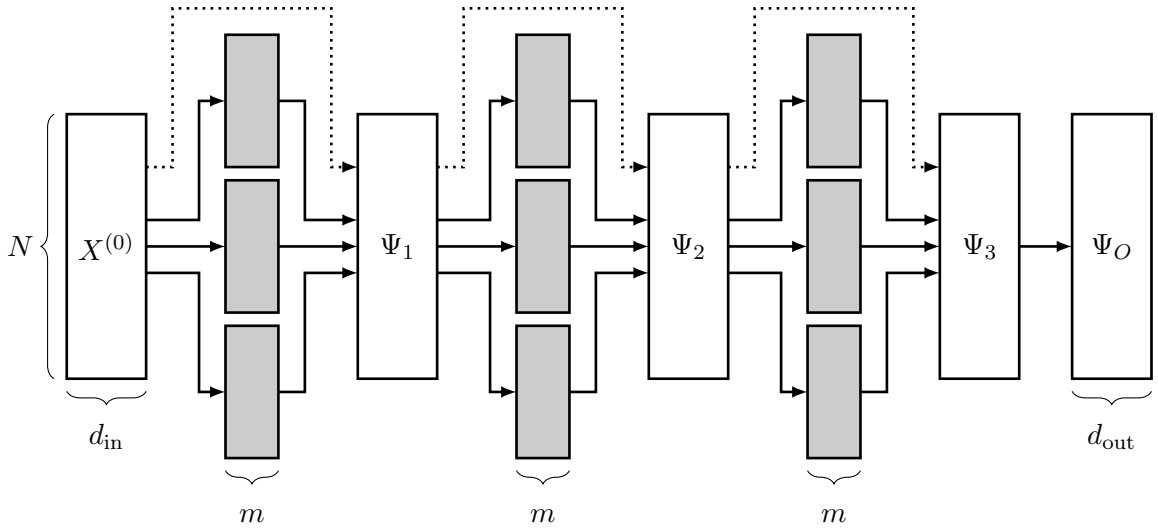


图 2：一个包含 $H = 3$ 和 $L = 3$的 Transformer。注意力头以浅灰色表示，多层感知机以 $\Psi$表示。残差连接以虚线表示，此外还将 $X^{(\ell-1)}$加到 $X^{(\ell)}$上。

类似地定义 *hardmax Transformer* ，将每个注意力头替换为 hardmax 注意力头。

请注意，在该定义中，我们将每个注意力头的结果相加，这种做法在许多理论研究中都有采用（例如 [SHT23、SHT24b、SHT24a]）。实践中使用的更标准版本则会将注意力头的结果进行拼接。此外，在求和聚合中，注意力头维度等于嵌入维度 $m$，而在拼接聚合中，注意力头维度等于 $m/H$。由于我们可以使用线性多层感知机从拼接结果中轻松获得求和结果，因此，我们所有的下界都可以调整到拼接版本。详见第 C节。

本文在证明困难性结果时，重点研究注意力机制，并给出不含多层感知机的 Transformer 的下界。在第 B节中，我们说明这些结果同样适用于实践中常见的包含标准多层感知机的 Transformer，即使所有多层感知机的计算都需要线性时间；也就是说，对于给定输入 $x \in \mathbb{R}^m$，计算 $\Psi(x)$ 需要 $O(m)$ 时间来计算多层感知机 $\Psi$。图 2 展示了一个包含3 个注意力头和 3层的 Transformer 示例。

最后，我们现在可以定义算法计算 Transformer 的含义了 s.

**定义 2.7**。 如果对于任意一个具有上下文长度 $N$、嵌入维度$m$、 $L$层、每层有 $H$ 个注意力头、输入维度为 $d_{\text{in}}$ 且输出维度为 $d_{\text{out}}$ 的 Transformer $T$，以及任意输入 $X \in \mathbb{R}^{N \times d_{\text{in}}}$，算法的输出为 $T(X)$，那么我们就称该算法能够计算具有上述参数的 Transformer。

对于电路，定义要稍微复杂一些。如果我们试图说电路 $C$ 计算一个 Transformer，那么该电路必须能够计算所有可能的嵌入映射和多层感知机映射；这就要求电路同时编码许多函数，条件过于严格。因此，我们的目标是针对满足弱得多的要求的电路给出下界：我们证明了针对计算固定 Transformer 的电路 $C$ 的下界。

尽管我们研究的是固定Transformer，但需要指出，嵌入映射可以是参数化的。在实际中，这对应于模型的学习权重。形式上，考虑一个注意力头 $f_{\ell,\ h}$ 其嵌入映射为 $Q_{h,\ \ell}$，$K_{h,\ \ell}$，$V_{h,\ \ell}$。然后，我们要求 $(Q_{h,\ \ell})_i = g(w_{Q_{h,\ \ell}},\ X_i)$，其中

$g$ 是一个可由 eAC 计算的函数，$w_{Q_{h,\ell}}$ 表示与该映射相关联的权重，而 $X_i$表示注意力头输入的第 $i$ 行。在我们的构造中，我们要求 $g$ 是一个规模为 $O(m)$ 的 eAC。令 $W$ 表示 Transformer 所有权重的拼接；对于任意 eAC Transformer $T$，令 $T_W(X)$ 表示 $T$在权重为 $W$、输入为 $X$时的计算。最后，我们定义电路计算 Transformer 的含义。

**定义 2.8**。 设 $T$ 是一个 Transformer，其上下文长度为 $N$，嵌入维度为 $m$，有 $L$ 层，且每层有 $H$ 个注意力头，输入维度为 $d_{\text{in}}$，输出维度为 $d_{\text{out}}$。假设 $T$ 具有权重$W$，输入为 $X$。若对于任意输入 $X \in \mathbb{R}^{N\times d_{\text{in}}}$ 和任意权重 $W$，电路 $\mathcal{C}$ 都满足 $\mathcal{C}(X, W) = T_W(X)$，则称该电路计算 $T$。

为了针对计算 Transformer 的小型 eAC 获得有意义的下界，我们证明 Transformer 可以计算扩展算术电路难以计算的函数。当然，要使下界有意义，我们指出 Transformer 本身也必须能够由 eAC 计算。

**Softmax 模拟 Hardmax**。 我们为 Transformer 计算引入一种实用的基本操作，并证明当最大值与次大值之间有足够大的间隔时，softmax 注意力可以模拟 hardmax 注意力。这与 [SHT24b] 的结果类似。

**引理 2.9**。 设 $f \in \text{Att}_m^N$是一个自注意力单元，其嵌入函数为 $Q$、$K$、$V$，满足

1. $V(X) \in \{0,1,2\}^{n\times m}$，且
2. 对每个 $X \in \mathbb{R}^{N\times m}$ 和 $i \in [N]$:

$$A(X)_{i,i'} \le \max_{i''} A(X)_{i,i''} - 1,\ \forall i' \notin I_{\max}(A(X)_i),$$

*where* $A(X) = Q(X)K(X)^\top$.

则存在一个注意力单元 $f' \in \text{Att}_m^N$，满足

$$\left\| f'(X) - \text{hardmax}(A(X))V(X) \right\|_\infty \le \frac{1}{\text{poly}(N)}$$

对于任意小的逆多项式。

该模拟背后的思路很简单：通过将 $Q$ 中的每个查询向量缩放一个足够大的常数 $c = \Theta(\log N)$，我们确保 softmax 运算分配给非最大值索引的任何概率相对于 $c$都呈指数级小，即小于 $\frac{1}{\text{poly}(N)}$，其中 A为任意多项式。证明留待第 A节给出。

## 2.3 细粒度复杂性

本节阐述我们在细粒度复杂性理论中的主要假设。有关这些假设应用的更多讨论，请参见第1节。

**假设 2.10** （强指数时间假设（强指数时间假设））。 对于任意 $\varepsilon > 0$，存在正整数 $k$，使得 $k$-SAT 需要 $\Omega(2^{(1-\varepsilon)n})$ 时间，其中 $n$ 是合取范式中的变量数。

我们称 $k$ 个向量 $v_1, \ldots, v_k \in \{0,1\}^d$ 是正交的，如果对所有 $j \in [d]$，$v_\ell[j] = 0$ 对某个 $\ell \in [k]$成立。换言之，它们的 $k$重内积为 0。于是 $k$-OV 问题定义如下：给定 $A_1, \ldots, A_k \subset \{0,1\}^d$，每个的规模均为 $n$，判定是否存在 $v_1 \in A_1, \ldots, v_k \in A_k$，使得 $v_1, \ldots, v_k$ 是正交的。检查所有向量 $k$ 元组的平凡算法需要$O(n^k d)$时间。 $k$-OV假设指出，对于 $d = \Theta(\log n)$，这本质上是最优的：即任何算法都需要 $n^{k-o(1)}$ 时间。

**假设2.11** ($k$-OV 假设)。 在由 $n$ 个向量组成、维度为 $\Theta(\log n)$的集合上计算 $k$-OV 需要 $n^{k-o(1)}$ 时间。

$k$-OV 假设可由广为接受的强指数时间假设（强指数时间假设） [IP01]推出，该假设指出，可满足性在 $n$ 个变量上需要 $2^{(1-o(1))n}$ 时间 [Wil04]。我们的下界基于3-OV 假设，即 $k$-OV 对于 $k = 3$的一个特例。

不平衡 $k$-OV问题定义如下：给定 $k$ 个向量集合 $A_1, \ldots, A_k$，其规模为$S_1, \ldots, S_k$，判定是否存在 $v_1 \in A_1, \ldots, v_k \in A_k$，使得 $v_1, \ldots, v_k$ 彼此正交。检查所有 $k$元组的平凡算法需要 $O(S_1 S_2 \ldots S_k d)$时间。同样，不平衡$k$-OV假设指出这本质上是最优的，并且众所周知，不平衡$k$-OV假设等价于 $k$-OV假设。

**假设 2.12** （不平衡 $k$-OV 假设）。 $k$-OV 在规模分别为 $n^{s_1}, \ldots, n^{s_k}$、维度为 $d = \Theta(\log n)$ 的集合 $S_1, \ldots, S_k$ 上需要 $n^{s_1+\cdots+s_k-o(1)}$ 时间。

显然，通过令 $s_1 = \cdots = s_k = 1$。下述引理给出了反向结论，为完整起见，我们在第 A节中给出证明。

**引理 2.13**。 设 $0 < s_2 \ldots, s_k \leq 1$。对 $d = \Theta(\log n)$，设 $A_1, \ldots A_k \subset \{0,1\}^d$，其中 $|A_1| = n$，且对所有 $i > 1$，均有 $|A_i| = n^{s_i}$。那么，在 $A_1, \ldots, A_k$ 上计算 $k$-OV 的任何算法，在 $k$-OV 假设下都需要 $n^{s_1+\ldots+s_k-o(1)}$ 时间。特别地，在 $k$-OV 假设下，不平衡 $k$-OV 假设成立。

# 3 小嵌入维度的下界

我们首先讨论小嵌入维度情形（即 $m = N^{o(1)}$）下的下界。我们的下界基于 $k$-OV假设（或强指数时间假设）。

首先，我们证明，在 $k$-OV假设下，当嵌入维度较小（例如 $m = \Theta(\log n)$）时，不存在能够比朴素算法显著更高效地计算 Transformer 的算法；朴素算法会分别计算每个注意力头。

**定理 3.1** (正式定理 1.1). 设 $m = \Theta(\log N)$ 且 $L, H = \text{poly}(N)$。任何计算输入长度为 $N$、嵌入维度为 $m$、$L$ 层、每层有 $H$ 个注意力头且输入和输出维度为 $m$ 的 *Transformer*，并达到逐项加性误差 $\frac{1}{10N}$，在 $LHN^{2-o(1)}$时间内，需要满足3-OV 假设。即使 *Transformer* 不含多层感知机，该下界仍然成立。

在上述陈述中，我们要求 $L$，$H = \text{poly}(N)$，因为在 3-OV 中，三个集合的规模必须是多项式相关的。我们指出，这是该问题有趣的情形，因为当 $L$，$H = N^{o(1)}$ 时，我们可以直接根据单个注意力头的困难性得到最优下界。

**引理 3.2**。假设存在一个算法 $\mathcal{A}$，能够在时间 $B$ 内，以逐项加性误差 $\frac{1}{10N}$ 计算任意输入长度为 $N+1$、嵌入维度为 $2d+2$、包含 $L$ 层且每层有 $H$ 个注意力头、输入和输出维度为 $2d+2$ 的 *Transformer*，那么就存在一个算法，能够在时间 $O(LHd+B)$ 内求解维度为 $d$、规模分别为 $N$、$N$、$LH$ 的非平衡 3-OV~。

首先，我们注意到定理 3.1可由引理 3.2推出。

假设引理 3.2，证明定理 3.1。反设存在一种算法，对于某个 $\varepsilon > 0$，可以在时间 $O\left(LHN^{2-\varepsilon}\right)$ 内计算任意具有输入长度 $N+1$、嵌入维度 $m = 2d+2$、$L$ 层（每层有 $H$ 个注意力头）、输入和输出维度 $m$ 的 Transformer，且逐项加性误差不超过 $\frac{1}{10N}$。现在考虑一个维度为 $d = \Theta(\log N)$、规模为 $N$、$N$、$LH$ 的非平衡 3-OV 实例。引理 3.2 表明，该问题存在一个 $\tilde{O}(LHd + LHN^{2-\varepsilon})$ 算法，这与 3-OV 假设矛盾。 □

现在我们证明引理3.2 以完成定理 3.1的证明。

引理的证明 *3.2*。考虑一个 3-OV 实例 $(A, B, C)$ 其中 $|A| = |B| = N$ 且 $|C| = LH$。我们构造如下 Transformer $T$，使得 $m = d_{\text{in}} = d_{\text{out}} = 2d+2$。假设我们有一个算法，能够计算 $T$，且误差至多为逐项加性误差 $\frac{1}{10N}$。令输入 $X \in \mathbb{R}^{(N+1)\times d_{\text{in}}}$ 为——

$$X = \begin{pmatrix} a_1^\top & b_1^\top & 1 & 0 \\ a_2^\top & b_2^\top & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ a_N^\top & b_N^\top & 1 & 0 \\ 0^\top & 0^\top & 2 & 0 \end{pmatrix},$$

其中 $a_i$，$b_i$ 是向量，属于 $A$，$B$。此外，我们对 $C = \{c_{h,\ell}$ 进行如下编号：$1 \le h \le H$，$1 \le \ell \le L\}$ 并定义 e

$$Q_{h,\ell}(X) = -\begin{pmatrix} a_1^\top & 0_{1\times(d+2)} \\ a_2^\top & 0_{1\times(d+2)} \\ \vdots & \vdots \\ a_N^\top & 0_{1\times(d+2)} \\ 0_{1\times d} & 0_{1\times(d+2)} \end{pmatrix}, K_{h,\ell}(X) = \begin{pmatrix} b_1^\top \odot c_{h,\ell}^\top & 0_{1\times(d+2)} \\ b_2^\top \odot c_{h,\ell}^\top & 0_{1\times(d+2)} \\ \vdots & \vdots \\ b_N^\top \odot c_{h,\ell}^\top & 0_{1\times(d+2)} \\ 0_{1\times d} & 0_{1\times(d+2)} \end{pmatrix}, V_{h,\ell}(X) = \begin{pmatrix} 0_{1\times(2d+1)} & 1 \\ \vdots & \vdots \\ 0_{1\times(2d+1)} & 1 \\ 0_{1\times(2d+1)} & 2 \end{pmatrix}$$

使得 $Q_{h,\ell}(X) \in \mathbb{R}^{(N+1)\times m}$，$K_{h,\ell}(X) \in \mathbb{R}^{(N+1)\times m}$，$V_{h,\ell} \in \mathbb{R}^{(N+1)\times m}$ 其中 $b \odot c \in \mathbb{R}^d$ 是两个向量 $a$，$b$的逐元素乘积。

我们声称，构造该 Transformer 需要 $O(dHL)$ 时间，因为每个注意力头都可以在 $O(d)$ 时间内构造（即嵌入映射可以在 $O(d)$ 时间内指定）。此外，我们可以按需对 $Q_{h,\ell}(X)K_{h,\ell}(X)$ 进行编码。特别地，$Q_{h,\ell}$，$K_{h,\ell}$：$\mathbb{R}^m \to \mathbb{R}^m$ 对每一行应用以下映射：

$$\begin{aligned} Q_{h,\ell} &: v \mapsto -(v_1, \ldots, v_d, 0, \ldots, 0), \\ K_{h,\ell} &: v \mapsto (v_{d+1} \cdot c_{h,\ell}[1], \ldots, v_{2d} \cdot c_{h,\ell}[d], 0, \ldots, 0), \\ V_{h,\ell} &: v \mapsto (0, \ldots, 0, 0, v_{2d+1}). \end{aligned}$$

注意，上述映射可以用 $\mathbb{R}^{m\times m}$ 中的权重矩阵表示：

$$W_{Q,h,\ell}=\begin{pmatrix}-I_{d\times d} & 0_{d\times(d+2)}\\ 0_{(d+2)\times d} & 0_{(d+2)\times(d+2)}\end{pmatrix},$$
$$W_{K,h,\ell}=\begin{pmatrix}0_{d\times d} & 0_{d\times(d+2)}\\ \mathrm{diag}(c_{h,\ell}) & 0_{d\times(d+2)}\\ 0_{2\times d} & 0_{2\times(d+2)}\end{pmatrix},$$
$$W_{V,h,\ell}=\begin{pmatrix}0_{2d\times(2d+1)} & 0_{2d\times 1}\\ 0_{1\times(2d+1)} & 1\\ 0_{1\times(2d+1)} & 0\end{pmatrix}$$

