Transformer 初始化的有效理论

EmilyDinan，* Sho Yaida，† 和
SusanZhang‡*Meta AI Meta Platforms, Inc.*§

我们对宽而深的 Transformer 中的前向—反向信号传播进行有效理论分析；这类模型是包含多头自注意力模块和多层感知机模块的残差神经网络。分析表明，这些模型的初始化和训练超参数应采用特定的宽度缩放。随后，我们采纳这些建议，在实际设置中训练视觉与语言 Transformer。

arXiv:2304.02034v1 [cs.LG] 4 Apr 2023

*电子邮箱：edinan@meta.com
†电子邮箱：shoyaida@meta.com
‡电子邮箱：susanz@meta.com§作者的排序由一次假想的抛硬币决定，该抛硬币百分之百遵循字母顺序。

目录

I. 理论基础 4

0. Transformer 速成课程 5

A. 词干模块 5
B. 层归一化 7
C. 多头自注意力块 7
D. 多层感知机块 8
E. 头部块 9

1. 预激活统计 10

A. 词干模块 11
B. 层归一化 13
C. 多头自注意力块 15
D. 多层感知机块 18
E. 头部块 20

2. 神经切线核速成课 23

A. 普通随机梯度下降 23
B. AdamW 24

3. 神经切线核的统计 26

A. 词干模块 28
B. 层归一化 31
C. 多头自注意力块 33
D. 多层感知机块 38
E. 头部块 39

II. 实际应用 43

1. 使用仅编码器 Transformer 进行图像分类 44

A. 数据集与任务 44
B. 架构超参数 44
C. 初始化超参数 45
D. 优化器与训练超参数 45
E. 缩放策略比较 46

2. 使用编码器–解码器 Transformer 进行跨度去噪 49

A. 数据集与任务 49 B. 架构超参数 49 C. 初始化超参数 51 D. 优化器与训练超参数 51 E. 缩放策略比较 52

A. 自注意力统计中的注意力 57**参考文献** 60

I. 理论基础

汽车人，变形出发！擎天柱

我们正处于人工智能科学与技术发展史上一个令人振奋的时代。一方面，最先进的模型正以眨眼般的速度不断变大 [1–5],截至本文撰写时，其模型参数数量已达万亿级。另一方面，我们拥有一种有效理论描述，随着模型变得更宽 [6–9] 、更深 [10–12]，它在渐近意义上会变得更加准确，也更具相关性。因此，如今我们没有理由不让理论与实践更加紧密地结合起来。

Transformer [13, 14] 就是一个理想的例子，这不仅是因为它们推动了这个时代的诸多技术创新，也因为为了实现这些创新，它们变得如此之宽、如此之深，以至于有效理论原则上应当越来越适用，也越来越切合实际。抱着这一乐观信念，在本文第一部分，我们将推出参考文献 [15] 中提出的通用蓝图，并对其进行调整，以建立 Transformer 的有效理论；在第二部分，我们将进一步探讨由此得到的理论建议在实践中的影响。[1]

第一部分其余内容安排如下。我们将在 §I 0 中简要回顾 Transformer 的构建模块，并介绍本文使用的符号记法，以此展开前向路径分析。接着，在 §I 1 中，我们将计算从输入到输出的初始化预激活统计量：这项分析有助于确定如何根据宽度缩放初始化超参数（1.25）–（1.33）。随后转向反向路径，在 §I 2 中简要介绍神经切线核之后，我们将在 §I 3 中计算梯度平方的统计均值：这项分析有助于确定如何根据宽度缩放分组学习率因子，分别适用于随机梯度下降优化器（1.99）–（1.107）和 AdamW 优化器（1.108）–（1.116）。[2]

[1] 现实地说，Transformer 不会永远处于神经网络演进的巅峰，而是会被其他模型家族取代。本说明的总体目标是提供一份 *meta-blueprint*，以便将所述有效理论蓝图推广到我们遇到的更复杂的神经网络架构中——该蓝图曾在参考文献 [15] 中以多层感知机这一简单情形为例进行阐述。为此，我们会尽可能详尽地介绍，以便读者相对容易地仿照我们推广这一蓝图的方式。[2] 对于了解内情的读者：我们的有效理论分析停留在元层面。也就是说，有两项工作我们 *won't* 会 *(a)* 做：临界性分析——因为归一化层和残差连接基本上已经解决了可能出现的梯度爆炸或梯度消失问题——以及 *(b)* 有限宽度分析——因为这样的处理会使本说明的篇幅增加四倍，而且要读出我们将在第二部分检验的超参数缩放，并不需要付出这番工夫。尽管如此，我们希望本说明能为那些勇于推进这些扩展工作的人奠定基础；例如，这些工作有助于探究如何审慎地缩放架构超参数。