使得 $Q_{h,\ell}(X)=XW_{Q,h,\ell}$，对于 $K_{h,\ell}$，$V_{h,\ell}$也同理。

首先，我们注意到，在整个计算过程中，前 $2d+1$ 列始终不会发生变化。提醒读者，$X^{(\ell)}$ 是第 $\ell$ 个 Transformer 层的输出。

**命题3.3**。对于所有 $\ell$，$X^{(\ell)}$ 的前 $2d+1$ 列是 $X$ 的前 $2d+1$ 列。

证明。该命题成立，因为对于所有 $h$，$\ell$，$V_{h,\ell}(X)$ 的前 $2d+1$ 列均为 0。因此，每个注意力头的输出在前 $2d+1$ 列均为 0。接着，由于 $X^{(\ell)}=X^{(\ell-1)}+\sum_h f_{h,\ell}(X^{(\ell-1)})$，其中 $f_{h,\ell}$ 表示第 $\ell$ 层中的第 $h$ 个注意力头，因此 $X^{(\ell)}$ 的前 $2d+1$ 列是 $X^{(\ell-1)}$ 的前 $2d+1$ 列。根据归纳法，$X^{(\ell)}$ 的前 $2d+1$ 列是 $X^{(0)}=X$ 的前 $2d+1$ 列。 □

其次，我们断言给定的 hardmax Transformer 计算该 3-OV 实例。注意，由于引理 2.9，这已经足够，因为计算任意 softmax Transformer 的算法都可以以相同维度逼近任意 hardmax Transformer。考虑第 $(h,\ell)$ 个注意力头。注意

$$(Q_{h,\ell}(X)K_{h,\ell}(X)^\top)_{i,j}=\begin{cases}0 & \text{if } \max(i,j)=N+1\\ -a_i\cdot b_j\cdot c_{h,\ell} & \text{otherwise.}\end{cases}$$

对于 $1\le i\le N$，第 $i$ 行中的最大元素为 0，因为 $(Q_{h,\ell}(X)K_{h,\ell}(X)^\top)_{i,N+1}=0$。回顾 $I_{\max}(A(X)_i)$ 表示第 $i$ 行中取得最大值的下标集合，我们有 $|I_{\max}(A(X)_i)|\ge 2$，若 $(a_i,c_{h,\ell})$ 参与一个正交三元组，而 $|I_{\max}(A(X)_i)|=1$，否则。特别地，令 $t_i:=t_{i,(h,\ell)}$ 表示 $j$ 的个数，使得 $a_i\cdot b_j\cdot c_{h,\ell}=0$。于是我们有

$$\mathrm{hardmax}(Q_{h,\ell}(X)K_{h,\ell}(X)^\top)V_{h,\ell}(X)=\begin{pmatrix}0 & \cdots & 0 & \frac{t_1+2}{t_1+1}\\ 0 & \cdots & 0 & \frac{t_2+2}{t_2+1}\\ \vdots & \ddots & \vdots & \vdots\\ 0 & \cdots & 0 & \frac{t_N+2}{t_N+1}\\ 0 & \cdots & 0 & \frac{N+2}{N+1}\end{pmatrix}$$

特别地，对于每个 $i\le N$，最后一列中的第 $i$ 个元素在 $t_i=0$ 时为 2，否则至多为 $\frac{3}{2}$。因此，引理 2.9 意味着，存在一个 Transformer，使得每个注意力头都计算出一个输出 $Y_{h,\ell}$，该输出也能很好地近似具有嵌入 $Q_{h,\ell}$，$K_{h,\ell}$，$V_{h,\ell}$ 的 hardmax 注意力头。特别地，第 $\ell$ 层中第 $h$ 个注意力头的输出（记作 $Y_{h,\ell}$）满足

$$\left\|Y_{h,\ell}-\mathrm{hardmax}(Q_{h,\ell}(X)K_{h,\ell}(X)^\top)V_{h,\ell}(X)\right\|_\infty\le\frac{1}{10NHL}\tag{2}$$

对于某个关于 $L$, $H$, $N$ 的逆多项式。

令 $Z_{h,\ell}$ 表示 hardmax$(Q_{h,\ell}(X)K_{h,\ell}(X)^\top)V_{h,\ell}(X)$ 的最后一列。如果 3-OV 实例是否实例（即不存在正交三元组），那么对于每个 $Z_{h,\ell}[i] = 2$，即

$$\sum_{h,\ell}\sum_{i=1}^{N} Z_{h,\ell}[i] = 2NHL.$$

另一方面，如果 3-OV 是是实例（即存在正交三元组），那么存在某个 $i$,$(h,\ell)$，其中 $Z_{h,\ell}[i] \le \frac{3}{2}$，即

$$\sum_{h,\ell}\sum_{i=1}^{N} Z_{h,\ell}[i] \le 2NHL - \frac{1}{2}.$$

接下来考虑输出 $X^{(L)}$，尤其关注最后一列，因为前$2d+1$ 列与 $X = X^{(0)}$ 完全相同，根据命题 3.3。注意，我们的 Transformer 对所有 $1 \le i \le N$ 的输出

$$X^{(L)}[i,m] = X^{(0)}[i,m] + \sum_{h,\ell} Y_{h,\ell}[i,m] = \sum_{h,\ell} Y_{h,\ell}[i,m].$$

由于我们的算法计算给定的 Transformer 时存在 $\frac{1}{10N}$ 的逐项加性误差，因此实际上我们在最后一列得到一个向量 $\hat{X}$，使得对于所有 $i \in [N]$，均有 $|\hat{X}_j - X^{(L)}[i,m]| \le \frac{1}{10N}$。

因此，根据 Transformer 的计算输出，我们可以在 $O(N)$ 时间内计算 $\sum_{j=1}^{N} \hat{X}[j]$ 使得

$$\begin{aligned}\left|\sum_{i=1}^{N}\hat{X}[i] - \sum_{h,\ell}\sum_{i=1}^{N} Z_{h,\ell}[i]\right| &\le \left|\sum_{i=1}^{N}\hat{X}[i] - \sum_{i=1}^{N} X^{(L)}[i,m]\right| \\ &\quad + \left|\sum_{i=1}^{N} X^{(L)}[i,m] - \sum_{h,\ell} Y_{h,\ell}[i,m]\right| \\ &\quad + \left|\sum_{h,\ell} Y_{h,\ell}[i,m] - \sum_{h,\ell} Z_{h,\ell}[i]\right| \\ &\le 0.2.\end{aligned}$$

这里，我们对第一项应用三角不等式和算法的近似误差，对第二项应用上面的等式，并对第三项应用 (2)。请注意，Transformer 的输出是一个 $(N+1)\times m$ 矩阵，但我们只对最后一列的前 $N$ 个元素求和。

因此，和 $\sum_{j=1}^{N} \hat{X}[j]$ 若不存在正交三元组，则至少为 $2NHL - 0.2$；否则至多为$2NHL - 0.3$。因此，我们可以计算 3-OV 实例的输出。

□

最后，我们给出一个简单的条件下界，这表明高效的 Transformer 需要较大的规模才能计算 3-OV，也说明规模较小的 Transformer 无法处理某些 3 路交互实例。

**定理 3.4**。 在3-OV 假设下，任何具有线性时间嵌入和多层感知机映射、能够计算 $d$ 维3-OV 的 *Transformer* 都需要 $LHm = N^{1-o(1)}$。

证明。假设存在一个 Transformer $T$，其嵌入维度为 $m = \Theta(\log N)$，有 $L$ 层，每层有 $H$ 个注意力头，输入维度为 $m$，输出维度为 $m$，并且能够计算维度为 $m$ 的 3-OV。回顾一下，在高效的 Transformer 中，所有嵌入映射和多层感知机映射都需要 $O(m)$ 时间，因此每个注意力头都在 $O(Nm + N^2 m) = O(N^2 m)$ 时间内完成计算。按照标准算法，可以在 $\tilde{O}(N^2 mHL)$ 时间内计算该 Transformer。特别地，在 3-OV 假设下，这需要 $LHm = N^{1-o(1)}$。 □

# 4 算术电路工具

本节将介绍一些用于（扩展）算术电路的工具，这些工具将用于我们针对大嵌入维度的下界证明。

虽然 eAC 通常是一个能力更强的类别（例如，算术电路无法计算 指数函数$(x)$），但我们证明，计算矩阵乘法并不需要额外的门。更一般地，我们认为，事实上可以用标准算术电路来模拟计算低次函数的 eAC。对于我们的应用，我们给出二次函数的相应结果。该证明推广了 [Str73] 的结果；该结果指出，计算二次函数不需要除法门。

**定理 4.1.** 设 $F = \{F_1, \ldots, F_m\}$，其中 $F_i = \sum_{k,\ \ell=1}^{n} t_{ik\ell} x_k x_\ell$。假设存在一个规模为 $s$ 、计算 $F$ 的eAC $\mathcal{C}$。那么存在一个规模为 $O(s)$ 、计算 $F$ 的算术电路 。

证明。设 $\mathcal{C}$ 是计算 $F$ 的最优 eAC，并设 $W = (w_1, \ldots, w_s)$ 是 $\mathcal{C}$ 中所有门的一种排序，其中较早的门先计算。不失一般性地，将每个 $w_i(x_1, \ldots, x_n)$ 视为一个函数，我们可以假设对任意 $i$，都有 $w_i \neq 0$。此外，由于 $\mathcal{C}$ 对所有输入 $x \in \mathbb{R}^n$ 都计算 $F$，因此 $\mathcal{C}$ 对所有 $x$ 都有定义。特别地，若某个 $w_i$ 被用作对数门的输入，则对所有 $x$，都有 $w_i(x_1, \ldots, x_n) > 0$。同样地，若某个 $w_i$ 被用作 $/$ 门的分母，则对所有 $x$，都有 $w_i(x_1, \ldots, x_n) \neq 0$。通过替换 $\tilde{x}_i \leftarrow x_i z$，我们可以将每个门 $w_j$ 视为一个形式幂级数 $\tilde{w}_j \in \mathbb{R}(x_1, \ldots, x_n)[[z]]$，其系数是定义在 $x_1, \ldots, x_n$ 上的函数。之所以能够这样做，是因为当 $\mathcal{C}$ 在所有输入 $x \in \mathbb{R}^n$ 下均有定义时，所有形式幂级数也都有定义。首先，任意两个形式幂级数的和、差和积都有定义，而形式幂级数的幂运算也有定义（稍后我们将给出 ˜ 的精确形式）。对于某个门对应的函数 $f$ 及其形式幂级数 $\tilde{f}$，
注意到，只要 $f$ 的常数项系数为正，$\ln(f)$ 就有定义（稍后给出精确形式˜）。 $f$ 的常数项系数为 $f(0, \ldots, 0)$；根据我们的假设，对数门的输入在所有输入情况下均为正，因此 $f(0, \ldots, 0) > 0$，这意味着 $f$ 的常数项系数满足所需条件。对于除法门 $w_j = g/h$，我们的假设是，任何作为 ˜ 除法门分母输入的门，在所有输入情况下均非零；这保证了 $h$ 的常数项系数为 $h(0, \ldots, 0) \neq 0$。于是，下面的事实保证了 $w_j$ 的形式幂级数有定义。

**事实 4.2.** 设 $R$ 是一个环。幂级数 $\tilde{h}(z) = \sum_i a_i z^i \in R[[z]]$ 可逆，当且仅当 $a_0 \neq 0$。此外，其逆元为

$$\frac{1}{\tilde{h}(z)} = \frac{1}{a_0}\left(1 + q + q^2 + q^3 + \ldots\right)$$

其中 $q = -\sum_{i=1}^{\infty} \frac{a_i}{a_0} z^i$。

将门视为形式幂级数后，我们认为只需利用相关门的幂级数的低阶系数，就能轻松计算出所需的输出。具体而言，由于

所有 $F_i$ 的次数都至多为 2，因此我们可以在所有输入 $x_i$ 中加入一个新变量 $z$，并且只考察 $z^0$, $z^1$, $z^2$对应的系数。假设 $w_j$ 是一个输出门 $F_i$。那么，我们有

$$F_i = w_j(x_1, \ldots, x_n) = \sum_{k,\ell=1}^{n} t_{ik\ell} x_k x_\ell.$$

注意到

$$w_j(x_1 z, \ldots, x_n z) = \sum_{k,\ell=1}^{n} t_{ik\ell} x_k x_\ell z^2,$$

其形式幂级数就是它本身。因此，我们可以从形式幂级数 $\tilde{w}_j$ 中 $z^2$ 的系数计算出 $F_i$。

我们已经证明，可以根据与输出门对应的幂级数的第二个系数，高效地计算出 $F$。虽然由于每个门可能有无穷多项，我们无法计算每个门的完整幂级数，但我们将介绍如何计算低阶系数，即对所有 $\tilde{w}_i$，计算 1、$z$、$z^2$ 的系数。我们通过归纳法进行。记 $\tilde{w}_i = \sum_{j\geq 0} c_j z^j$，并计算 $c_0$、$c_1$、$c_2$。对于输入门，$w_i(x_1, \ldots\ldots, x_n) = x_j$，其中某个 $x_j$ 满足 $\tilde{w}_i = x_j z$。特别地，$c_0 = c_2 = 0$ 且 $c_1 = x_j$。这需要 $O(1)$ 个门。

**情形 1：加法与减法**。设 $\tilde{w}_i = \tilde{g}_i + h_i$ 其中 $\tilde{g}_i$, $h_i$ 是幂级数，且

$$\tilde{g}_i = \sum_{j=0}^{\infty} u_j z^j \quad , \quad \tilde{h}_i = \sum_{j=0}^{\infty} v_j z_j.$$

前三个系数可以很容易地通过加法和减法合并，即 $c_j = u_j \pm v_j$ 对所有 $j$均成立。同样，这需要 $O(1)$ 个门。

**情形 2：乘法**。假设 $\tilde{w}_i = \tilde{g}_i \cdot \tilde{h}_i$，其中 $\tilde{g}_i = \sum_{j=0}^{\infty} u_j z^j$ 且 $\tilde{h}_i = \sum_{j=0}^{\infty} v_j z^j$。那么，若 $\tilde{w}_i = \sum_j c_j z^j$，我们有 $c_0 = u_0 v_0$、$c_1 = u_0 v_1 + u_1 v_0$和 $c_2 = u_2 v_0 + u_1 v_1 + u_0 v_2$。这需要 $O(1)$ 个门。

**情形 3：除法**。假设 $\tilde{w}_i = \tilde{g}_i / \tilde{h}_i$ 其中 $\tilde{g}_i = \sum_{j=0}^{\infty} u_j z^j$ 且 $\tilde{h}_i = \sum_{j=0}^{\infty} v_j z^j$。由于$v_0 \neq 0$，根据事实 4.2，我们得到其逆元：

$$\frac{1}{\tilde{h}_i} = \frac{1}{v_0}\left(1 + q + q^2 + q^3 + \ldots\right)$$

与 $q = -\sum_{j=1}^{\infty} \frac{v_j}{v_0} z^j$。特别地，我们有

$$\begin{aligned}
\frac{\tilde{g}_i}{\tilde{h}_i} &= \frac{1}{v_0}(u_0 + u_1 z + u_2 z^2 + \ldots) \cdot \left(1 + \left(-\frac{v_1}{v_0} z - \frac{v_2}{v_0} z^2 - \ldots\right) + \left(-\frac{v_1}{v_0} z - \ldots\right)^2 + \ldots\right) \\
&= (u'_0 + u'_1 z + u'_2 z^2 + u'_3 z^3 + \ldots) \cdot \left(1 + \left(-v'_1 z - v'_2 z^2 - \ldots\right) + \left(-v'_1 z - \ldots\right)^2 + \ldots\right)
\end{aligned}$$

其中我们设定 $u'_j = \frac{u_j}{v_0}$ 和 $v'_j = \frac{v_j}{v_0}$。最后，我们有

$$\begin{aligned}
c_0 &= u'_0 \\
c_1 &= u'_1 - u'_0 v'_1 \\
c_2 &= u'_2 - u'_1 v'_1 + u'_0 (v'^2_1 - v'_2).
\end{aligned}$$

这需要 $O(1)$ 个门。

**情况 4：指数**。 假设 $\tilde{w}_i = \text{指数函数}(\tilde{g}_i)$，其中 $\tilde{g}_i = \sum_{j=0}^{\infty} u_j z^j$。然后，利用指数函数$(x)$ 的形式幂级数 $= \sum_{k=0}^{\infty} \frac{x^k}{k}!$，我们得到

$$\begin{aligned}\exp(\tilde{g}_i) &= \exp\left(\sum_{j\geq 0} u_j z^j\right) \\ &= \exp(u_0)\exp\left(\sum_{j\geq 1} u_j z^j\right) \\ &= \exp(u_0)\left(\sum_{k\geq 0}\frac{1}{k!}\left(\sum_{j\geq 1} u_j z^j\right)^k\right).\end{aligned}$$

注意，对于每个固定的 $j \geq 0$，上述求和中只有有限项会对 $z^j$ 的系数作出贡献，因此形式幂级数是良定义的。特别地，

$$\begin{aligned}c_0 &= e^{u_0} \\ c_1 &= e^{u_0}u_1 \\ c_2 &= e^{u_0}\left(u_2 + \frac{u_1^2}{2}\right).\end{aligned}$$

这需要 $O(1)$ 个门。尽管公式中使用了 $\exp(u_0)$，但它是一个常数，可以在算术电路中实现。

**情况 5：对数**。设 $\tilde{w}_i = \ln(\tilde{g}_i)$，其中 $\tilde{g}_i = \sum_{j\geq 0} u_j z^j$。然后应用幂级数恒等式 $\ln(1+x)$ $= \sum_{k=1}^{\infty}(-1)^{k+1}\frac{x^k}{k}$，可得 —

$$\begin{aligned}\ln(\tilde{g}_i) &= \ln u_0 + \ln\left(1 + \sum_{j\geq 1}\frac{u_j}{u_0}z^j\right) \\ &= \ln u_0 + \sum_{k\geq 1}\frac{(-1)^{k+1}}{k}\cdot\left(\sum_{j\geq 1}\frac{u_j}{u_0}z^j\right)^k\end{aligned}$$

同样，对于每个固定的 $j \geq 0$，上述求和中只有有限项会对 $z^j$ 的系数作出贡献，因此形式幂级数是良定义的。特别地，

$$\begin{aligned}c_0 &= \ln u_0 \\ c_1 &= \frac{u_1}{u_0} \\ c_2 &= \frac{u_2}{u_0} - \frac{1}{2}\left(\frac{u_1}{u_0}\right)^2.\end{aligned}$$

这需要 $O(1)$ 个门。与指数函数的情况类似，我们只对一个常数使用 ln，因此可以在算术电路模型中实现。

总体而言，我们的电路需要 $O(s)$个门来计算所有相关系数。 □

我们需要的下一个工具是 Baur-Strassen 定理。该定理指出，给定一个计算某个函数的算术电路，我们可以用规模相近的电路计算出所有偏导数。

**定理 4.3** ([BS83])。 设 $F$： $\mathbb{R}^n \to \mathbb{R}$ 是一个函数，且 $F' = (\frac{\partial F}{\partial x_1}, \ldots\ldots, \frac{\partial F}{\partial x_n})$。如果存在一个规模为 $s$、用于计算 $F$的算术电路，则存在一个规模为 $O(s)$、用于计算 $F'$的算术电路。

Baur 和 Strassen [BS83] 得出这一结果后，Morgenstern [Mor85] 给出了 Baur-Strassen定理的一个简化构造性证明。沿用 Morgenstern 的证明策略，我们将 Baur-Strassen定理推广到 eAC。

**定理 4.4** (扩展 Baur-Strassen 定理)。 设 $F: \mathbb{R}^n \to \mathbb{R}$ 是一个函数，且 $F' = \left\{\frac{\partial F}{\partial x_1}, \ldots, \frac{\partial F}{\partial x_n}\right\}$。如果存在一个规模为 $s$、用于计算 $F$ 的 eAC，则存在一个规模为 $O(s)$、用于计算$F'$的 eAC。

证明。 设 $\{g_1, \ldots, g_s\}$是计算 $F$ 的最小规模 eAC。设 $F^{(i)}$是由 $g_i, \ldots, g_s$ 计算得到的函数。注意，$F^{(i)}$有 $s-i+1$ 个运算，输入变量为 $\{x_1, \ldots, x_n, g_1, \ldots, g_{i-1}\}$。我们用归纳法证明所需结论。为简化记号，对于 $1 \le j \le n+s$,$z_j = x_j$，若 $j \le n$，则定义为.....；若 $z_j = g_{j-n}$，则定义为 $j > n$。我们声称， $F_{(i)}$的偏导数可以由规模为 5$(s-i+1)+2$ 的 eAC 计算。于是， $F = F^{(1)}$的偏导数可以由规模为 $O(s)$ 的电路计算。

我们从基础情形开始。注意 $F^{(s+1)}(x_1、\ldots\ldots、x_n、g_1、\ldots\ldots、g_s) = g_s$有偏导数$\frac{\partial F^{(s+1)}}{\partial g_s} = 1$和~~$\frac{\partial F}{\partial z}_{(s+1)} = 0$~~，对任意 $z \in \{x_i\} \cup \{g_1, \ldots, g_{s-1}\}$均成立。这些值都是常数，因此我们可以用规模为 1 的 eAC 来计算它们。虽然我们需要门来存储这两个不同的值，但不必分别存储所有导数。在后续计算中，只需记住在相应的门处使用哪个值即可。这会使计算得到的电路额外增加 2 个门。

现在我们继续讨论归纳情形。假设对于所有 $\frac{\partial F^{(i+1)}}{\partial z_j}$， ~~$j \le n+i$~~均已计算得到。于是， $F_{(i)}$是一个 eAC， $F^{(i+1)}$其输入 $g_i$被定义 2.2 中定义的某种运算替代2.2。对于变量 $z_j$，针对 $j \le n+i-1$，我们可以根据链式法则计算 $\frac{\partial F^{(i)}}{\partial z_j}$： ——

$$\frac{\partial F^{(i)}}{\partial z_j} = \frac{\partial F^{(i+1)}}{\partial z_j} + \frac{\partial F^{(i+1)}}{\partial g_i}\frac{\partial g_i}{\partial z_j}.$$

对应 $g_i$中的一个运算的情形 $\{+, -, \times, /\}$已在标准 Baur-Strassen定理的 Morgen- stern 证明中处理 [Mor85]。因此，我们关注 $g_i$对应于一个 指数函数或 自然对数函数门的情形。

假设 $g_i$涉及某个 $k \le n+i-1$的输入 $z_k$，并且如果 $j \ne k$，那么 ~~$\frac{\partial F}{\partial z_j}_{(i)} = \frac{\partial F^{(i+1)}}{\partial z_j}$~~。因此，只需计算 $\frac{\partial F^{(i)}}{\partial z_k}$。 ——

如果对于某个 $k \le n+i-1$， $g_i =$为 $z_k$的指数函数，我们有

$$\frac{\partial F^{(i)}}{\partial z_k} = \frac{\partial F^{(i+1)}}{\partial z_k} + \frac{\partial F^{(i+1)}}{\partial g_i}\exp(z_k)$$

其中，我们有 $\frac{\partial F^{(i+1)}}{\partial z_k}$, ~~$\frac{\partial F}{(i+1)}\partial g_i$~~，这是通过归纳法计算得到的，而 exp($z_k$) 可通过一次运算计算得到。求和还需要额外的一次运算。

类似地，如果 $g_i =$为 $z_k$的自然对数函数，我们有

$$\frac{\partial F^{(i)}}{\partial z_k} = \frac{\partial F^{(i+1)}}{\partial z_k} + \frac{\partial F^{(i+1)}}{\partial g_i}\frac{1}{z_k}$$

其中，上述项可通过 3 次运算计算得出。这证明了归纳命题。 □

我们的最后一个要素是：对计算相互独立的矩阵乘积的任意算术电路的规模给出一个下界。

**引理 4.5**。对于任意整数 $\ell \geq 1$，计算 $\ell$ 个相互独立的 $n \times n \times n$矩阵乘积的任意算术电路，其规模为 $\ell \cdot n^{\omega - o(1)}$。

其证明可由渐近求和不等式的标准应用得到 [Sch81]。为完整起见，我们给出该引理的证明及必要的定义。

定义一个关于三组变量 $X = \{x_1, \ldots, x_{|X|}\}$，$Y = \{y_1, \ldots, y_{|Y|}\}$，$Z = \{z_1, \ldots, z_{|Z|}\}$的张量如下：

$$T = \sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} \sum_{k=1}^{|Z|} a_{ijk} x_i y_j z_k$$

对于某些系数 $a_{ijk} \in \mathbb{R}$。计算该张量的代数电路，是指这样一个代数电路：给定输入 $X$，$Y$，对于每个 $k \in [|Z|]$输出 $\sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} a_{ijk} x_i y_j$。

张量 $T$ 的秩 $R(T)$ 定义为满足以下条件的最小整数 $r$：$T$ 可以表示为

$$\sum_{\ell=1}^{r} \left( \sum_{i=1}^{|X|} \alpha_{\ell,i} x_i \right) \left( \sum_{j=1}^{|Y|} \beta_{\ell,j} y_j \right) \left( \sum_{k=1}^{|Z|} \gamma_{\ell,k} z_k \right)$$

~对于某些值 $\alpha_{\ell,\,i}$，$\beta_{\ell,\,j}$，$\gamma_{\ell,\,k}$。张量 $T$的渐近秩 $R(T)$ 定义为 $\lim_{m\to\infty}(R(T^{\otimes m}))^{1/m}$，其上界为 $R(T)$，其中 $T^{\otimes m}$表示 $T$的第 $m$次张量幂。

$n \times n \times n$ 个矩阵乘积的矩阵乘法张量是

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} x_{ij} y_{jk} z_{ik},$$

并将其记为 $\langle n,\ n,\ n\rangle$。我们还将 $\ell$ 个 $n\times n\times n$ 的不相交并记为 $\ell\odot\langle n,\ n,\ n\rangle$。现在我们可以陈述渐近求和不等式了。

**定理 4.6**（渐近求和不等式 [Sch81]）。对于整数 $\ell$，$n \geq 1$，

$$\ell \cdot n^{\omega} \leq \tilde{R}\left(\ell \odot \langle n, n, n\rangle\right).$$

现在我们证明引理4.5。

引理 4.5 的证明。假设存在一个规模为 $O(\ell \cdot n^{\omega-\varepsilon})$ 的算术电路，用于计算 $\ell \odot \langle n, n, n\rangle$，其中某个 $\varepsilon > 0$。根据定义，该电路计算 $\ell$ 个 $n \times n \times n$ 矩阵乘积的独立实例。众所周知，计算某张量的电路的秩至多是其规模的两倍 [Str73]（另见例如 [Blä13，定理 4.7]），因此 $R(\ell\odot\langle n, n, n\rangle) \leq O(\ell\cdot n^{\omega-\varepsilon})$，进而推出$R(\ell \odot \langle n, n, n\rangle) \leq O(\ell \cdot n^{\omega-\varepsilon})$。因此，根据渐近求和不等式 [Sch81]，

$$\begin{aligned}
\omega &\leq \frac{\log\left(\tilde{R}(\ell \odot \langle n, n, n\rangle)/\ell\right)}{\log n} \\
&\leq \frac{\log\left(O(n^{\omega-\varepsilon})\right)}{\log n} \\
&\leq \frac{\delta + (\omega - \varepsilon)\log n}{\log n} \\
&= \omega - \varepsilon + \delta/\log n,
\end{aligned}$$

其中 $\delta$ 是一个常数，取决于大-$O$ 表示法中隐藏的常数。由于我们可以取$n \to \infty$，这就产生了矛盾。因此，$\ell \odot \langle n,\ n,\ n\rangle$ 对于任意 $\varepsilon > 0$，都没有规模为$O(\ell \cdot n^{\omega-\varepsilon})$ 的算术电路，也就是说，它的规模至少为 $\ell \cdot n^{\omega - o(1)}$。 □

## 5 大嵌入维度的下界

本节中，我们在扩展算术电路模型下证明了针对大嵌入维度 Transformer 的下界。回顾定理 1.2:

**定理 5.1** （形式化定理 1.2）。 存在一个 *Transformer* $T$，其输入长度为 $N+1$，嵌入维度为 $m=2N+3$， $L$ 层，每层 $H$ 个注意力头，输入维度和输出维度为 $2N+3$，并满足以下条件:

1. $T$ 可由一个eAC计算，每个嵌入可由规模为$O(m)$的算术电路计算，且所有多层感知机映射均为恒等映射。
2. 任何计算 $T$的eAC，其规模至少为$LHN^{\omega-o(1)}-O\left(LHN^2\right)$。

我们首先定义一个计算矩阵乘积行和的 Transformer。首先，我们给出一个简单的注意力头，它在进行逐元素指数运算后，计算单个矩阵乘积的（归一化）行和。该注意力头还会输出归一化因子，我们随后将用其恢复矩阵乘积的元素。

**引理 5.2.** 设 $m \geq N$。存在一个注意力单元 $\mathrm{Att}_{\mathsf{MatMul}}$，其输入长度为 $N+1$，嵌入维度为 $m'=\max(m+1, 2N+3)$，输入维度为 $2N+3$，使得对任意 $N \times m$矩阵 $A$、$B$以及向量 $D \in \mathbb{R}^N$，都存在嵌入权重，使得在给定输入 $X=\begin{pmatrix} Z_1 & I_N & 1_{N\times 1} & 0_{N\times 1} \\ Z_2 & 0_{1\times N} & 0 & 1 \end{pmatrix} \in \mathbb{R}^{(N+1)\times(2N+3)}$（其中 $Z_1 \in \mathbb{R}^{N\times(N+1)}$、$Z_2 \in \mathbb{R}^{1\times(N+1)}$ 为任意矩阵）时，输出 $Y \in \mathbb{R}^{(N+1)\times m'}$ 满足

$$Y[i,j]=\begin{cases} \frac{N}{N+1} & \textit{if } i=N+1 \textit{ and } j=1 \\ \frac{D[j-1]}{N+1} & \textit{if } i=N+1 \textit{ and } 1<j\leq N+1 \\ \frac{S_i}{1+S_i} & \textit{if } i\in[N] \textit{ and } j=1 \\ \frac{D[j-1]}{1+S_i} & \textit{if } i\in[N] \textit{ and } 1<j\leq N+1 \\ 0 & \text{o/w} \end{cases}$$

其中 $S_i=\sum_{j=1}^{N}$指数函数$(AB^\top)_{ij}$（对 $i\in[N]$）。此外，所有嵌入映射均可由规模为 $O(Nm)$ 的算术电路计算。

证明。考虑嵌入

$$Q(X)=\begin{pmatrix} A & 0_{N\times 1} \\ 0_{1\times m} & 0 \end{pmatrix}, \quad K(X)=\begin{pmatrix} B & 0_{N\times 1} \\ 0_{1\times m} & 0 \end{pmatrix} \in \mathbb{R}^{(N+1)\times(m+1)}$$

使得

$$\exp\left(Q(X)K(X)^\top\right)=\begin{pmatrix} \exp(AB^\top) & 1_{N\times 1} \\ 1_{1\times N} & 1 \end{pmatrix}$$

注意到嵌入映射可以表示为 $Q(X)=XW_Q$，$K(X)=XW_K$ 对于矩阵$W_Q$，$W_K \in \mathbb{R}^{(2N+3)\times(m+1)}$:

$$W_Q=\begin{pmatrix} 0_{(N+1)\times m} & 0_{(N+1)\times 1} \\ A & 0_{N\times 1} \\ 0_{2\times m} & 0_{2\times 1} \end{pmatrix}, \quad W_K=\begin{pmatrix} 0_{(N+1)\times m} & 0_{(N+1)\times 1} \\ B & 0_{N\times 1} \\ 0_{2\times m} & 0_{2\times 1} \end{pmatrix}$$

如果 $m' > m+1$，我们可以向 $W_Q$，$W_K$ 追加零列，因此 $Q(X)$ $Q(K)$ 而不改变 $Q(X)K(X)^\top$ 的值。然后，考虑该嵌入

$$V(X) = \begin{pmatrix} 1_{N\times 1} & 0_{N\times 1} & 0_{N\times 1} & \dots & 0_{N\times 1} \\ 0 & D_1 & D_2 & \dots & D_N \end{pmatrix} \in \mathbb{R}^{(N+1)\times(N+1)}.$$

注意到，$V$可以表示为 $V(X) = XW_V$，其中

$$W_V = \begin{pmatrix} 0_{(2N+1)\times 1} & 0_{(2N+1)\times 1} & 0_{(2N+1)\times(N-2)} & 0_{(2N+1)\times 1} \\ 1 & 0 & 0_{1\times(N-2)} & 0 \\ 0 & D_1 & \dots & D_N \end{pmatrix}$$

与之前一样，若 $m' > N+1$，我们可以在 $W_V$后追加零列，因此得到 $V(X)$，这只会在输出 $Y$后追加零列。注意，$Q(X)$ $K(X)$ $V(X)$可以通过 $X$计算得到，所需的是规模为 $O(Nm)$的 eAC（实际上是 AC）电路，并将相关矩阵 $A$、$B$ 以及向量 $D$ 编码到权重中。接着，令$Y \in \mathbb{R}^{(N+1)\times m'}$ 表示注意力的输出，并令 $S_i := \sum_j \exp(AB^T)_{ij}$。注意，输出正如所述。 □