0. Transformer 速成课程

给定数据集 $\mathcal{D}$，我们将其分词后的输入记为 $x_{\alpha;t;i}$，其中样本索引为 $\alpha = 1, \ldots, |\mathcal{D}|$，词元索引为 $t = 1, \ldots, T$，向量索引为 $i = 1, \ldots, n_{\text{in}}$（具体示例见 §I 0 A）。接着，嵌入维度为 n 的标准 Transformer [13, 14] 递归地对这些输入进行变换：

$$z^{(1)}_{\alpha;t;i} = \mathtt{R}^{(1)}_{t;i}\left(x_\alpha; \theta^{(1)}\right) \qquad \text{for} \quad i = 1, \ldots, n\,, \tag{1.1}$$

$$s^{(\ell)}_{\alpha;t;i} = \mathtt{LN}_i\left(z^{(\ell)}_{\alpha;t}\right) \qquad \text{for} \quad i = 1, \ldots, n; \quad \ell = 1, \ldots, L-1\,, \tag{1.2}$$

$$z^{(\ell+1)}_{\alpha;t;i} = \mathtt{R}^{(\ell+1)}_{t;i}\left(s^{(\ell)}_\alpha; \theta^{(\ell+1)}\right) + z^{(\ell)}_{\alpha;t;i} \quad \text{for} \quad i = 1, \ldots, n; \quad \ell = 1, \ldots, L-2\,, \tag{1.3}$$

$$z^{(L)}_{\alpha;t;i} = \mathtt{R}^{(L)}_{t;i}\left(s^{(L-1)}_\alpha; \theta^{(L)}\right) \qquad \text{for} \quad i = 1, \ldots, n_{\text{out}}\,, \tag{1.4}$$

其中 $\theta^{(\ell)}$ 是第 ℓ 个块的模型参数。具体来说，首先，stem-block 操作 $\mathtt{R}^{(1)}$ (1.1) 将输入 $x_{\alpha;t;i}$ 转换为第一块预激活 $z^{(1)}_{\alpha\, t\, i;;}$，这些预激活是 T 序列的 n 维向量；接着，这些预激活依次通过一系列层归一化操作 (1.2) [16] (§I 0 B) 和块操作 (1.3)——其中每个残差路径 $\mathtt{R}^{(\ell)}$ 要么是多头自注意力块 (§I 0 C)，要么是多层感知机块 (§I 0 D)——并由跳跃连接夹在两侧；最后，输出模块操作 $\mathtt{R}^{(L)}$ (1.4) 将它们转换为 T 序列的 n_{out} 维向量 $z^{(L)}_{\alpha\, t\, ;;i}$（具体示例请参见 §I 0 E）。[3]

现在我们已经有了整体示意图，接下来逐块详细定义每项操作。

A. 词干模块

Vision: patchify embedding and positional embedding

对于视觉 Transformer， [14], 输入图像通常会被划分为不重叠图像块。更具体地说，在典型的 ImageNet [18] 训练设置中，给定一张 3 通道、224×224 像素的彩色图像

[3] 更一般地，每个跳跃连接都可以拥有自己的一组参数，如下所示

$$z^{(\ell+1)}_{\alpha;t;i} = \mathtt{R}^{(\ell+1)}_{t;i}\left(s^{(\ell)}_\alpha; \theta^{(\ell+1)}\right) + \xi^{(\ell+1)}_i z^{(\ell)}_{\alpha;t;i}\,, \tag{1.5}$$

其中，逐元素仿射参数 $\xi^{(\ell)}_i$ 决定了我们在每个通道中复制并传递多少信号。不过，在本文中，为简单起见，我们将其固定为 $\xi^{(\ell)}_i = 1$ ，这在实践中也很常见。类似地，每个层归一化操作都可以带有各自可训练的逐元素仿射参数（§I 0 B），但在理论分析和实践中，我们仍将其固定；这种做法不太常见，但也并非闻所未闻: https://pytorch.org/docs/stable/generated/torch.nn.LayerNorm.html。顺带一提，归一化层最初的放置位置有所不同 [13], ，但上述布局如今更为标准；对于残差神经网络，这种布局还能确保块与块之间的信号传播 *everywhere critical* [17]，也就是说，它能降低训练成功对初始化超参数选择的敏感性。

图像，例如，我们可以将其划分为 ($T = 14^2 = 196$) 个由不重叠图像块组成的词元，每个词元都可以视为一个 ($n_{\text{in}} \equiv n_{\text{patch}} = 16 \cdot 16 \cdot 3 = 768$) 维向量。为实现这一点，我们可以使用分块权重——步幅为 16、大小为 16×16 的卷积权重，输入 3 个输入通道并输出 n 个输出通道——它们的作用是

$$\tilde{z}_{\alpha;t;i} = \sum_{j=1}^{n_{\text{patch}}} W_{ij}^{\text{patch}} x_{\alpha;t;j} \,, \tag{1.6}$$

使用嵌入索引 $i = 1, \ldots, n$。请注意，由于卷积权重共享，这些权重对所有词元 $t = 1, \ldots, T$的作用方式都相同。随后，通常会用位置嵌入参数对它们进行修正，如下所示

$$z_{\alpha;t;i}^{(1)} = b_{t;i}^{\text{PE}} + \tilde{z}_{\alpha;t;i} = b_{t;i}^{\text{PE}} + \sum_{j=1}^{n_{\text{patch}}} W_{ij}^{\text{patch}} x_{\alpha;t;j} \,, \tag{1.7}$$

其中， T×n 维张量 $b_{t;i}^{\text{PE}}$起到偏置参数的作用，通常认为它有助于区分不同的图像块。[4]

语言：词嵌入和位置嵌入

对于语言 Transformer， [13], 输入通常以独热向量 $x_{\alpha;t;j} = \delta_{jj_\star}$ 的形式给出，或者至少可以解释为独热向量；其中，向量索引 $j = 1, \ldots, n_{\text{vocab}}$ 遍历词汇表中的所有可能词元。对于给定的样本—标记索引对 (α， t)，输入向量在特定索引 $j_\star = j_\star(\alpha; t)$处取单位值，其余位置均为零。随后，每个词都会被嵌入到一个 n 维空间中，如下所示

$$\tilde{z}_{\alpha;t;i} = \sum_{j=1}^{n_{\text{vocab}}} W_{ij}^{\text{WE}} x_{\alpha;t;j} = W_{ij_\star(\alpha;t)}^{\text{WE}} \,, \tag{1.9}$$

其中，词嵌入参数 W_{ij}^{WE}的作用类似于一个 n×n_{vocab} 权重矩阵。与视觉情形 (1.7) 一样，随后会通过位置嵌入参数对其进行修正，形式如下： $z_{\alpha\, t\, ;;i}^{(1)} = \tilde{z}_{\alpha;t;i} + b_{t;i}^{\text{PE}}$。

---

[4]视觉 Transformer 的词干模块有几种变体。其中一种变体是添加一个 n 维的类别词元 c_i，例如添加在 $t = 0$，这样我们就得到 ($T + 1$)×n 维的第一块预激活

$$z_{\alpha;t;i}^{(1)} = \begin{cases} c_i + b_{0;i}^{\text{PE}} & \text{for} \quad t = 0 \,, \\ b_{t;i}^{\text{PE}} + \sum_{j=1}^{n_{\text{patch}}} W_{ij}^{\text{patch}} x_{\alpha;t;j} & \text{for} \quad t = 1, \ldots, T \,, \end{cases} \tag{1.8}$$

对每个输入 α [14] （另见参考文献 [19]）。另一种变体是堆叠若干卷积层 [14, 20]。虽然我们将重点讨论正文中所述的词干模块，但要考虑这些变体也很简单。

B. 层归一化

一般而言，层归一化操作 [16]定义为

$$s^{(\ell)}_{\alpha;t;i} = \gamma^{(\ell)}_i \left[\frac{z^{(\ell)}_{\alpha;t;i} - \left(\frac{1}{n} \sum_{j=1}^{n} z^{(\ell)}_{\alpha;t;j} \right)}{\sqrt{\frac{1}{n} \sum_{j=1}^{n} \left(z^{(\ell)}_{\alpha;t;j} \right)^2 - \left(\frac{1}{n} \sum_{j=1}^{n} z^{(\ell)}_{\alpha;t;j} \right)^2 + \epsilon}} \right] + \beta^{(\ell)}_i \,, \tag{1.10}$$

正则化参数 ϵ 以及逐元素仿射参数 $\gamma^{(\ell)}_i$ 和 $\beta^{(\ell)}_i$。为简化起见，在本文中，我们将这些逐元素仿射参数固定为 $\gamma^{(\ell)}_i = 1$ 和 $\beta^{(\ell)}_i = 0$，即，

$$s^{(\ell)}_{\alpha;t;i} = \mathtt{LN}_i \left(z^{(\ell)}_{\alpha;t} \right) = \frac{z^{(\ell)}_{\alpha;t;i} - \left(\frac{1}{n} \sum_{j=1}^{n} z^{(\ell)}_{\alpha;t;j} \right)}{\sqrt{\frac{1}{n} \sum_{j=1}^{n} \left(z^{(\ell)}_{\alpha;t;j} \right)^2 - \left(\frac{1}{n} \sum_{j=1}^{n} z^{(\ell)}_{\alpha;t;j} \right)^2 + \epsilon}} \,. \tag{1.11}$$

特别是，当 $\epsilon = 0$ 时，每个信号都恰好归一化为单位值，如 $(1/n) \sum_{i=1}^{n} \left(s^{(\ell)}_{\alpha\, t\,;;i} \right)^2 = 1$，而当 ϵ足够小时，则近似归一化为单位值。