接下来，我们展示如何计算多个矩阵乘积之和。

**引理 5.3**。设 $L$、$H \geq 1$和 $m \geq N$。存在一个 *Transformer* $T$，其输入长度为 $N+1$，嵌入维度为 $m' = \max(m+1, 2N+3)$，有 $L$ 层，每层有 $H$ 个注意力头，输入维度为 $2N+3$，输出维度为 $m'$，使得对于任意一组矩阵 $A_k$、$B_k \in \mathbb{R}^{N\times m}$（其中 $k \in [LH]$）和向量 $D_k \in \mathbb{R}^N$，都存在相应的嵌入权重，使得给定输入 $X = \begin{pmatrix} 0_{N\times(N+1)} & I_N & 1_{N\times 1} & 0_{N\times 1} \\ 0_{1\times(N+1)} & 0_{1\times N} & 0 & 1 \end{pmatrix}$时，输出 $Y \in \mathbb{R}^{(N+1)\times m'}$ 的各项为

$$Y[i,j] = \begin{cases} \frac{LHN}{N+1} & \text{if } i = N+1 \text{ and } j = 1 \\ \sum_{k=1}^{LH} \frac{D_k[j-1]}{N+1} & \text{if } i = N+1 \text{ and } 1 < j \leq N+1 \\ \sum_{k=1}^{LH} \frac{S_{ki}}{1+S_{ki}} & \text{if } i \in [N] \text{ and } j = 1 \\ \sum_{k=1}^{LH} \frac{D_k[j-1]}{1+S_{ki}} & \text{if } i \in [N] \text{ and } 1 < j \leq N+1 \\ 0 & \text{o/w} \end{cases}$$

其中 $S_{ki} = \sum_j$指数函数$(A_k B_k^\top)_{ij}$。此外，所有嵌入映射都可以由规模为 $O(LHNm)$ 的算术电路计算，并且多层感知机可以取为恒等映射。

对计算嵌入映射和多层感知机映射的电路规模所作的最后一项规定，确保这些计算不会成为计算 Transformer 的计算瓶颈。

证明。首先，我们指出，在引理 5.2的构造中，可以假设输入维度实际上为 $m'$，方法是在输入中附加零列 $X$，并在权重矩阵中附加零行$W_Q$，$W_K$，$W_V$而不改变所计算的嵌入 $Q(X)$ $K(X)$ $V(X)$。

我们构造一个 Transformer，用于计算 $S_0$，$S_1$，……，$S_L \in \mathbb{R}^{(N+1)\times(N+1)}$ 其中，对于 $\ell > 0$，$S_\ell$ 是满足以下条件的矩阵

$$S_\ell[i,j] = \begin{cases} \frac{\ell HN}{N+1} & \text{if } i = N+1 \text{ and } j = 1 \\ \sum_{k=1}^{\ell H} \frac{D_k[j-1]}{N+1} & \text{if } i = N+1 \text{ and } 1 < j \leq N+1 \\ \sum_{k=1}^{\ell H} \frac{S_{ki}}{1+S_{ki}} & \text{if } i \in [N] \text{ and } j = 1 \\ \sum_{k=1}^{\ell H} \frac{D_k[j-1]}{1+S_{ki}} & \text{if } i \in [N] \text{ and } 1 < j \leq N+1 \\ 0 & \text{o/w} \end{cases}$$

其中 $S_{ki} = \sum_{j=1}^{N} \exp(A_k B_k^\top)_{ij}$，而 $X$ 表示 Transformer 的输入 .

我们进行归纳论证。在基础情形 $S_0 = X$中，Transformer 的输入为 $S_{\ell-1}$。假设已计算出 $h \in [H]$。对于任意 $h \in [H]$，引理5.2 表明，我们可以得到一个注意力头 $\mathrm{Att}^{(h,\ell)} \in \mathrm{Att}_{m+1,\ 2m+1}^{N+1}$ 其输出矩阵 $Y^{(h,\ell)}$ 满足

$$S_\ell = S_{\ell-1} + \sum_h Y^{(h,\ell)}.$$

这是因为 $Y^{(h,\ell)}$ 仅在前 $N+1$ 列中非零，并且我们将在引理 5.2 中证明，注意力头的输入在前 $N+1$ 列中可以是任意的。

对 $H$ 个注意力头以及前一层的输出求和，我们得到第$\ell$层的输出为 $S_\ell$，正如所需。最后，我们注意到 $T(X)$ 恰好是 $S_L$，因此将输出多层感知机$\Psi$ 定义为恒等映射，即将 $S_L$ 的每一行 $v$ 映射为 $\psi(v) = v$。注意，每个嵌入都可以由规模为 $O(Nm)$ 的算术电路计算得到。 □

我们现在准备证明定理 5.1。

定理 *5.1*的证明。 假设存在一个规模为 $s$的 eAC，用于计算 Transformer $T$，其上下文长度为 $N+1$，嵌入维度为 $2N+3$，共 $L$ 层，每层 $H$ 个注意力头，输入和输出维度为 $2N+3$，如引理 5.3 所规定（注意，该 Transformer 满足第一个期望性质）。我们构造一个具有 $LHN^2$ 个额外输入的 eAC，记为 $C_{kij}$ 和 $D_{ki}$，其中$k \in [LH]$ 且 $i,\ j \in [N]$。现在我们描述一个由三部分组成的 eAC：构造 Transformer 输入的预处理阶段、模拟该 Transformer 的电路，以及对该 Transformer 的输出求和的后处理阶段。

我们从预处理阶段开始。对于 $k \in [LH]$，将 $C_k$ 定义为元素为 $C_k[i,j] = C_{kij}$的 $N \times N$ 矩阵，并将 $D_k$ 定义为元素为 $D_k[i] = D_{ki}$的 $N$维向量。然后，给定 $A_k,\ B_k \in \mathbb{R}^{N\times N}$，我们如下构造矩阵 $\tilde{A}_k$，$\tilde{B}_k \in \mathbb{R}^{N\times 2N}$：

$$\tilde{A}_k = \begin{pmatrix} A_k & C_k \end{pmatrix}, \quad \tilde{B}_k = \begin{pmatrix} B_k & I \end{pmatrix}$$

注意到存在一个规模为 $O\left(LHN^2\right)$ 的扩展算术电路，可计算 $\tilde{A}_k$，$\tilde{B}_k$，对所有 $k \in [LH]$ 均成立。

令 $T$为输入长度为 $N+1$、嵌入维度为 $2N+3$、$L$ 层、每层有 $H$ 个注意力头、输入维度为 $2N+3$、输出维度为 $2N+3$ 的 Transformer，由引理 5.3 给出，其输出 $Y$ 是一个矩阵，矩阵元素为

$$Y_{i,1} = \sum_{k=1}^{LH} \frac{S_{ki}}{1+S_{ki}}$$

对于 $i \in [N]$ 和

$$Y_{i,j} = \sum_k \frac{D_k[j-1]}{1+S_{ki}}$$

对于 $i \in [N]$和 $1 < j \le N+1$，其中 $S_{ki} = \sum_j$指数函数 $\left(\tilde{A}_k \tilde{B}_k^\top\right)_{ij}$。回想一下，我们将计算 $T$ 的 eAC 的规模记为 $s$。通过增加 $N$ 个门，我们可以对第一列求和，并得到一个规模为 $O\left(s + LHN^2\right)$、计算

$$f\left(\{A_k, B_k, C_k, D_k\}_{k=1}^{LH}\right) = \sum_{i=1}^{N} Y_{i,1} = \sum_{i=1}^{N} \sum_{(h,\ell)} Y_{i,1}^{(h,\ell)} = \sum_{i=1}^{N} \sum_{k=1}^{LH} \frac{S_{ki}}{1+S_{ki}}.$$

现在，考虑 $f$的偏导数。特别地，对于任意 $k \in [LH]$，$i_0$，$j_0 \in [N]$，其中 $k$被标识为 $(h, \ell)$，$f$ 关于 $C_{ki_0j_0}$ 的偏导数为

$$\begin{aligned}
\frac{\partial f}{\partial C_{ki_0j_0}} &= \frac{\partial}{\partial C_{ki_0j_0}} \sum_{i=1}^{N} \sum_{(h,\ell)}^{LH} Y_{i,1}^{(h,\ell)} \\
&= \frac{\partial}{\partial C_{ki_0j_0}} Y_{i_0,1}^{(h,\ell)} \\
&= \frac{\partial}{\partial C_{ki_0j_0}} \frac{S_{ki_0}}{1+S_{ki_0}} \\
&= \frac{\frac{\partial S_{ki_0}}{\partial C_{ki_0j_0}}}{(1+S_{ki_0})^2} \\
&= \frac{\exp((A_k B_k^\top)_{i_0j_0} + C_{ki_0j_0})}{(1+S_{ki_0})^2}
\end{aligned}$$

其中在第二个和第五个等式中，我们只包含涉及 $C_{ki_0j_0}$ 的加项；在第五个等式中，我们使用

$$(\tilde{A}_k \tilde{B}_k^\top)_{i0j0} = (A_k B_k^\top)_{i0j0} + C_{ki0j0}.$$

那么，如果将所有辅助 $C_{kij}$ 的值设为 0，我们就得到

$$\frac{\partial f}{\partial C_{ki_0j_0}} = \frac{\exp((A_k B_k^\top)_{i_0j_0})}{(1+S_{ki_0})^2}.$$

同样，将 $Y_{i,\ i+1}$ 相加，我们得到一个计算该函数的 eAC

$$g\left(\{A_k, B_k, C_k, D_k\}_{k=1}^{LH}\right) = \sum_{i=1}^{N} \sum_{k=1}^{LH} \frac{D_k[i]}{1+S_{ki}}.$$

对 $D_k[i_0]$ 求偏导数，

$$\frac{\partial g}{\partial D_k[i_0]} = \frac{\partial}{\partial D_k[i_0]} Y^{(h,\ell)}[i_0, i_0+1] = \frac{1}{1+S_{ki_0}}.$$

将扩展 Baur-Strassen 定理（定理 4.4）应用于 $f$ 和 $g$，存在一个规模为 $O(s+LHN^2)$ 的 eAC，可计算 $f$ 和 $g$ 的所有偏导数。然后，我们可以轻松计算

$$\left(A_k B_k^\top\right)_{i_0j_0} = \ln\left(\frac{\partial f}{\partial C_{ki_0j_0}}\right) - 2\ln\left(\frac{\partial g}{\partial D_k[i_0]}\right)$$

对于所有 $k \in [LH]$ 和 $i_0$，$j_0 \in [N]$，并使用 $O\left(LHN^2\right)$ 个额外的门。

到目前为止，我们已经得到一个规模为 $O(s+LHN^2)$ 的 eAC，它计算 $A_k B_k^\top$，其中 $k \in [LH]$。根据定理 4.1，还存在一个规模为 $O(s+LHN^2)$ 的算术电路，它计算 $A_k B_k^\top$，其中 $k \in [LH]$，即它计算 $LH$ 个规模为 $N \times N \times N$的独立矩阵乘积。然后，根据引理 4.5，任何计算这些独立矩阵乘积的算术电路，其规模都至少为 $LHN^{\omega-o(1)}$。因此，必有$s \geq LHN^{\omega-o(1)} - O(LHN^2)$，如所需。

□

## 6 讨论与开放问题

我们给出了关于高效计算 Transformer 的下界。具体而言，我们证明，在小嵌入情形 $m = N^{o(1)}$ 下，一个 $L$ 层、每层具有 $H$ 个注意力头的 Transformer，在 $LHN^{2-o(1)}$ 时间内无法计算，前提是 3-OV 假设成立。因此，我们建立了一个条件下界，表明标准 $LHN^{2+o(1)}$ 算法在次多项式因子范围内是最优的，这显著改进了先前仅基于注意力困难性的下界 [KWH23、AS24、GHS+26]。在大嵌入情形下，我们证明，即使多层感知机和嵌入层可以由规模较小的电路计算，计算 Transformer 的电路仍能计算许多独立矩阵乘积，因此必须具有较大规模。具体而言，即使多层感知机和嵌入函数可以由线性规模电路计算，计算一个 $L$ 层、每层具有 $H$ 个注意力头且嵌入维度为 $N$的 Transformer，也必需且只需一个规模为 $LHN^{\omega\pm o(1)}$的电路。

最后，我们以一些开放问题作结。首先，一个有趣的方向是将定理 5.1扩展到 Word-RAM模型中的任意算法（即将下界推广到扩展算术电路模型之外）。

其次，尽管定理 5.1 排除了这样一种算术电路：对于嵌入维度为 $m = N$ 的情形，其性能优于朴素算法，但要排除对于任意多项式 $m$ 都能改进标准算法（使用快速矩阵乘法加速）的算术电路，仍然是一个有趣的开放问题。我们指出，对于大多数多项式 $m$，我们的技术能够推出比定理 1.1 更强的下界。特别地，我们可以构造一个嵌入维度为 $m$ 的 Transformer，计算 $LH$ 个相互独立的 $m \times N$ 与 $N \times m$ 矩阵的乘积。因此，任何计算嵌入维度为 $m = N^a$ 的 Transformer 的电路都需要 $a > 0$ $LHN^{\omega(1,a,a)-o(1)}$ 规模。这里，$\omega(a, b, c)$ 表示矩形矩阵乘法指数，即满足如下条件的最小常数：对于任意 $\varepsilon > 0$，存在一个规模为 $O\left(n^{\omega(a,b,c)+\varepsilon}\right)$ 的算术电路，能够计算一个 $n^a \times n^b \times n^c$ 矩阵乘积。于是，问题在于弥合这一上界与 $LHN^{\omega(1,a,1)+o(1)}$ 下界之间的差距。我们指出，即使对于单个注意力头，也没有已知的从 $N \times m \times N$ 矩形矩阵乘法到嵌入维度为 $m$ 的注意力计算的归约。

最后，我们的工作聚焦于最坏情况下界。是否存在某些输入分布，使我们能够开发出更快的算法？对于结构假设，也可以提出类似的问题：除已知的注意力计算 [AS24、GHS+26]之外，哪些结构假设能够支持高效算法？

## 致谢

我们感谢匿名审稿人提出的许多有益意见和建议。

## 参考文献

[ABHS19] Amir Abboud、Karl Bringmann、Danny Hermelin 和 Dvir Shabtay。子集和与双准则路径的基于 SETH 的下界。载于第 *30* 届 *ACM-SIAM* 离散算法研讨会（*SODA*）论文集，第 41–57 页，2019年。4[ADV+25] Josh Alman、Ran Duan、Virginia Vassilevska Williams、Yinzhan Xu、Zixuan Xu 和 Renfei Zhou。更大的不对称性带来更快的矩阵乘法。载于*2025* 年 *ACM-SIAM* 离散算法研讨会（*SODA*）论文集，第 2005–2039 页。SIAM，2025年。3

[ARW17] Amir Abboud、Aviad Rubinstein 和 Ryan Williams。用于 P 中近似困难性的分布式 PCP定理。载于《2017 年 IEEE 第 58 届计算机科学基础年度研讨会（FOCS）论文集》，第 25–36 页。IEEE，2017年。4[AS24] Josh Alman 和 Zhao Song。快速注意力需要有界项。载于《第 37 届神经信息处理系统国际会议论文集》。Curran Associates Inc.，2024年。2、4、9、27[ATYZ24] Josh Alman、Ethan Turok、Hantao Yu 和 Hengzhi Zhang。张量秩与动态规划的精细复杂度。载于《第 15 届理论计算机科学创新会议（ITCS 2024）论文集》，第 4:1–4:23 页，2024年。4[AY25] Josh Alman 和 Hantao Yu。Transformer 次二次替代方案的根本局限性。载于《第 13 届国际学习表征会议论文集》，ICLR，2025年。2、9[BB19] Eric Blais 和 Joshua Brody。随机化查询复杂度的最优分离与强直接和。载于《第 34 届计算复杂性会议论文集》，CCC，第 29:1–29:17 页，2019年。3[BCS13] Peter Bürgisser、Michael Clausen 和 Mohammad A Shokrollahi。《代数复杂性理论》，第 315 卷。Springer Science & Business Media，2013年。4[BI15] Arturs Backurs 和 Piotr Indyk。编辑距离无法在强次二次时间内计算（除非 SETH 不成立）。载于《第 47 届 ACM 计算理论研讨会年会论文集》，第 51–58 页，2015。4[Blä13] Markus Bläser。快速矩阵乘法。《Theory Comput.》，5:1–60，2013年。8、22[BMR+20] Tom B. Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell、Sandhini Agarwal、Ariel Herbert-Voss、Gretchen Krueger、Tom Henighan、Rewon Child、Aditya Ramesh、Daniel M. Ziegler、Jeffrey Wu、Clemens Winter、Christopher Hesse、Mark Chen、Eric Sigler、Mateusz Litwin、Scott Gray、Benjamin Chess、Jack Clark、Christopher Berner、Sam McCandlish、Alec Radford、Il ya Sutskever 和 Dario Amodei。语言模型是少样本学习器。载于《神经信息处理系统大会》，NeurIPS，2020年。9[Bon22] Édouard Bonnet。稀疏无向无权图直径的 4 对 7 近似在时间 n^(4/3) 下是 SETH 困难的。《ACM算法汇刊》（TALG），18(2):1–14，2022年。4[BRS+18] Arturs Backurs、Liam Roditty、Gilad Segal、Virginia Vassilevska Williams 和 Nicole Wein。迈向图直径与偏心率的紧近似界。载于《第 50 届 ACM SIGACT 计算理论研讨会年会论文集》，第 267–280 页，2018年。4[BS83] Walter Baur 和 Volker Strassen。偏导数的复杂度。《Theor. Comput. Sci.》，22:317–330，1983年。7、21[Bsh89] Nader H Bshouty。关于扩展直接和猜想。载于《第 21 届 ACM 计算理论研讨会年会论文集》，第 177–185 页，1989年。3