C. 多头自注意力模块

多头自注意力（MHSA）模块的残差路径接收一个沿 T 排列的 n 维信号 $s_{t;i}$，并输出另一个 (T, n) 维张量 $r_{t;i}$，其在序列方向 t上发挥非平凡作用。具体而言——省略块索引 ℓ ——具有 H 个注意力头的标准 MHSA 残差块操作，

$$r_{\alpha;t;i} = \mathtt{R}^{\mathrm{MHSA}}_{t;i}(s_\alpha; Q, K, V, U) \,, \tag{1.12}$$

定义如下。

1. 将查询向量、键向量和值向量定义为

$$q^h_{\alpha;t;c} \equiv \sum_{i=1}^{n} Q^h_{ci} s_{\alpha;t;i} \,, \tag{1.13}$$

$$k^h_{\alpha;t;c} \equiv \sum_{i=1}^{n} K^h_{ci} s_{\alpha;t;i} \,, \tag{1.14}$$

$$v^h_{\alpha;t;c} \equiv \sum_{i=1}^{n} V^h_{ci} s_{\alpha;t;i} \,, \tag{1.15}$$

对 $t = 1, \ldots\ldots, T$、 $c = 1, \ldots\ldots, C$ 和 $h = 1, \ldots\ldots, H$，其中每个头部的通道数 $C \equiv n/H$ 必须为整数。

2. 将查询–键点积定义为

$$\widetilde{\Omega}^h_{\alpha;tt'} \equiv \frac{1}{\sqrt{C}} \sum_{c=1}^{C} q^h_{\alpha;t;c} k^h_{\alpha;t';c} = \frac{1}{\sqrt{C}} \sum_{c=1}^{C} \sum_{i_1,i_2=1}^{n} Q^h_{ci_1} K^h_{ci_2} s_{\alpha;t;i_1} s_{\alpha;t';i_2} \,. \tag{1.16}$$

（我们将在 §I 1 C 中首先说明因子 $1/\sqrt{C}$ 的合理性，然后在 §I 3 C 中再次说明。）

3. 将自注意力矩阵 $\Omega^h_{\alpha;tt'}$ 定义为基于查询–键点积计算的 Softmax。具体而言，对于双向 多头自注意力块（用于编码器），我们通常设定

$$\Omega^h_{\alpha;tt'} \equiv \frac{\exp\left(\widetilde{\Omega}^h_{\alpha;tt'}\right)}{\sum_{t''=1}^{T} \exp\left(\widetilde{\Omega}^h_{\alpha;tt''}\right)} \,, \tag{1.17}$$

而对于掩码 多头自注意力块（用于解码器），我们通常设定

$$\Omega^h_{\alpha;tt'} \equiv \begin{cases} \frac{\exp\left(\widetilde{\Omega}^h_{\alpha;tt'}\right)}{\sum_{t''=1}^{t} \exp\left(\widetilde{\Omega}^h_{\alpha;tt''}\right)} & \text{for} \quad t' \leq t \,, \\ 0 & \text{for} \quad t' > t \,, \end{cases} \tag{1.18}$$

该矩阵会因果性地屏蔽查询者无法访问的未来词元。[5]

4. 对每个注意力头，沿词元方向用自注意力矩阵对值向量加权，然后通过统一矩阵（亦称输出投影矩阵） U^h_{ic} 将 H 个注意力头和 C 个通道合并，定义为

$$r_{\alpha;t;i} \equiv \sum_{h=1}^{H} \sum_{c=1}^{C} U^h_{ic} \left(\sum_{t'=1}^{T} \Omega^h_{\alpha;tt'} v^h_{\alpha;t';c} \right) = \sum_{h=1}^{H} \sum_{t'=1}^{T} \Omega^h_{\alpha;tt'} \sum_{c=1}^{C} \sum_{j=1}^{n} U^h_{ic} V^h_{cj} s_{\alpha;t';j} \,. \tag{1.19}$$

D. 多层感知机块

Transformer 中多层感知机（MLP）块的残差路径通常具有如下形式

$$w_{\alpha;t;i} = \sum_{j=1}^{n} W_{ij} s_{\alpha;t;j} \qquad \text{for} \quad i = 1, \ldots, Mn \,, \tag{1.20}$$

$$r_{\alpha;t;i} = \sum_{j=1}^{Mn} X_{ij} \sigma\left(w_{\alpha;t;j}\right) \quad \text{for} \quad i = 1, \ldots, n \,, \tag{1.21}$$

其中，σ 是激活函数，M 是多层感知机倍数；对于足够大的 Transformer，通常将其设为 $M = 4$。[6]

[5] 一般而言，自注意力矩阵可以是查询–键点积 $\Omega^h_{\alpha;\,tt'} = \mathcal{F}_{tt'}\left(\widetilde{\Omega}^h_{\alpha}\right)$ 的任意一阶矩阵值函数，该函数将一个 $T\times T$ 矩阵映射为一个 $T\times T$ 矩阵。事实上，在本文中，我们不会用到函数 $\mathcal{F}$ 的任何特定性质，只要求它是一阶的。[6] 关于多层感知机块的某种变体，参见例如参考文献 [21]。