[CDW+21] Beidi Chen、Tri Dao、Eric Winsor、Zhao Song、Atri Rudra 和 Christopher Ré。Scat- terbrain：统一稀疏与低秩注意力。载于《神经信息处理系统进展》（NeurIPS），2021。9[CGS15] Marek Cygan、Harold N Gabow 和 Piotr Sankowski。Baur-Strassen定理的算法应用：最短环、直径与匹配。《ACM学报》（JACM），62(4):1–30，2015。7[CLD+21] Krzysztof Marcin Choromanski、Valerii Likhosherstov、David Dohan、Xingyou Song、Andreea Gane、Tamás Sarlós、Peter Hawkins、Jared Quincy Davis、Afroz Mohiuddin、Lukasz Kaiser、David Benjamin Belanger、Lucy J. Colwell 和 Adrian Weller。重新思考 Performer 的注意力机制。载于《国际学习表征会议》（ICLR），2021。2, 9[CND+23] Aakanksha Chowdhery、Sharan Narang、Jacob Devlin、Maarten Bosma、Gaurav Mishra、Adam Roberts、Paul Barham、Hyung Won Chung、Charles Sutton、Sebastian Gehrmann、Parker Schuh、Kensen Shi、Sasha Tsvyashchenko、Joshua Maynez、Abhishek Rao、Parker Barnes、Yi Tay、Noam Shazeer、Vinodkumar Prabhakaran、Emily Reif、Nan Du、Ben Hutchinson、Reiner Pope、James Bradbury、Jacob Austin、Michael Isard、Guy Gur-Ari、Pengcheng Yin、Toju Duke、Anselm Levskaya、Sanjay Ghemawat、Sunipa Dev、Henryk Michalewski、Xavier Garcia、Vedant Misra、Kevin Robinson、Liam Fedus、Denny Zhou、Daphne Ippolito、David Luan、Hyeontaek Lim、Barret Zoph、Alexander Spiridonov、Ryan Sepassi、David Dohan、Shivani Agrawal、Mark Omernick、Andrew M. Dai、Thanumalayan Sankaranarayana Pillai、Marie Pellat、Aitor Lewkowycz、Erica Moreira、Rewon Child、Oleksandr Polozov、Katherine Lee、Zongwei Zhou、Xuezhi Wang、Brennan Saeta、Mark Diaz、Orhan Firat、Michele Catasta、Jason Wei、Kathy Meier-Hellstern、Douglas Eck、Jeff Dean、Slav Petrov 和 Noah Fiedel。PaLM：借助 Pathways 扩展语言建模。《机器学习研究期刊》，24:240:1–240:113，2023。36[CPW25] Lijie Chen、Binghui Peng 和 Hongxun Wu。多层Transformer的理论局限。载于第66届IEEE计算机科学基础年度研讨会（FOCS）论文集，页码待定，2025。3, 9[DBK+21] Alexey Dosovitskiy、Lucas Beyer、Alexander Kolesnikov、Dirk Weissenborn、Xiaohua Zhai、Thomas Unterthiner、Mostafa Dehghani、Matthias Minderer、Georg Heigold、Sylvain Gelly、Jakob Uszkoreit 和 Neil Houlsby。一张图像胜过16×16个词：面向大规模图像识别的Transformer。载于第9届《国际学习表征会议》（ICLR）论文集，2021。2[DFE+22] Tri Dao、Dan Fu、Stefano Ermon、Atri Rudra 和 Christopher Ré。FlashAttention：具备IO感知能力的快速、节省内存的精确注意力。《神经信息处理系统进展》，35:16344–16359，2022。9[DLV25] Mina Dalirrooyfard、Ray Li 和 Virginia Vassilevska Williams。近似直径的困难性：无向图的新进展。《ACM学报》，72(1):1–32，2025。4[FEN+23] Daniel Y. Fu、Elliot L. Epstein、Eric Nguyen、Armin W. Thomas、Michael Zhang、Tri Dao、Atri Rudra 和 Christopher Ré。用于序列建模的简洁高效硬件长卷积。载于2023年《国际机器学习会议》（ICML）论文集，页码10373–10391，2023。9

[Fid72] Charles M. Fiduccia。通过除法算法进行多项式求值：重新审视快速傅里叶变换。载于第4届ACM计算理论研讨会年度会议论文集，页码88–93，1972。3[Fis24] Nick Fischer。通过Baur-Strassen定理实现确定性稀疏模式匹配。载于2024年ACM-SIAM离散算法年度研讨会（SODA）论文集，页码3333–3353。SIAM，2024。7[FKNN95] Tomás Feder、Eyal Kushilevitz、Moni Naor 和 Noam Nisan。摊销通信复杂度。《SIAM计算杂志》，24(4):736–750，1995。3[GF81] Giulia Galbiati 和 Michael J Fischer。关于双输出布尔网络的复杂度。《理论计算机科学》，16(2):177–185，1981。3[GHS+26] Shreya Gupta、Boyang Huang、Barna Saha、Yinzhan Xu 和 Christopher Ye。任意温度下注意力机制的次二次时间算法与困难性。载于第14届《国际学习表征会议》（ICLR）论文集，2026。2, 4, 9, 27[Hah20] Michael Hahn。神经序列模型中自注意力的理论局限。《计算语言学协会汇刊》，8:156–171，2020。9[HHG+20] John Hewitt、Michael Hahn、Surya Ganguli、Percy Liang 和 Christopher D. Manning。循环神经网络能够以最优记忆生成有界层级语言。载于2020年自然语言处理实证方法会议（EMNLP）论文集，页码1978–2010，2020。9[HJK+24] Insu Han、Rajesh Jayaram、Amin Karbasi、Vahab Mirrokni、David Woodruff 和 Amir Zandieh。HyperAttention：近线性时间的长上下文注意力。载于第十二届《国际学习表征会议》，2024。9[IP01] Russell Impagliazzo 和 Ramamohan Paturi。关于k-SAT的复杂度。《计算机与系统科学杂志》，62(2):367–375，2001。4, 14[JBKM24] Samy Jelassi、David Brandfonbrener、Sham M. Kakade 和 Eran Malach。跟我重复：Transformer在复制任务上优于状态空间模型。载于第41届《国际机器学习会议》（ICML）论文集，2024。9[JSLC25] Selim Jerad、Anej Svete、Jiaoda Li 和 Ryan Cotterell。唯一硬注意力：两面之说。载于第63届计算语言学协会年会论文集（第2卷：短文），ACL，页码977–996，2025。11[KKL20] Nikita Kitaev、Lukasz Kaiser 和 Anselm Levskaya。Reformer：高效的Transformer。载于《国际学习表征会议》（ICLR），2020。2, 9[KMR+23] Alexander Kirillov、Eric Mintun、Nikhila Ravi、Hanzi Mao、Chloe Rolland、Laura Gustafson、Tete Xiao、Spencer Whitehead、Alexander C Berg、Wan-Yen Lo 等。分割一切。载于IEEE/CVF国际计算机视觉会议论文集，页码4015–4026，2023。2[KMZ24] Praneeth Kacham、Vahab Mirrokni 和 Peilin Zhong。Polysketchformer：通过多项式核草图化实现快速Transformer。载于第41届《国际机器学习会议》，2024。9

[KUJ+25] Alexander Kozachinskiy、Felipe Urrutia、Hector Jimenez、Tomasz Steifer、Germán Pizarro、Matías Fuentes、Francisco Meza、Cristian B. Calderon 与 Cristóbal Rojas。Transformer 中的 Strassen 注意力、分裂 VC 维数与组合性，2025年。3[KVPF20] Angelos Katharopoulos、Apoorv Vyas、Nikolaos Pappas 与 François Fleuret。Transformer 是循环神经网络：具有线性注意力的快速自回归 Transformer。见国际机器学习会议，ICML，2020年。9[KWH23] Feyza Duman Keles、Pruthuvi Mahesakya Wijewardena 与 Chinmay Hegde。论自注意力的计算复杂度。见国际算法学习理论会议，第 597–619 页。PMLR，2023年。2, 9, 27[Li21] Ray Li。解决强指数时间假设与近似稀疏有向无权直径之争（直至 (nu) nseth）。见第53届年度 ACM SIGACT 计算理论研讨会论文集，第1684–1696页，2021。4[LKM23] Yaniv Leviathan、Matan Kalman 与 Yossi Matias。通过推测解码实现 Transformer 的快速推理。见国际机器学习会议，第19274–19286页。PMLR，2023年。9[LYS+25] Jingwen Liu、Hantao Yu、Clayton Sanford、Alexandr Andoni 与 Daniel Hsu。快速注意力机制：并行性的故事。见第39届年度神经信息处理系统大会，2025年。2[Mor85] Jacques Morgenstern。如何快速计算一个函数及其所有导数：Baur-Strassen 定理的一个变体。SIGACT News，16(4):60–62，1985年。21[MS23] William Merrill 与 Ashish Sabharwal。并行性权衡：对数精度 Transformer 的局限性。《计算语言学协会汇刊》，11:531–545，2023年。9[MS25] Simon Mackenzie 与 Abdallah Saffidine。反驳确定性通信复杂度中全函数的直和猜想。见第57届年度 ACM 计算理论研讨会论文集，第572–583页，2025年。3[MSS22] William Merrill、Ashish Sabharwal 与 Noah A. Smith。饱和 Transformer 是常数深度阈值电路。《计算语言学协会汇刊》，10:843–856，2022年8月。11[MWY13] Marco Molinaro、David P Woodruff 与 Grigory Yaroslavtsev。在通信复杂度中突破直和定理及其对草图化的启示。见第24届年度 ACM-SIAM 离散算法研讨会论文集，第1738–1756页。SIAM，2013年。3[PNP24] Binghui Peng、Srini Narayanan 与 Christos Papadimitriou。论 Transformer 架构的局限性。见第1届语言建模会议论文集，2024年。9[RHW86] David E. Rumelhart、Geoffrey E. Hinton 与 Ronald J. Williams。通过误差反向传播学习表征。《自然》，323(6088):533–536，1986年。7

[RV13] Liam Roditty 与 Virginia Vassilevska Williams。稀疏图直径与半径的快速近似算法。见第45届年度ACM计算理论研讨会论文集，第515–524页，2013年。4[RWC+19] Alec Radford、Jeffrey Wu、Rewon Child、David Luan、Dario Amodei、Ilya Sutskever 等。语言模型是无监督多任务学习器。OpenAI 博客，1(8):9，2019。2[Sch81] Arnold Schönhage。部分与完全矩阵乘法。SIAM J. Comput.，10(3):434–455，1981年。22[Sha20] Noam Shazeer。GLU变体改进 Transformer。arXiv 预印本 arXiv:2002.05202，2020年。36[SHT23] Clayton Sanford、Daniel J. Hsu 与 Matus Telgarsky。Transformer 的表征能力与局限性。见《神经信息处理系统进展》第36卷，NeurIPS，2023年。3, 9, 12[SHT24a] Clayton Sanford、Daniel Hsu 与 Matus Telgarsky。单层Transformer 无法解决归纳头任务。CoRR，abs/2408.14332，2024年。12[SHT24b] Clayton Sanford、Daniel Hsu 与 Matus Telgarsky。Transformer、并行计算与对数深度。见第41届国际机器学习会议论文集，ICML，2024年。3, 6, 9, 11, 12, 13[Str73] Volker Strassen。避免除法。《纯粹与应用数学杂志》，264:184–202，1973年。8, 18, 22[SY24] Barna Saha 与 Christopher Ye。注意力的输入/输出复杂度，或 FlashAttention 有多最优?见第41届国际机器学习会议论文集，ICML，2024年。9[Tea24] Llama团队。Llama 3 模型群。CoRR，abs/2407.21783，2024年。36[TMH+24] Gemma团队、Thomas Mesnard、Cassidy Hardin、Robert Dadashi、Surya Bhupatiraju、Shreya Pathak、Laurent Sifre、Morgane Rivière、Mihir Sanjay Kale、Juliette Love 等。Gemma：基于 Gemini 研究与技术的开放模型。arXiv 预印本 arXiv:2403.08295，2024年。36[VSP+17] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N. Gomez、Lukasz Kaiser 与 Illia Polosukhin。注意力就是你所需要的一切。见神经信息处理系统 NeurIPS，2017年。2, 44[VW18] Virginia Vassilevska Williams 与 R Ryan Williams。路径、矩阵与三角形问题之间的次立方等价关系。《ACM学报》（JACM），65(5):1–38，2018年。3[WDL25] Kaiyue Wen、Xingyu Dang 与 Kaifeng Lyu。循环神经网络（尚）不是 Transformer：上下文内检索的关键瓶颈。见第13届国际学习表征会议论文集，ICLR，2025年。9[Wil04] Ryan Williams。最优约束满足的新算法及其启示。见第31届国际自动机、语言与编程研讨会（ICALP）论文集，2004年。4, 14

[YA25] Hantao Yu 和 Josh Alman。两头胜过一头：用小型 Transformer 模拟大型 Transformer。CoRR，abs/2506.12220，2025年。9[YPPN21] Shunyu Yao、Binghui Peng、Christos H. Papadimitriou 和 Karthik Narasimhan。自注意力网络能够处理有界层级语言。载于第59届计算语言学协会年会暨第11届自然语言处理国际联合会议论文集（ACL/IJCNLP），第3770–3785页，2021。9[YSBS+25] Gilad Yehudai、Clayton Sanford、Maya Bechler-Speicher、Orr Fischer、Ran Gilad-Bachrach 和 Amir Globerson。Transformer 在图任务上的深度—宽度权衡。载于第39届神经信息处理系统大会，2025年。4[ZGD+20a] Manzil Zaheer、Guru Guruganesh、Avinava Dubey、Joshua Ainslie、Chris Alberti、Santiago Ontanon、Philip Pham、Anirudh Ravula、Qifan Wang、Li Yang 和 Amr Ahmed。Big Bird：面向更长序列的 Transformer。载于第34届神经信息处理系统国际会议论文集，2020年。9[ZGD+20b] Manzil Zaheer、Guru Guruganesh、Kumar Avinava Dubey、Joshua Ainslie、Chris Alberti、Santiago Ontañón、Philip Pham、Anirudh Ravula、Qifan Wang、Li Yang 和 Amr Ahmed。Big Bird：面向更长序列的 Transformer。载于神经信息处理系统大会（NeurIPS），2020年。2[ZHDK23] Amir Zandieh、Insu Han、Majid Daliri 和 Amin Karbasi。KDEFormer：通过核密度估计加速 Transformer。载于国际机器学习会议，第40605–40623页。PMLR，2023年。9

# A 省略的证明

本节给出省略的证明。

**引理 2.9**。设 $f \in \mathrm{Att}_m^N$ 是一个自注意力单元，其嵌入函数为 $Q, K, V$，满足

1. $V(X) \in \{0,1,2\}^{n\times m}$，并且
2. 对于每个 $X \in \mathbb{R}^{N\times m}$ 和 $i \in [N]$:

$$A(X)_{i,i'} \le \max_{i''} A(X)_{i,i''} - 1,\ \forall i' \notin I_{\max}(A(X)_i),$$

其中 $A(X) = Q(X)K(X)^\top$。

那么存在一个注意力单元 $f' \in \mathrm{Att}_m^N$，满足

$$\left\| f'(X) - \mathrm{hardmax}(A(X))V(X) \right\|_\infty \le \frac{1}{\mathrm{poly}(N)}$$

对于任意小的逆多项式。

引理 2.9 的证明。取足够大的常数 $c \gg \log N$。定义嵌入映射$Q'(v) = c\cdot Q(v)$，将嵌入 $Q$ 缩放 $c$ 倍，并令 $K'(v) = K(v)$ $V'(v) = V(v)$ 与$f$中完全相同。注意，这些嵌入映射可以表示为权重矩阵，它们要么是 $m\times m$单位矩阵 $I_m$，要么是 $I_m$的标量倍数。固定一个指标 $i$，不失一般性地假设 $1 \in I_{\max}(A(X)_i)$。令 $A(X)_i$ 表示 $A(X)$的第 $i$行。那么，

$$\sum_{i'\notin I_{\max}(A(X)_i)} \exp(cA(X)_{ii'}) \le \frac{N}{\exp(c)} \exp(cA(X)_{i1}) = \frac{1}{\mathrm{poly}(N)} \exp(cA(X)_{i1}).$$

那么，对于 $i' \in I_{\max}(A(X)_i)$，注意到~~$|A_i| \ge$~~ $\mathrm{Softmax}(cA(X))_{ii'}$，因此

$$|\mathrm{softmax}(cA(X))_{ii'} - \mathrm{hardmax}(cA(X))_{ii'}| = \frac{1}{|I_{\max}(A(X)_i)|} - \frac{\exp(cA(X))_{i1}}{\sum_{i'}\exp(cA(X))_{ii'}}.$$

现在，我们可以给出一个上界

$$\sum_{i'} \exp(cA(X))_{ii'} \le |I_{\max}(A(X)_i| \exp(cA(X))_{i1} + \frac{1}{\mathrm{poly}(N)}$$

因此

$$|\mathrm{softmax}(cA(X))_{ii'} - \mathrm{hardmax}(cA(X))_{ii'}| \le \frac{1}{|I_{\max}(A(X)_i)|}\left(1 - \frac{1}{1+\frac{1}{\mathrm{poly}(N)}}\right) \le \frac{1}{\mathrm{poly}(N)}.$$