E. 头部块

视觉：线性分类层

对于视觉 Transformer，头部块通常只由一个标准线性层组成 as

$$z^{(L)}_{\alpha;t;i} = b^{\text{head}}_i + \sum_{j=1}^{n} W^{\text{head}}_{ij} s^{(L-1)}_{\alpha;t;j} \quad \text{for} \quad i = 1, \ldots, n_{\text{out}}\,, \tag{1.22}$$

其中 例如，对于图像分类任务， $n_{\text{out}} \equiv n_{\text{class}}$ 表示类别数量。[7]

Language: word embedding, transposed (and rescaled)

对于语言 Transformer，我们通常将词干与头部模块之间的权重绑定 [22],这意味着我们将信号乘以词嵌入参数的 转置，即

$$z^{(L)}_{\alpha;t;i} = \mathcal{N}_{\text{rescale}} \sum_{j=1}^{n} (W^{\text{WE}})^{\top}{}_{ij} s^{(L-1)}_{\alpha;t;j} = \mathcal{N}_{\text{rescale}} \sum_{j=1}^{n} W^{\text{WE}}_{ji} s^{(L-1)}_{\alpha;t;j} \quad \text{for} \quad i = 1, \ldots, n_{\text{out}}\,, \tag{1.24}$$

与 $n_{\text{out}} \equiv n_{\text{vocab}}$。这里，我们还引入了重缩放因子 $\mathcal{N}_{\text{rescale}}$；其存在的原因及其宽度缩放将在 §I 1 E 中首次阐明，并在 §I 3 E 中再次得到论证。

[7] 在一种实现方式中（我们将在 §II1 的视觉Transformer实验中采用），实际输出通过沿词元方向进行均值池化得到:

$$\widetilde{z}^{(L)}_{\alpha;i} = \frac{1}{T} \sum_{t=1}^{T} z^{(L)}_{\alpha;t;i} = b^{\text{head}}_i + \sum_{j=1}^{n} W^{\text{head}}_{ij} \left(\frac{1}{T} \sum_{t=1}^{T} s^{(L-1)}_{\alpha;t;j} \right) = b^{\text{head}}_i + \sum_{j=1}^{n} W^{\text{head}}_{ij} \widetilde{s}^{(L-1)}_{\alpha;j}\,, \tag{1.23}$$

其中 $\widetilde{s}^{(L-1)}_{\alpha\,;\,j} \equiv (1/T) \sum_{t=1}^{T} s^{(L-1)}_{\alpha\,t\,j}$。在另一种实现中——我们在脚注 4 中提到过，但这里不采用——类别词元会在词干模块的 $t = 0$ 处附加，输出则由 $\widetilde{z}^{(L)}_{\alpha\,;\,i} = z^{(L)}_{\alpha\,0\,;\,;\,i}$ 给出。

1. 预激活的统计

在开始任何训练之前，都需要以某种方式初始化模型参数。通常，它们是通过从均值为零的均匀分布、正态分布或截断正态分布中独立同分布地抽取得到的。为统一记号，我们先约定，它们的协方差应按如下方式缩放

$$\mathbb{E}\left[W^{\text{patch}}_{i_1 j_1} W^{\text{patch}}_{i_2 j_2}\right] = \left(\frac{C_{\text{patch}}}{n_{\text{patch}}}\right) \delta_{i_1 i_2} \delta_{j_1 j_2} \qquad \Big|\Big| \quad \mathbb{E}\left[W^{\text{WE}}_{i_1 j_1} W^{\text{WE}}_{i_2 j_2}\right] = \left(C_{\text{WE}}\right) \delta_{i_1 i_2} \delta_{j_1 j_2} \,, \tag{1.25}$$

$$\mathbb{E}\left[b^{\text{PE}}_{t_1;i_1} b^{\text{PE}}_{t_2;i_2}\right] = \left(C_{\text{PE}}\right) \delta_{t_1 t_2} \delta_{i_1 i_2} \,, \tag{1.26}$$

$$\mathbb{E}\left[Q^{h_1}_{c_1 i_1} Q^{h_2}_{c_2 i_2}\right] = \left(\frac{C_Q}{n}\right) \delta_{c_1 c_2} \delta_{i_1 i_2} \delta^{h_1 h_2} \,, \tag{1.27}$$

$$\mathbb{E}\left[K^{h_1}_{c_1 i_1} K^{h_2}_{c_2 i_2}\right] = \left(\frac{C_K}{n}\right) \delta_{c_1 c_2} \delta_{i_1 i_2} \delta^{h_1 h_2} \,, \tag{1.28}$$

$$\mathbb{E}\left[V^{h_1}_{c_1 i_1} V^{h_2}_{c_2 i_2}\right] = \left(\frac{C_V}{n}\right) \delta_{c_1 c_2} \delta_{i_1 i_2} \delta^{h_1 h_2} \,, \tag{1.29}$$

$$\mathbb{E}\left[U^{h_1}_{i_1 c_1} U^{h_2}_{i_2 c_2}\right] = \left(\frac{C_U}{n}\right) \delta_{i_1 i_2} \delta_{c_1 c_2} \delta^{h_1 h_2} \,, \tag{1.30}$$

$$\mathbb{E}\left[W_{i_1 j_1} W_{i_2 j_2}\right] = \left(\frac{C_W}{n}\right) \delta_{i_1 i_2} \delta_{j_1 j_2} \,, \tag{1.31}$$

$$\mathbb{E}\left[X_{i_1 j_1} X_{i_2 j_2}\right] = \left(\frac{C_X}{Mn}\right) \delta_{i_1 i_2} \delta_{j_1 j_2} \,, \tag{1.32}$$

$$\mathbb{E}\left[W^{\text{head}}_{i_1 j_1} W^{\text{head}}_{i_2 j_2}\right] = \left(\frac{C_{\text{head}}}{n}\right) \delta_{i_1 i_2} \delta_{j_1 j_2} \,, \; b^{\text{head}}_i = 0 \quad \Big|\Big| \quad \mathcal{N}_{\text{rescale}} = \sqrt{\frac{1}{n}} \,, \tag{1.33}$$

其中， $\mathbb{E}\left[\cdot\right]$ 表示相对于初始化分布的期望值， δ_{ij} 是克罗内克δ（即，当 $i = j$ 时为 $\delta_{ij} = 1$ ，当 $i \neq j$时为 $\delta_{ij} = 0$ ），并且对于模型参数的每一组 G ，所有初始化超参数 C_G 都是一阶量。在这里，“一阶量”是指：当我们缩放网络的宽度 n、深度L以及其他任何架构超参数时，这些超参数保持不变；我们也常说“可观测量保持为一阶量”，意思是，随着网络规模不断增大，所说的可观测量数值不会发散到无穷大，也不会趋于零。[8]

[8] 更一般地，如果我们想采用通用的元原理缩放策略 [23] ，在神经切线缩放策略 [9] （取 $s = 0$）与最大更新缩放策略 [24] （取 $s = 1$）之间进行插值，那么应将输出头缩放（1.33）修改为

$$\mathbb{E}\left[W^{\text{head}}_{i_1 j_1} W^{\text{head}}_{i_2 j_2}\right] = \left(\frac{C_{\text{head}}}{n^{1+s}}\right) \delta_{i_1 i_2} \delta_{j_1 j_2} \,, \;\; b^{\text{head}}_i = 0 \quad \Big|\Big| \quad \mathcal{N}_{\text{rescale}} = \sqrt{\frac{1}{n^{1+s}}} \,, \tag{1.34}$$

正文中，我们将重点讨论满足 $s = 0$的神经切线缩放策略，而学习率因子相应的元变化只会在后文脚注 26 中提及。

在本节余下部分，我们将看到，如何选择这些初始化超参数的宽度缩放方式（1.25）–（1.33），以确保随着 Transformer 宽度增加，预激活 $z^{(\ell)}_{\alpha\, t\;;;i}$ 保持为一阶量。更具体地说，我们将递归地证明，它们的协方差具有如下形式

$$\mathbb{E}\left[z^{(\ell)}_{\alpha_1;t_1;i_1} z^{(\ell)}_{\alpha_2;t_2;i_2}\right] = \delta_{i_1 i_2} G^{(\ell)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\,, \tag{1.35}$$

对于阶为一的核 $G^{(\ell)}{}_{(\alpha_1;t_1)(\alpha_2;t_2)}$ 此外，我们还将在 §I 1 B 中证明，层归一化操作之后信号的期望平方范数为

$$F^{(\ell)}_{(\alpha_1;t_1)(\alpha_2;t_2)} \equiv \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} s^{(\ell)}_{\alpha_1;t_1;i} s^{(\ell)}_{\alpha_2;t_2;i}\right] = \frac{G^{(\ell)}_{(\alpha_1;t_1)(\alpha_2;t_2)}}{\sqrt{G^{(\ell)}_{(\alpha_1;t_1)(\alpha_1;t_1)}+\epsilon}\sqrt{G^{(\ell)}_{(\alpha_2;t_2)(\alpha_2;t_2)}+\epsilon}} + O\left(\frac{1}{n}\right)\,, \tag{1.36}$$

在 $1/n$ 的主阶上。总之，我们将证明，在规定的初始化超参数缩放 (1.25)–(1.33) 下，Transformer 遵循临界性原理 [15], 也就是说，前向路径上的信号保持在一阶量级。[9]

考虑到这些目标，我们来逐块向前分析预激活的统计量 .