对于 $i' \notin I_{\max}(A(X)_i)$，注意到 $\mathrm{Hardmax}(cA(X))_{ii'} = 0$，因此

$$|\mathrm{softmax}(cA(X))_{ii'} - \mathrm{hardmax}(cA(X))_{ii'}| \le \frac{\exp(cA(X))_{ii'}}{|I_{\max}|\exp(cA(X))_{i1}} \le \frac{1}{\mathrm{poly}(N)}.$$

然后，对于 $V(X)$ 中一个固定的列，记作 $V(X)^j$，我们利用 Hölder 不等式得到

$$\begin{aligned}\left|f'(X)_{ij} - \mathrm{hardmax}(cA(X)_i)\cdot V(X)^j\right| &= \left|(\mathrm{softmax}(cA(X)_i) - \mathrm{hardmax}(cA(X)_i))\cdot V(X)^j\right| \\ &\le \|\mathrm{softmax}(cA(X)_i) - \mathrm{hardmax}(cA(X)_i)\|_1 \left\|V(X)^j\right\|_\infty \\ &\le \frac{1}{\mathrm{poly}(N)}.\end{aligned}$$

特别地，只需为 $c$ 选择更大的常数因子，输出中的~~每个~~元素都可以控制在任意小的逆多项式 $\frac{1}{\mathrm{poly}(N)}$ 范围内。 □

**引理 2.13**。 设 $0 < s_2 \ldots, s_k \leq 1$。设 $A_1, \ldots A_k \subset \{0,1\}^d$ 对于 $d = \Theta(\log n)$ 成立，其中 $|A_1| = n$ 且对于所有 $i > 1$，都有 $|A_i| = n^{s_i}$。那么，在 $k$-OV 假设下，任何在 $A_1, \ldots, A_k$ 上计算 $k$-OV 的算法都需要 $n^{s_1+\ldots+s_k-o(1)}$ 时间。特别地，在 $k$-OV 假设下，不平衡 $k$-OV 假设成立。

引理 *2.13* 的证明。 假设给定一个规模为 $n$、包含集合 $B_1, \ldots, B_k$ 的（平衡） $k$-OV 实例。对于 $2 \leq j \leq k$，将 $B_j$ 划分为 $t_j := O(n^{1-s_j})$ 个规模为 $n^{s_j}$ 的集合，记为 $B_j^{(1)}, \ldots, B^{(t_j)\,j}$。现在，对于每个 $k$-元组 $(i_1, \ldots, i_k) \in \prod_{j=1}^{k}[t_j]$，构造一个由集合 $B_1, B^{(i_2)\,2}, \ldots, B^{(i_k)\,k}$ 组成的不平衡 $k$-OV 实例。注意，这会生成 $O\left(\prod_{j=2}^{k} t_j\right)$ 个不平衡 $k$-OV 实例。假设存在一个算法，能在时间 $O(n^{s_1+\ldots+s_k-\varepsilon})$ 内计算不平衡 $k$-OV，其中 $\varepsilon > 0$ 为某个值。于是，我们可以在以下时间内计算上述所有实例

$$O\left(\prod_{j=2}^{k} t_j n^{1+s_2+\cdots+s_k-\varepsilon}\right) = O\left(n^{k-\varepsilon}\right).$$

由于原始 $k$-OV 实例中存在一个正交的 $k$ 元组，当且仅当其中一个不平衡 $k$-OV 实例包含一个正交的 $k$ 元组，因此这违背了 $k$-OV 假设。 □

## A.1 去归一化注意力

在考虑注意力时，通常可以忽略 Softmax 所需的归一化。为此，我们定义去归一化注意力：对于 $Q(X)K(X)^\top$ 的每一行 $v$，用 $\exp(v)$ 替代 $\mathrm{softmax}(v)$。

**定义 A.1**（去归一化注意力）。 去归一化注意力头 是一个映射 $g_{Q,\,K,\,V}: \mathbb{R}^{N \times d_{\text{in}}} \to \mathbb{R}^{N \times m}$，定义为

$$g_{Q,K,V}(X) = \exp(Q(X)K(X)^\top)V(X)$$

并由按行排列的查询、键和值嵌入 $Q$，$K$，$V$：$\mathbb{R}^{N \times d_{\text{in}}} \to \mathbb{R}^{N \times m}$ 参数化（例如，$Q(X) = (Q_1(X_1)\ldots\ldots, Q_N(X_N))$。

我们证明，在标准设置（即具有归一化）下，可以使用一个小型 Transformer 高效地计算去归一化注意力。遗憾的是，这通常并不意味着具有归一化的 Transformer 可以模拟没有归一化的 Transformer，因为多层感知机作用于每一层的输出（即对所有注意力头求和之后），而下面的构造要求将多层感知机直接作用于每个注意力头的输出。

为简单起见，我们描述如下 Transformer

**引理 A.2.** 设 $g_{Q,K,V}$ 为一个去归一化注意力头，其输入长度为 $N$，输入维度为 $d_{\text{in}}$，嵌入维度为 $m$。存在一个 *Transformer* $T$，其输入长度为 $N+1$，嵌入维度为 $m + d_{\text{in}} + 1$，1 层，1 个注意力头，输入维度为 $d_{\text{in}}$，输出维度为 $m$，使得

$$T\left(\begin{pmatrix} X \\ 0 \end{pmatrix}\right) = \begin{pmatrix} g_{Q,K,V}(X) \\ 1_{1\times N} V(X) \end{pmatrix}$$

对所有 $X \in \mathbb{R}^{N \times d_{\text{in}}}$ 均成立。此外，所有嵌入和多层感知机映射都可以通过规模为 $O(Nm)$ 的算术电路计算。

证明。 考虑以下嵌入：

$$Q'(X) = \begin{pmatrix} Q(X) & 0_{N\times(d_{\text{in}}+1)} \\ 0_{1\times m} & 0_{1\times(d_{\text{in}}+1)} \end{pmatrix},$$
$$K'(X) = \begin{pmatrix} K(X) & 0_{N\times(d_{\text{in}}+1)} \\ 0_{1\times m} & 0_{1\times(d_{\text{in}}+1)} \end{pmatrix},$$
$$V'(X) = \begin{pmatrix} 0_{N\times d_{\text{in}}} & V(X) & 0_{N\times 1} \\ 0_{1\times d_{\text{in}}} & 0_{1\times m} & 1 \end{pmatrix}$$

其中 $Q'(X)$ $K'(X)$ $V'(X) \in \mathbb{R}^{(N+1)\times(m+d_{\text{in}}+1)}$。然后

$$\exp(Q'(X)K'(X)^\top) = \begin{pmatrix} \exp(Q(X)K(X)^\top) & 1_{N\times 1} \\ 1_{1\times N} & 1 \end{pmatrix}.$$

具体而言，前 $N$ 行的行和为 $S_i + 1$，其中 $S_i$ 是去归一化注意力头中第 $i$ 行的行和。注意 $S_i \geq 0$，因此分母始终为正。因此，在输出矩阵中，前 $N$ 行由输入 $X^{(0)} = \begin{pmatrix} X \\ 0 \end{pmatrix}$ 构成。接着，对于 $i \in [N]$，第 $(i, d_{\text{in}} + m + 1)$ 个元素为 $\frac{1}{1+S_i}$；而对于 $j \in [m]$，第 $(i, d_{\text{in}} + j)$ 个元素为 $\frac{Y_{ij}}{1+S_i}$，其中 $Y_{ij}$ 是去归一化注意力头 $g$ 的第 $(i, j)$ 个元素。对于 $i = N+1$，注意第 $(N+1, d_{\text{in}}+m+1)$ 个元素为 $\frac{1}{N+1}$；而对于 $j \in [m]$，第 $(N+1, j)$ 个元素为 $\frac{\|V_{(X\ [j]\|_1)}}{N+1}$。具体而言，我们得到向量 $1_{1\times N}V(X)$，并将其缩放 $\frac{1}{N+1}$。

然后，应用由 $\psi_i$：$v \mapsto v_{d_{\text{in}}+i}/v_{d_{\text{in}}+m+1}$ 定义的输出多层感知机 $\psi = (\psi_1, \ldots, \psi_m)$：$\mathbb{R}^{d_{\text{in}}+m+1} \to \mathbb{R}^m$，即可得到所需的输出 $Y$。这里请注意，$v_{d_{\text{in}}+m+1} = \frac{1}{1+S_i} > 0$，因为 $S_i > 0$。注意，每个嵌入映射和多层感知机都可以用规模为 $O(m)$ 的 eAC 实现。 □

## B 多层感知机

在标准 Transformer 架构中，多层感知机应用于注意力层之后。在我们的情形中，$X = \sum_{h=1}^{H} \text{Att}_i(X')$ 是对应层中各注意力头输出的总和。多层感知机是应用于输入 $X$ 的函数。

**GLU 多层感知机。** 在现代语言模型中，门控线性单元（GLU）是一种广受欢迎的激活函数 [Sha 20], 例如 PaLM [CND+23]、LLaMA [Tea24], 和 Gemma [TMH+24]。我们将证明，不失一般性，带有 GLU 多层感知机的 Transformer 可以模拟不带多层感知机的 Transformer，从而得到带有 GLU 多层感知机的 Transformer 的下界。下面定义门控线性单元。

**定义 B.1**（门控线性单元（GLU）多层感知机）。 给定一个激活函数 $\sigma$：$\mathbb{R} \to \mathbb{R}$，一个隐藏维度为 $d_{\text{hid}}$、激活函数为 $\sigma$ 的 GLU 是一个映射 $\psi$：$\mathbb{R}^m \to \mathbb{R}^m$，定义为

$$x \mapsto \left(\begin{pmatrix} x^\top & 1 \end{pmatrix} W_0 \odot \sigma\left(\begin{pmatrix} x^\top & 1 \end{pmatrix} W_1\right)\right) \quad 1\Big) W_2$$

其中 $W_0$, $W_1 \in \mathbb{R}^{(m+1)\times d_{\text{hid}}}$, $W_2 \in \mathbb{R}^{(d_{\text{hid}}+1)\times m}$，并且 $\sigma$ 逐元素应用。

我们证明，只要激活函数 $\sigma$ 不恒等于 0，带有 GLU 多层感知机的 Transformer 就能模拟不带多层感知机的 Transformer。特别地，我们关于定理 3.1和定理 5.1 的下界对带有 GLU 多层感知机的 Transformer 也成立。

**命题 B.2**。 假设存在某个 $c \in \mathbb{R}$ 满足 $\sigma(c) \neq 0$。

设 $T$ 是一个 ***Transformer***，其输入长度为 $N$，嵌入维度为 $m$，包含 $L$ 层、 $H$ 个注意力头，输入维度为 $d_{\text{in}}$，输出维度为 $d_{\text{out}}$，且不含多层感知机。

那么，存在一个 ***Transformer*** $T'$，其输入长度为 $N$，嵌入维度为 $m$，包含 $L$ 层、$H$ 个注意力头，输入维度为 $d_{\text{in}}$，输出维度为 $d_{\text{out}}$，并带有 ***GLU*** 多层感知机，其 $d_{\text{hid}} = m$ 且激活函数为 $\sigma$，使得 $T'(X) = T(X)$ 对所有输入 $X$均成立。

此外，如果 $\sigma$ 可以由规模为 $t$ 的eAC 计算得到，那么 ***GLU*** 多层感知机可以由规模为 $O(tm)$ 的eAC 计算得到。

证明。 设 $\sigma(c) = z \neq 0$。我们构造一个计算恒等映射的 GLU 多层感知机。令 $d_{\text{hid}} = m$，并且

$$W_0 = \begin{pmatrix} z^{-1} \cdot I_m \\ 0_{1\times m} \end{pmatrix}, \quad W_1 = \begin{pmatrix} 0_{m\times m} \\ c \cdot 1_{1\times m} \end{pmatrix}, \quad W_2 = \begin{pmatrix} I_m \\ 0_{1\times m} \end{pmatrix}.$$

那么，对于任意 $x$，GLU 多层感知机映射

$$x \mapsto \frac{x}{z} \odot (\sigma(c)1_m) = x$$

因此，GLU 多层感知机 $\psi$ 是恒等映射。 □

在本节余下部分，我们将证明我们的下界对传统（非门控）多层感知机同样成立。传统上，给定输入 $X \in \mathbb{R}^{N\times m}$，多层感知机层会计算 $f(XW_1)W_2$其中 $W_1$，$W_2 \in \mathbb{R}^{m\times d_{\text{hid}}}$ 是可学习的权重矩阵，而 $f$ 是逐元素激活函数。一般而言，当我们乘以 $W_1$，$W_2$时，可以假设输入矩阵附加了一列 1，以便将偏置项添加到 $W_1$，$W_2$所计算的线性函数中。根据上下文，若矩阵 $X \in \mathbb{R}^{N\times m}$，我们会滥用记号并令

$$XW := \begin{pmatrix} X & 1_{N\times 1} \end{pmatrix} W$$

其中 $W \in \mathbb{R}^{(m+1)\times d_{\text{hid}}}$。

**定义 B.3**（多层感知机）。 给定激活函数 $\sigma : \mathbb{R} \to \mathbb{R}$，隐藏维度为$d_{\text{hid}}$ 且激活函数为 $\sigma$ 的多层感知机是一个映射 $\psi : \mathbb{R}^m \to \mathbb{R}^m$，定义为

$$x \mapsto \begin{pmatrix} \sigma\left(\begin{pmatrix} x^\top & 1 \end{pmatrix} W_1\right) & 1 \end{pmatrix} W_2$$

其中 $W_1 \in \mathbb{R}^{(m+1)\times d_{\text{hid}}}$, $W_2 \in \mathbb{R}^{(d_{\text{hid}}+1)\times m}$，且 $\sigma$ 按元素应用。

具体而言，我们考虑两种标准的激活函数选择：ReLU 和 Sigmoid。

**ReLU.** 对于 ReLU 激活函数 $x \mapsto \max(0, x)$，我们断言，带有 ReLU 多层感知机层的 Transformer 可以模拟不含多层感知机的 Transformer。于是，我们的下界意味着，存在带有 ReLU 多层感知机层的 Transformer，对任意 $c > 0$都无法用规模为 $LHN^{\omega-c}$ 的 eAC 计算。然而在这种情况下，尚不清楚如何用 eAC 高效地模拟 ReLU 函数，因此可能需要规模更大的 eAC。

**引理 B.4.** 设 $T$ 为一个 ***Transformer***，其输入长度为 $N$，嵌入维度为 $m$，有 $L$ 层、 $H$个注意力头，输入维度为 $d_{\text{in}}$，输出维度为 $d_{\text{out}}$，且不含多层感知机。

那么，存在一个 ***Transformer*** $T'$，其输入长度为 $N$、嵌入维度为 $m$、包含 $L$ 层和 $H$ 个注意力头、输入维度为 $d_{\text{in}}$、输出维度为 $d_{\text{out}}$，并带有具有 $d_{\text{hid}} = 2m$ 和 ***ReLU*** 激活函数的多层感知机，使得 $T'(X) = T(X)$ 对所有输入 $X$ 均成立。

证明。 给定 $X \in \mathbb{R}^{N\times m}$，我们说明如何使用带有 ReLU 激活函数的多层感知机来计算单位映射。令

$$W_1 = \begin{pmatrix} I_m & -I_m \\ 0_{1\times m} & 0_{1\times m} \end{pmatrix}, \quad W_2 = \begin{pmatrix} I_m \\ -I_m \\ 0_{1\times m} \end{pmatrix}.$$

现在，令 $f(x) = \max(x, 0)$ 表示 ReLU 函数，注意到

$$f(XW_1)W_2 = \begin{pmatrix} f(X) & f(-X) \end{pmatrix} W_2 = f(X) - f(-X).$$

我们通过注意到以下事实得出证明： $f(x) - f(-x) = \max(0, x) - \max(0, -x) = x$ 对所有 $x \in \mathbb{R}$ 均成立。 □

**Sigmoid**。 对于 Sigmoid 激活函数（记作 $\sigma(x) = 1/(1 + e^{-x})$），我们需要对定理 3.1 和定理 5.1进行更精细的修改。事实上，我们将针对一类更一般的激活函数证明这些结果。

**定义 B.5**。 若激活函数 $\sigma : \mathbb{R} \to \mathbb{R}$ 满足以下条件，则称其为间隙保持的：

1. $\sigma(0) \neq \sigma(1)$,
2. 存在 $c^*$，使得对于所有 $x \leq c^* - 1$, $\sigma(x) \leq \sigma(c^* - 1) < \sigma(c^*)$。

注意，Sigmoid 是间隙保持的，且 $c^* = 0$。我们指出，常数的具体选择（例如 0、1、 −1、 −.1）并不十分重要，证明可以针对其他常数选择进行修改。我们证明，小嵌入维度下的下界（定理 3.1）对于具有间隙保持激活函数的多层感知机成立。

**命题 B.6**。 令 $m = \Theta(\log N)$，且 $L$、 $H = \mathrm{poly}(N)$。在3-OV 假设下，任何一种计算输入长度为 $N$、嵌入维度为 $m$、具有 $L$ 层且每层有 $H$ 个注意力头、输入维度为 $m$、输出维度为1的 *Transformer*，并达到 $\frac{1}{10N}$ 的逐项加性误差的算法，都需要 $LHN^{2-o(1)}$ 时间。即使 *Transformer* 具有隐藏维度为 $O(m)$ 且使用间隙保持激活函数的多层感知机，该下界仍然成立。

我们概要介绍这些修改。回顾定理 3.1，我们构造了一个 Transformer，使得每一层的输出 (1) 在 $m - 1$ 列上保留其输入；(2) 保留一列，用于记录是否已找到正交三元组。在最后一列中，若 $a_i$ 迄今尚未参与任何正交三元组，则其值为 $2H\ell$；否则至多为 $2H\ell - 0.5$，从而使是实例与否实例之间的加性间隔达到 0.5。因此，我们设计了一个多层感知机，在 $m - 1$ 列上计算恒等映射，并在最后一列保持间隔。为计算恒等映射，我们注意到前 $m - 1$ 列中的每一列至多有 2 个不同的值，因此可以设计适当的线性函数，将 Sigmoid $\sigma$ 的输出映射回输入。在最后一列中，我们构造 $W_1$ 来映射 $x \mapsto x - 2H$，并构造 $W_2$ 来映射 $\sigma(0) \mapsto 0$ 和$\sigma(-1/2) \mapsto y < -0.5$，从而使输出中的间隔至少保持为 0.5。

证明。 我们将说明定理3.1所需的修改，从第一层开始。令 $X^{(0)}$ 表示 Transformer 的输入，$Z_{h,\ell}$ 表示定理 $h$中第 $\ell$层 hardmax Transformer 的第3.1个注意力头的输出。回顾

$$X^{(0)} = \begin{pmatrix} a_1^\top & b_1^\top & 1 \\ \vdots & \vdots & \vdots \\ a_N^\top & b_N^\top & 1 \\ 0 & 0 & 2 \end{pmatrix} \in \mathbb{R}^{(N+1)\times(m-1)}, \quad Z_{h,\ell} = \begin{pmatrix} 0 & \dots & 0 & z_{1,h,\ell} \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & z_{N,h,\ell} \\ 0 & \dots & 0 & \frac{N+2}{N+1} \end{pmatrix} \in \mathbb{R}^{(N+1)\times m}$$

其中 $z_{i,\ h,\ \ell}=2$ 若 $a_i$，$c_{h,\ \ell}$ 不构成正交三元组；否则为 $z_{i,\ h,\ \ell}\le\frac{3}{2}$。于是，

$$\sum_h Z_{h,\ell}=\begin{pmatrix}0 & \dots & 0 & z_{1,\ell}\\ \vdots & \ddots & \vdots & \vdots\\ 0 & \dots & 0 & z_{N,\ell}\\ 0 & \dots & 0 & \frac{H(N+2)}{N+1}\end{pmatrix}$$

其中 $z_{i,\ \ell}=\sum_h z_{i,\ h,\ \ell}$。

我们构造一个隐藏维度为 $d_{\text{hid}}=m$ 的多层感知机，其激活函数为间隙保持激活函数。定义线性函数 $f_1$，$f_2$，$f_3$，使得 $f_1$ 是一个递增线性函数，将 $\sigma(c^*)$ 映射为 $\mapsto 0$，并将 $\sigma(c^*-1)$ 映射为 $\mapsto -1$，且

$$\begin{aligned}f_1(x)&=b_1x+b_2\\ f_2(x)&=\frac{x-\sigma(0)}{\sigma(1)-\sigma(0)}=c_1x+c_2\\ f_3(x)&=1+f_2(x)=c_1x+(1+c_2)\end{aligned}$$

对于某些常数 $b_1$，$b_2$，$c_1$，$c_2\in\mathbb{R}$，并注意到

$$\begin{aligned}&f_1:\sigma(c^*)\mapsto 0,\quad \sigma(c^*-1)\mapsto -1\\ &f_2:\sigma(0)\mapsto 0,\quad \sigma(1)\mapsto 1\\ &f_3:\sigma(0)\mapsto 1,\quad \sigma(1)\mapsto 2.\end{aligned}$$

注意，根据定义 B.5的前两个性质，我们可以构造 $f_1$，$f_2$，$f_3$。将权重矩阵设为

$$W_1=\begin{pmatrix}I_{2d} & 0_{2d\times 1} & 0_{2d\times 1}\\ 0_{1\times 2d} & 1 & 0\\ 0_{1\times 2d} & 0 & 2\\ 0_{1\times 2d} & -1 & c^*-4H\end{pmatrix},\quad W_2=\begin{pmatrix}c_1I_{2d} & 0_{2d\times 1} & 0_{2d\times 1}\\ 0_{1\times 2d} & c_1 & 0\\ 0_{1\times 2d} & 0 & b_1\\ (c_2)_{1\times 2d} & 1+c_2 & b_2\end{pmatrix}.$$

注意，$W_1$，$W_2$ 有 $m+1$ 行（而不是 $m$行），因为最后一行编码了线性函数 $W_1$，$W_2$的偏置项。

由于 $X^{(0)}$ 的前 $2d$ 列为 0 或 1，我们确保多层感知机保留前 $2d$列，因为 $W_1$ 是恒等映射，且 $f_2$ 将 $\sigma(0)=\frac{1}{2}$ 映射为 0，并将 $\sigma(1)$ 映射为 1。

同样，由于第 $(2d+1)$ 列为 1 或 2，我们也确保多层感知机保留这一点。这里注意到，$W_1$ 分别将 1 和 2 映射为 0 和 1，而 $f_3$ 将 $\sigma(0)$ 映射为 1，并将$\sigma(1)$ 映射为 2。

最后，考虑第 $(2d+2)$ 列中的一个元素。和之前一样，如果 $a_i$ 不与任何 $\{c_{h,1}\}$ 构成正交三元组，那么 $z_{i,\ 1}=2H$；否则，$z_{i,\ 1}\le 2H-\frac{1}{2}$。那么，在前一种情况下，$\sigma(Y_{(1)}W_1)[i,m]=\sigma(c^*)$；在后一种情况下，$\sigma(x)\le\sigma(c^*-1)$。特别地，在前一种情况下，多层感知机层的输出为 $X^{(1)}[i,m]=(\sigma(Y_{(1)}W_1)W_2)[i,m]=0$；在后一种情况下，该输出至多为 $-1$（因为 $f_1$ 是递增的）。

对于后续层，我们可以进行归纳。注意，除第 $m$列外，上述相同的证明表明，输入矩阵 $X^{(0)}$ 的内容保持不变。只需证明第 $m$列能够帮助我们区分 3-OV 的是实例与否实例。我们使用以下归纳命题：对于所有 $i$，如果 $a_i$ 与某个 $c_{h,\ t}$构成正交三元组，其中 $h\in[H]$，$t\le\ell$，那么 $X^{(\ell)}[i,m]\le -1$。否则，$X^{(\ell)}[i,m]=0$。上文中，我们已通过 $\ell=1$ 建立了基础情形。

令 $Y^{(\ell)}$ 表示注意力层的输出，我们有

$$Y^{(\ell)}[i,m] = X^{(\ell-1)}[i,m] + \sum_h z_{i,h,\ell}.$$

在否实例中，我们有 $X^{(\ell-1)}[i,m] = 0$（根据归纳法）以及 $\sum_h z_{i,\ h,\ \ell} = 2H$。按照基础情形的证明，我们得出结论 $X^{(\ell)}[i,m] = 0$。

现在我们说明，在是实例中，只要 $a_i$ 与某个 $c_{h,\ t}$ 针对 $t \le \ell$, $X^{(\ell)}[i,m] \le -1$构成一个正交三元组。根据归纳法，如果 $t < \ell$，那么我们有 $X^{(\ell-1)}[i,m] \le -1$；否则我们有 $\sum_h z_{i,\ h,\ \ell} \le 2H - 0.5$。由于在两种情况下都有 $X^{(\ell-1)}[i,m] \le 0$ 和 $\sum_h z_{i,\ h,\ \ell} \le 2H$，我们有$Y^{(\ell)}[i,m] \le 2H - 0.5$。由于 $W_1$ 对第 $m$ 列应用递增函数，我们再次得到，否实例中输入 $\sigma$ 为 $c^*$，而是实例中至多为 $c^* - 1$。接着，我们按照基础情形中的相同方式完成证明。

因此，按上述方式对多层感知机进行编码后，只需检查 Transformer 输出的第 $m$ 列即可计算 3-OV。对于否实例，所有输出元素都应为 0。对于是实例，至少有一个元素会小于 $-1$。

□

现在我们转向定理 5.1，我们将针对一类不同的激活函数证明该定理（Sigmoid 也满足这一性质）。

**定义 B.7.** 若激活函数 $\sigma : \mathbb{R} \to \mathbb{R}$ 满足以下条件，则称其为 eAC 梯度高效的：

1. $\sigma(0) \neq \sigma(1)$,
2. $\sigma^{-1}$ 可由规模为 $O(1)$ 的 eAC 实现。
3. $\sigma'(x) = h(\sigma(x))$，对于某个 $h$，使得 $\sigma$，$h$ 可由规模为 $O(1)$ 的 eAC 实现 .

注意，Sigmoid 函数是 eAC 梯度高效的，因为 $\sigma'(x) = \sigma(x)(1 - \sigma(x))$，并且 $\sigma^{-1}$ 可在 Sigmoid 函数的取值范围 (0，1) 上计算。我们指出，如果最后一层没有多层感知机，则不需要满足第二个条件。

**命题 B.8.** 存在一个 *Transformer* $T$，其输入长度为 $N+1$，嵌入维度为$m = 2N+1$，共有 $L$ 层，每层有 $H$ 个注意力头，输入维度为 1，输出维度为 $N+1$，并满足以下条件：

*1.* $T$ 可由一个eAC计算，且每个嵌入和多层感知机均可由规模为$O(m)$的eAC 计算。

*2.* 任何计算 $T$ 的eAC规模至少为 $LHN^{\omega - o(1)} - O\left(LHN^2\right)$。

即使 $T$ 具有使用 $d_{\text{hid}} = O(m)$ 且梯度高效的 eAC 激活函数的多层感知机，下界仍然成立。

证明。我们保留定理5.1中的构造，只修改多层感知机。与命题 B.6一样，每个多层感知机权重矩阵都额外增加一行，以便在线性函数中加入偏置项。我们设定

$$W_1 = \begin{pmatrix} I_{N+1} & 0_{(N+1)\times(m'-(N+1))} \\ 0_{(m'-(N+1))\times(N+1)} & I_{m'-(N+1)} \\ 0_{1\times(N+1)} & 0_{1\times(m'-(N+1))} \end{pmatrix}, W_2 = \begin{pmatrix} I_{N+1} & 0_{(N+1)\times(m'-(N+1))} \\ 0_{(m'-(N+1))\times(N+1)} & c_1 I_{m'-(N+1)} \\ 0_{1\times(N+1)} & -(c_2)_{1\times(m'-(N+1))} \end{pmatrix}$$

其中 $c_1$，$c_2$ 是命题 B.6 中定义的常数。注意，要定义 $c_1$，$c_2$，我们只需要性质 $\sigma(0) \neq \sigma(1)$。因此，多层感知机层将 $\sigma$ 逐元素应用于前 $N+1$ 列，并且

分别将最后 $m'-(N+1)$ 列中的 0，1 映射为 0，1。由于 $\sigma$ 可以由规模为 $O(1)$ 的 eAC 计算，因此每个多层感知机映射都可以由规模为 $O(m)$ 的 eAC 计算，满足第一条性质。

我们首先论证，引理 5.2 中所描述的注意力头的输出保持不变。由于引理 5.2 的保证只要最后 $m'-(N+1)$ 列与输入相同 $X^{(0)}$ 就成立，因此只需论证这一点在每一层之后仍然保持不变，以下命题将对此进行形式化。

**命题 B.9**。 在定理 *5.1* 所定义的 *Transformer* 中，对于以上由 $W_1, W_2$ 指定的多层感知机，每一层的输出 $\ell \geq 0$ 满足 $X^{(\ell)}[i,j] = X^{(0)}[i,j]$ 对所有 $i \in [N+1], j > N+1$ 即，输入在最后 $m'-N+1$ 列中得以保留。

证明。 我们进行归纳论证。基础情形显然成立，因此考虑归纳情形。根据引理 5.2，每个注意力头在前 $N+1$ 列之外的输出均为 $0$ ，因此每个注意力层在这些列上的输出恰好等于其输入。由于我们输入 Transformer 的内容记为 $X^{(0)}$，在这些列中仅包含 $0$、$1$ ，多层感知机会对这些值计算恒等映射，并保留前 $N+1$ 列之外的输入，从而证明归纳假设。

□

现在说明，我们可以从计算所述 Transformer 的 eAC 中高效地恢复矩阵乘积。令 $h$ 为满足以下条件的函数： $\sigma'(x) = h(\sigma(x))$。下文假设 $j \leq N+1$，因为我们只需要 Transformer 输出的前 $N+1$ 列中的元素来恢复矩阵乘积。与命题 B.6中一样，我们定义

$$Y^{(\ell)} := X^{(\ell-1)} + \sum_h Y^{(h,\ell)}$$

其中 $Y^{(h,\ell)}$ 表示 $\ell$层中第 $h$个注意力头的输出。回顾一下，我们在上文已经证明，$Y(h,\ell)$ 恰好是定理 5.1中构造的 Transformer 中对应注意力头的输出。

In 特别地，如果 $Y$ 表示 Transformer 的输出，则对于 $j \leq N+1$，我们有

$$\begin{aligned} Y_{i,j} &= \sigma\left(Y_{i,j}^{(L)}\right) \\ &= \sigma\left(X_{i,j}^{(L-1)} + \sum_h Y_{i,j}^{(h,L)}\right) \end{aligned}$$

并且对于所有 $\ell > 0$ 和 $j \leq N+1$,

$$\begin{aligned} X_{i,j}^{(\ell)} &= \sigma\left(Y_{i,j}^{(\ell)}\right) \\ &= \sigma\left(X_{i,j}^{(\ell-1)} + \sum_h Y_{i,j}^{(h,\ell)}\right). \end{aligned}$$

与不存在多层感知机时的情况相反，其中 $Y_{i,\,j} = \sum_{h,\,\ell} Y_{i\,(h,\,\ell)\,,\,j}$ 我们现在有

$$Y_{i,j} = \sigma\left(\sum_h Y_{i,j}^{(h,L)} + \sigma\left(\sum_h Y_{i,j}^{(h,L-1)} + \sigma\left(\dots\right)\right)\right). \tag{3}$$

回想一下，在定理 5.1 的证明中，我们使用 Transformer 的输出来计算以下两个函数：

$$f\left(\{A_k, B_k, C_k, D_k\}_{k=1}^{LH}\right) = \sum_{i=1}^{N} Y_{i,1}$$
$$g\left(\{A_k, B_k, C_k, D_k\}_{k=1}^{LH}\right) = \sum_{i=1}^{N} Y_{i,i+1}.$$

我们现在说明如何恢复 $LH$ 独立矩阵乘积 $A_k B_k^\top$，从而完成证明。下面的归纳引理对此进行了形式化。

引理 *B.10.* 回顾一下，我们将 $k \in [LH]$ 索引为$(h, \ell)$。对于所有 $0 \le \ell \le L$，都存在一个规模为$O(H(L-\ell)N^2)$的 *eAC*，它可以计算 $\left\{A_k B_k^\top\right\}$，适用于 $k = (h, p)$，其中 $p \ge \ell$，$\left\{\prod_{p=t}^{L} \sigma'\left(Y_{(p)\, i_0, 1}\right)\right\}_{t \in [\ell-1, L]}^{i_0}$，且$\left\{\prod_{p=t}^{L} \sigma'\left(Y_{(p)\, i_0, i_0+1}\right)\right\}_{t \in [\ell-1, L]}^{i_0}$。

*Proof.* 对于基础情形，固定 $\ell = L$。然后，对于固定的 $k = (h_0, L)$ 和 $i_0, j_0$，我们有

$$\begin{aligned}
\frac{\partial f}{\partial C_{ki_0j_0}} &= \frac{\partial}{\partial C_{ki_0j_0}} \sum_{i=1}^{N} Y_{i,1} \\
&= \frac{\partial}{\partial C_{ki_0j_0}} Y_{i_0,1} \\
&= \sigma'\left(Y_{i_0,1}^{(L)}\right) \frac{\partial Y_{i_0,1}^{(h_0,L)}}{\partial C_{ki_0j_0}}.
\end{aligned}$$

在第二个等式中，我们只保留依赖于 $C_{ki_0j_0}$ 的求和项；在第三个等式中，我们应用链式法则，并注意到在 $Y_{(L)\, i_0,1}$ 中，只有 $Y^{(h_0,\ell)}{}_{i_0,1}$ 依赖于 $C_{ki_0j_0}$。由于 $\sigma$ 是 eAC 梯度高效的，我们可以利用 Transformer 的输出计算 $\sigma'\left(Y_{(L)\, i_0,1}\right)$、$h\left(\sigma\left(Y_{(L)\, i_0,1}\right)\right) = h(Y_{i_0,1})$。对于所有的 $h \in [H]$ 和 $i_0 \in [N]$，我们使用规模为 $O(HN)$ 的 eAC。特别地，我们可以计算

$$\frac{\partial Y_{i_0,1}^{(h_0,L)}}{\partial C_{ki_0j_0}} = \frac{\frac{\partial f}{\partial C_{ki_0j_0}}}{h(Y_{i_0,1})}.$$