A. 词干模块

视觉：分块嵌入与位置嵌入

在视觉 Transformer 的词干模块中，patchify 权重和位置嵌入参数以零均值分布进行初始化，其协方差按以下方式缩放；这些参数会产生第一块预激活 $z^{(1)}_{\alpha\, t\;;;i} = b^{\mathrm{PE}}_{t;i} + \sum_{j=1}^{n_{\mathrm{patch}}} W^{\mathrm{patch}}_{ij} x_{\alpha;t;j}$

$$\mathbb{E}\left[W^{\mathrm{patch}}_{i_1 j_1} W^{\mathrm{patch}}_{i_2 j_2}\right] = \left(\frac{C_{\mathrm{patch}}}{n_{\mathrm{patch}}}\right)\delta_{i_1 i_2}\delta_{j_1 j_2}\,, \quad \mathbb{E}\left[b^{\mathrm{PE}}_{t_1;i_1} b^{\mathrm{PE}}_{t_2;i_2}\right] = (C_{\mathrm{PE}})\,\delta_{t_1 t_2}\delta_{i_1 i_2}\,, \tag{1.37}$$

分别采用一阶初始化超参数 C_{patch} 和 C_{PE}。这样，我们的均值为零,

$$\mathbb{E}\left[z^{(1)}_{\alpha;t;i}\right] = 0\,, \tag{1.38}$$

[9] 也就是说，这是在元层面上的。如果没有层归一化，我们就会进一步遵循非元层面的临界性原理 [10–12, 15]，以微调初始化超参数 C_G 和（初始）逐元素仿射参数（$\gamma^{(\ell)}_i, \beta^{(\ell)}_i, \xi^{(\ell)}_i$），从而避免信号问题，即信号呈指数级爆炸或消失；由于通常存在归一化层，我们将在 §I 1 C 和 §I 1 D 中看到，核 $G^{(\ell)}$ 随深度 ℓ 仅线性增长，直到它在进入 $\ell = L$ 处的头部块之前得到进一步归一化（§I 1 E）。

协方差为一阶，

$$\begin{aligned}\mathbb{E}\left[z^{(1)}_{\alpha_1;t_1;i_1}z^{(1)}_{\alpha_2;t_2;i_2}\right]=&\mathbb{E}\left[\left(b^{\mathrm{PE}}_{t_1;i_1}+\sum_{j_1=1}^{n_{\mathrm{patch}}}W^{\mathrm{patch}}_{i_1j_1}x_{\alpha_1;t_1;j_1}\right)\left(b^{\mathrm{PE}}_{t_2;i_2}+\sum_{j_2=1}^{n_{\mathrm{patch}}}W^{\mathrm{patch}}_{i_2j_2}x_{\alpha_2;t_2;j_2}\right)\right] \\ =&C_{\mathrm{PE}}\delta_{i_1i_2}\delta_{t_1t_2}+\delta_{i_1i_2}C_{\mathrm{patch}}\left(\frac{1}{n_{\mathrm{patch}}}\sum_{j=1}^{n_{\mathrm{patch}}}x_{\alpha_1;t_1;j}x_{\alpha_2;t_2;j}\right) \\ =&\delta_{i_1i_2}\left(C_{\mathrm{PE}}\delta_{t_1t_2}+C_{\mathrm{patch}}G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\right) \\ =&\delta_{i_1i_2}G^{(1)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\,.\end{aligned} \tag{1.39}$$

在这里的倒数第二行，我们引入了输入核，

$$G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\equiv\frac{1}{n_{\mathrm{patch}}}\sum_{j=1}^{n_{\mathrm{patch}}}x_{\alpha_1;t_1;j}x_{\alpha_2;t_2;j}\,, \tag{1.40}$$

由于所有像素值通常都经过预处理，量级为一，因此它的量级为一；在最后一行，我们引入了词干模块核，

$$G^{(1)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\equiv C_{\mathrm{patch}}G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)}+C_{\mathrm{PE}}\delta_{t_1t_2}\,, \tag{1.41}$$

因此它的量级为一。特别是，第一块预激活的每个分量的预期量级均为一： $\mathbb{E}\left[\left(z^{(1)}_{\alpha\,t\,;;i}\right)^2\right]=G^{(1)}_{(;t)(\alpha;t)}=C_{\mathrm{patch}}\left(\frac{1}{n_{\mathrm{patch}}}\sum_{j=1}^{n_{\mathrm{patch}}}x^2_{\alpha;t;j}\right)+C_{\mathrm{PE}}=O(1)$。

语言：词嵌入和位置嵌入

在语言 Transformer 的词干模块中，该模块输出第一块预激活$z^{(1)}_{\alpha\,t\,;;i}=b^{\mathrm{PE}}_{t;i}+\sum_{j=1}^{n_{\mathrm{vocab}}}W^{\mathrm{WE}}_{ij}x_{\alpha;t;j}$，位置嵌入参数按照视觉情形进行初始化，其均值为零、协方差为$\mathbb{E}\left[b^{\mathrm{PE}}_{t_1;i_1}b^{\mathrm{PE}}_{t_2\,;i_2}\right]=C_{\mathrm{PE}}\delta_{t_1t_2}\delta_{i_1i_2}$；而词嵌入参数的初始化均值为零，协方差为

$$\mathbb{E}\left[W^{\mathrm{WE}}_{i_1j_1}W^{\mathrm{WE}}_{i_2j_2}\right]=\left(C_{\mathrm{WE}}\right)\delta_{i_1i_2}\delta_{j_1j_2}\,, \tag{1.42}$$