我们对 $g$ 也作类似的论证，如下所示。

$$\frac{\partial g}{\partial D_{ki_0}} = \frac{\partial}{\partial D_{ki_0}} Y_{i_0,i_0+1} = \sigma'(Y_{i_0,i_0+1}^{(L)}) \frac{\partial Y_{i_0,i_0+1}^{(h_0,L)}}{\partial D_{ki_0}}.$$

通过相同的证明，我们可以使用规模为 $O(HN)$ 的 eAC 恢复 $\frac{\partial Y^{(h_0,\, L)}{}_{i_0\, i_0+1}}{\partial D_{ki_0}}$。

由于在定理 5.1 中，我们展示了如何计算由 $\frac{\partial Y^{(h,\, \ell)}{}_{i_0 1}}{\partial C_{ki_0j_0}}$ 和 $\frac{\partial Y_{i_0}^{(h,\ell)}}{\partial D_{ki_0}}{}^{i_0+1}$ 得到的乘积，因此可以利用上述导数恢复矩阵乘积 $A_k B_k^\top$，其 eAC 规模为 $O(N^2)$。对所有 $H$ 重复上述论证，我们便得到一个 $O(HN^2)$ eAC，用于计算第 $L$ 层中的所有矩阵乘积。给定所得的矩阵乘积，我们可以通过对每个矩阵乘积的各行求和，利用规模为 $O(HN^2)$ 的 eAC 计算所有 $Y_{i_0\, 1}^{(h,L)}$。然后，由于

$$Y_{i_0,1} = \sigma\left(\sum_h Y_{i_0,1}^{(h,L)} + \sigma\left(Y_{i_0,1}^{(L-1)}\right)\right)$$

我们可以计算

$$\sigma'\left(Y_{i_0,1}^{(L-1)}\right)=h\left(\sigma\left(Y_{i_0,1}^{(L-1)}\right)\right)=h\left(\sigma^1(Y_{i_0,1})-\sum_h Y_{i_0,1}^{(h,L)}\right)$$

对于所有 $i_0$，其 eAC 规模为 $O(HN)$。请注意，这是唯一需要 $\sigma$ 可逆的步骤。特别地，我们已经描述了如何计算以下量：

$$\left\{\sigma'\left(Y_{i_0,1}^{(L-1)}\right)\sigma'\left(Y_{i_0,1}^{(L)}\right)\right\}_{i_0},\ \left\{A_kB_k^\top\right\}_{k=(h,L)}$$

使用规模为 $O(HN^2)$ 的 eAC，从而完成基础情形。

现在，考虑归纳情形 $\ell<L$。我们论证，所有额外所需的值都可以使用规模为 $O(HN^2)$ 的 eAC 计算得到。应用链式法则，对于 $k\leftarrow(h_0,\ \ell)$ $i_0,\ j_0$，我们有

$$\begin{aligned}\frac{\partial f}{\partial C_{ki_0j_0}}&=\frac{\partial}{\partial C_{ki_0j_0}}Y_{i_0,1}\\&=\left(\prod_{t=\ell}^{L}\sigma'\left(Y_{i_0,1}^{(t)}\right)\right)\frac{\partial Y_{i_0,1}^{(h_0,\ell)}}{\partial C_{ki_0j_0}}\end{aligned}$$

遵循与基础情形类似的逻辑。根据归纳假设，我们有 $\prod_{t=\ell}^{L}\sigma'\left(Y_{(t)\,i_0,\ 1}\right)$，它来自前一次（$\ell+1$）次迭代。因此，使用规模为 $O(HN)$ 的 eAC，我们可以对所有 $i_0$ 计算：

$$\frac{\partial Y_{i_0,1}^{(h_0,\ell)}}{\partial C_{ki_0j_0}}=\frac{\frac{\partial f}{\partial C_{ki_0j_0}}}{\prod_{t=\ell}^{L}\sigma'\left(Y_{i_0,1}^{(t)}\right)}.$$

从这里开始，我们再次遵循定理5.1 的证明来恢复

$$\{A_kB_k\}_{k=(h,\ell)}$$

并将矩阵行相加，以获得 $\left\{Y_{(\ell)\,i_0,\ 1}\right\}_{i_0}$，使用规模为 $O(HN^2)$ 的 eAC。然后，由于

$$Y_{i_0,1}^{(\ell)}=\sum_h Y_{i_0,1}^{(h,\ell)}+\sigma\left(Y_{i_0,1}^{(\ell-1)}\right)$$

我们可以计算

$$\sigma'\left(Y_{i_0,1}^{(\ell-1)}\right)=h\left(\sigma\left(Y_{i_0,1}^{(\ell-1)}\right)\right)=h\left(Y_{i_0,1}^{(\ell)}-\sum_h Y_{i_0,1}^{(h,\ell)}\right)$$

对于所有 $i_0$，使用规模为 $O(HN)$ 的 eAC。利用归纳假设，我们计算对于所有 $i_0$,

$$\prod_{t=\ell-1}^{L}\sigma'\left(Y_{i_0,1}^{(t)}\right)=\sigma'\left(Y_{i_0,1}^{(\ell-1)}\right)\prod_{t=\ell}^{L}\sigma'\left(Y_{i_0,1}^{(t)}\right)$$

使用规模为 $O(N)$ 的 eAC，从而完成归纳情形。对于 $g$ 关于 $D_{ki_0}$ 的偏导数，也可以采用类似的论证。 □

The ab 上述引理使我们能够完成证明，计算所有 $LH$ 矩阵乘积 . □

# C 注意力头聚合

考虑一个具有 $L$ 层、每层包含 $H$ 个注意力头且嵌入维度为 $m$ 的 Transformer。对 $\ell$层的注意力头输出进行聚合有两种标准方式，记为 $X^{(\ell)}$，给定输入$X^{(\ell-1)}$：求和与拼接。求和聚合在定义 2.6 中有所描述。在拼接聚合中（参见例如 [VSP+17]），每个注意力头将维度为 $m$的每一行映射到维度为 $m/H$。输出随后为

$$X^{(\ell)} = \Psi_\ell \left( X^{(\ell-1)} + \left( \mathrm{Att}_{Q_1,K_1,V_1}\left(X^{(\ell-1)}\right) \quad \mathrm{Att}_{Q_2,K_2,V_2}\left(X^{(\ell-1)}\right) \quad \dots \quad \mathrm{Att}_{Q_H,K_H,V_H}\left(X^{(\ell-1)}\right) \right) \right)$$

其中，$\Psi$ 表示拼接后应用的多层感知机。

我们证明，这样的 Transformer 可以模拟采用求和聚合且嵌入维度为 $m/H$的 Transformer，因此，针对采用求和聚合且嵌入维度为 $m/H$ 的 Transformer 的任何下界，也适用于采用拼接聚合且嵌入维度为 $m$ 的变换。

**命题 C.1.** 假设 $H \leq m$ 整除 $m$。设 $T$ 是一个输入长度为 $N$、嵌入维度为 $m/H$、具有 $L$层且每层有 $H$ 个注意力头的 *Transformer*，并采用求和聚合。

那么，存在一个 *Transformer* $T'$，其输入长度为 $N+1$、嵌入维度为 $m+H$、具有 $L$ 层且每层有$H$ 个注意力头，并采用拼接聚合；此外，存在一个线性函数 $f\colon \mathbb{R}^m \to \mathbb{R}^m$，使得对于所有输入 $X$，$f(T'(X)) = T(X)$。此外，以下结论成立：

1. 如果 $T$ 可由一个eAC计算，那么 $T'$ 也可由 *eAC* 计算。
2. 如果 $T$ 的每个嵌入映射都可由规模为 $O(m/H)$ 的算术电路计算，那么一层中 $T$ 的所有嵌入映射都可由规模为 $O(m)$ 的算术电路计算。
3. 如果 $T$ 没有多层感知机，那么 $T'$ 也没有。
4. 如果 $T$ 具有规模为 $d_{\text{hid}} = O(m/H)$ 且激活函数为 $\sigma$ 的多层感知机或 *GLU* 多层感知机，那么 $T'$ 具有规模为 $d_{\text{hid}} = O(m)$ 且激活函数为 $\sigma$ 的多层感知机（参见定义 *B.3* 和定义 *B.1*）。

特别地，定理 3.1 意味着，只要 $m/H = N^{o(1)}$，用于计算采用拼接聚合的 Transformer 的标准 $LHN^{2+o(1)}$时间算法就是最优的；而定理 5.1意味着，只要 $m/H = \Theta(N)$，相应的 $LHN^{\omega+o(1)}$时间算法就是最优的。此外，无论是在正文还是第 B 节中，我们对多层感知机所作的任何假设，在归约中都会得到保留。

证明。我们证明每一层都可以被模拟。令 $Y_{(\ell)}$ 为 $T$ 的第 $\ell$ 层的输出。此外，令 $Y^{(\ell,h)}$ 为求和之前 $T$ 的第 $\ell$ 层中第 $h$ 个注意力头的输出。根据定义，$Y_{(\ell)} = \Psi_\ell \left( Y^{(\ell-1)} + \sum_h Y_{(\ell,h)} \right)$ 对于 $\ell \geq 1$（如果 $T$没有多层感知机，则 $\Psi_\ell$ 只是恒等映射）。考虑 $T'$ 的一层，其输入为 $X^{(\ell-1)}$，并将其视为由 $H$ 个块拼接而成：

$$\left( X_1^{(\ell-1)} \quad X_2^{(\ell-1)} \quad \dots \quad X_H^{(\ell-1)} \right).$$

我们将 Transformer $T'$ 的输入定义为 $X_1^{(0)} = Y_{(0)}$ 以及 $X_h^{(\ell-1)} := 0_{N\times(m/H)}$ 对所有 $h > 1$均成立。我们将分别证明两种情况：$T$ 不包含多层感知机，以及 $T$ 包含多层感知机，其参数为 $d_{\text{hid}} = O(m)$，激活函数为 $\sigma$。

**情况 1：** $T$ **不包含多层感知机**。我们根据归纳假设继续证明

$$\sum_{h=1}^{H} X_h^{(\ell-1)} = Y^{(\ell-1)}.$$

也就是说，所有块的总和就是 $T$ 的第 $\ell-1$ 层的输出。注意，基础情形按构造满足。

考虑第 $h$ 个注意力头。固定 $T$ 中一个具有嵌入映射 $Q_h$、$K_h$、$V_h$ :$\mathbb{R}^{N\times(m/H)} \to \mathbb{R}^{N\times(m/H)}$ 的注意力头（回想一下，$T$ 的嵌入维度为 $m/H$）。我们设计具有类似嵌入映射的注意力头。定义映射 $g$ : $\mathbb{R}^m \to \mathbb{R}^{m/H}$，其中

$$g : v \mapsto v \begin{pmatrix} I_{m/H} \\ \vdots \\ I_{m/H} \end{pmatrix} := vW_g$$

使得 $g(v)_i = \sum_{i\equiv i' \text{ 模 } (m/H)} v_{i'}$。

If $T$ 没有多层感知机，我们设计嵌入映射 $Q'_h$、$K'_h$、$V'_h$: $\mathbb{R}^{N\times m} \to \mathbb{R}^{N\times(m/H)}$ a s

$$Q'_h = Q_h \circ g, K'_h = K_h \circ g, V'_h = V_h \circ g.$$

注意，如果 $Q_h : X \to XW_{Q_h}$ 可以表示为线性函数，那么 $Q'_h : X \to XW_gW_{Q_h}$ 也可以（$K_h$、$V_h$ 的情况同理）。根据归纳假设，我们有

$$Q'_h\left(\begin{pmatrix} X_1^{(\ell-1)} & X_2^{(\ell-1)} & \dots & X_H^{(\ell-1)} \end{pmatrix}\right) = Q_h\left(\sum_h X_h^{(\ell-1)}\right) = Q_h(Y^{(\ell-1)})$$

正是由 $T$ 的第 $\ell$ 层的第 $h$ 个注意力头计算得到的嵌入。对于 $K'_h$，$V'_h$，类似的论证同样成立。因此，拼接后的注意力头输出

$$\begin{pmatrix} Y^{(\ell,1)} & \dots & Y^{(\ell,H)} \end{pmatrix}.$$

该层的输出随后为

$$\begin{pmatrix} Y^{(\ell,1)} + X_1^{(\ell-1)} & \dots & Y^{(\ell,H)} + X_H^{(\ell-1)} \end{pmatrix}$$

从而

$$\sum_h Y^{(\ell,h)} + X_h^{(\ell-1)} = Y^{(\ell-1)} + \sum_h Y^{(\ell,h)} = Y^{(\ell)}$$

满足归纳假设。

在最后一层，Transformer $T'$ 输出

$$\begin{pmatrix} Y^{(L,1)} & Y^{(L,2)} & \dots & Y^{(L,H)} \end{pmatrix}$$

使得 $g(T'(X)) = T(X)$。为完成证明，注意到 $g$ 可以很容易地用规模为 $O(m)$ 的算术电路（因此也能用 eAC）实现。由于每个嵌入映射 $Q_h$，$K_h$，$V_h$ 接收相同的输入（即 $g$ 的输出），我们可以用规模为 $O(m)$ 的算术电路实现所有 $H$ 嵌入映射。

**情况 2：** $T$ **具有带有** $d_{\text{hid}} = O(m/H)$ **和** $\sigma$ **激活函数的多层感知机或 GLU 多层感知机。** 如果 $T$ 具有隐藏维度为 $d_{\text{hid}} = O(m)$、激活函数为 $\sigma$的多层感知机，则我们改为使用归纳假设

$$X_1^{(\ell-1)} = Y^{(\ell-1)} \text{ and } X_h^{(\ell-1)} = 0_{N\times m} \text{ for } h > 1.$$

也就是说，$X_1^{(\ell-1)}$ 是 $T$ 的第（$\ell-1$）层的输出，其余块均为 0。基础情形同样由定义成立。

考虑第 $\ell$ 层。定义映射 $\pi\colon\ \mathbb{R}^m \to \mathbb{R}^{m/H}$ 如下：

$$\pi : v \mapsto v \begin{pmatrix} I_{m/H} \\ 0_{(m-m/H)\times(m/H)} \end{pmatrix} = (v_1, \ldots, v_{m/H}).$$

此外，对于 $a \le b$，定义 $i\colon\ \mathbb{R}^a \to \mathbb{R}^b$ 为映射 $i\colon\ v \mapsto (v,\ 0_{b-a})$，该映射是填充为 0 的单射。对于每个固定的注意力头 $h$，类似地定义 $Q'_h = Q_h \circ \pi$ 和 $K'_h,\ V'_h$。这里请注意，$Q'_h,\ K'_h,\ V'_h$ 是嵌入，实际上可以用规模为 $O(m/H)$ 的算术电路实现，因此所有嵌入映射都可以用规模为 $O(m)$ 的算术电路实现。

现在，对于多层感知机，如果 $T$ 具有由 $W_1\colon\ \mathbb{R}^{(m/H)+1} \to \mathbb{R}^{d_{\text{hid}}},\ W_2\colon\ \mathbb{R}^{d_{\text{hid}}+1} \to \mathbb{R}^m,\ \sigma$ 给出的多层感知机，我们令 $T'$ 具有如下给出的多层感知机：

$$W'_1(X) := \begin{pmatrix} X & 1 \end{pmatrix} \begin{pmatrix} I_{m/H} & 0_{(m/H)\times 1} \\ \vdots & \vdots \\ I_{m/H} & 0_{(m/H)\times 1} \\ 0_{1\times(m/H)} & 1 \end{pmatrix} W_1, \quad W'_2(X) := \begin{pmatrix} \begin{pmatrix} X & 1 \end{pmatrix} W_2 & 0_{1\times(m-(m/H))} \end{pmatrix}$$

并且 $\sigma$ 激活 函数。注意，$W_1, W_2$ 都可以实现为规模为 $O(m)$ 的 eAC .

我们断言，该构造得出了所需结果。根据归纳假设，在第 $\ell$ 层中，注意力头的输出（拼接后）为

$$Z := \begin{pmatrix} Y^{(\ell,1)} & Y^{(\ell,2)} & \ldots & Y^{(\ell,H)} \end{pmatrix}.$$

特别地，多层感知机 $W'_1$ 的第一个线性映射计算

$$\begin{aligned} W'_1\left(Y^{(\ell-1)} + Z\right) &= W'_1\left(\begin{pmatrix} Y^{(\ell,1)} + Y^{(\ell-1)} & Y^{(\ell,2)} & \ldots & Y^{(\ell,H)} \end{pmatrix}\right) \\ &= \begin{pmatrix} Y^{(\ell-1)} + \sum_h Y^{(\ell,h)} & 1_{N\times 1} \end{pmatrix} W_1 \end{aligned}$$

恰好是由 $T$ 计算得到的激活函数 $\sigma$ 的输入。这就完成了归纳情形的证明，因为 $W'_2$ 将多层感知机的输出输出到 $X_1^{(\ell)}$，而在其他位置输出 0。通过以类似方式修改 $W_1$，可以对 GLU 多层感知机进行类似修改。 □