其中，初始化超参数 C_{WE} 为一阶量。这里需要注意，我们没有 将协方差除以词汇表大小 n_{vocab}；这样一来，我们就有

$$\begin{aligned}\mathbb{E}\left[z^{(1)}_{\alpha_1;t_1;i_1}z^{(1)}_{\alpha_2;t_2;i_2}\right]=&\mathbb{E}\left[\left(b^{\mathrm{PE}}_{t_1;i_1}+\sum_{j_1=1}^{n_{\mathrm{vocab}}}W^{\mathrm{WE}}_{i_1j_1}x_{\alpha_1;t_1;j_1}\right)\left(b^{\mathrm{PE}}_{t_2;i_2}+\sum_{j_2=1}^{n_{\mathrm{vocab}}}W^{\mathrm{WE}}_{i_2j_2}x_{\alpha_2;t_2;j_2}\right)\right] \\ =&\delta_{i_1i_2}\left[C_{\mathrm{PE}}\delta_{t_1t_2}+C_{\mathrm{WE}}\left(\sum_{j=1}^{n_{\mathrm{vocab}}}x_{\alpha_1;t_1;j}x_{\alpha_2;t_2;j}\right)\right] \\ =&\delta_{i_1i_2}\left[C_{\mathrm{PE}}\delta_{t_1t_2}+C_{\mathrm{WE}}G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\right] \\ =&\delta_{i_1i_2}G^{(1)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\,,\end{aligned} \tag{1.43}$$

其中，输入核

$$G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)} \equiv \sum_{j=1}^{n_{\text{vocab}}} x_{\alpha_1;t_1;j} x_{\alpha_2;t_2;j}\,, \tag{1.44}$$

其为 一阶，这是由词输入的独热结构所致；因此，词干块核 el

$$G^{(1)}_{(\alpha_1;t_1)(\alpha_2;t_2)} \equiv C_{\text{WE}} G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)} + C_{\text{PE}}\delta_{t_1 t_2}\,, \tag{1.45}$$

也相应地为一阶。特别地，输入核 $G^{(0)}\ {}_{(\alpha_1;\ t_1)}\ {}_{(\alpha_2;\ t_2)}$ 当且仅当（α_1；t_1）和（α_2；t_2）对应词汇表中的同一个词元时取值为 1，否则取值为 0。

B. 层归一化

在这里，我们将分析每个层归一化操作 (1.11) 如何变换核。在此之前，我们注意到，层归一化仅沿嵌入方向 i 进行；具体而言，它对每一对样本—标记索引 $(\alpha; t)$ 独立作用。因此，为了使分析更简洁，我们暂时省略这些索引以及块索引 ℓ；待一切尘埃落定后，再将这些索引加回来。有鉴于此，我们将分析对象 (1.36) 的简化版本，

$$F \equiv \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} s_i s_i\right] = \frac{\left(\frac{1}{n}\sum_{i=1}^{n} z_i^2\right) - \left(\frac{1}{n}\sum_{i=1}^{n} z_i\right)^2}{\left[\sqrt{\left(\frac{1}{n}\sum_{i=1}^{n} z_i^2\right) - \left(\frac{1}{n}\sum_{i=1}^{n} z_i\right)^2 + \epsilon}\ \right]^2}\,, \tag{1.46}$$

对于一个向量随机变量 z_i，其均值为零，协方差为一阶

$$\mathbb{E}\left[z_{i_1} z_{i_2}\right] = \delta_{i_1 i_2} G\,. \tag{1.47}$$

直观地说，对于足够大的 n，我们应有 $\frac{1}{n}\sum_{i=1}^{n} z_i^2 \approx G$；而由于 z_i 是均值为零、量级为一的随机数，因此 $\frac{1}{n}\sum_{i=1}^{n} z_i \sim \frac{\sqrt{n}}{n} \approx 0$。为了将这一直觉表述得更精确，我们引入两类有限-$n$ 修正：不同实例间的幅度波动，

$$\widehat{\Delta G} \equiv \left(\frac{1}{n}\sum_{i=1}^{n} z_i^2\right) - G\,, \tag{1.48}$$

以及预激活的均值的平方，

$$\widehat{\nabla G} \equiv \left(\frac{1}{n}\sum_{i=1}^{n} z_i\right)^2\,. \tag{1.49}$$

根据这些定义，我们可以将婴儿对象 (1.46) 简洁地改写为

$$F = \frac{G + \widehat{\Delta G} - \widehat{\nabla G}}{\left[\sqrt{G + \epsilon + \widehat{\Delta G} - \widehat{\nabla G}}\right]^2} \,. \tag{1.50}$$

现在我们将证明

$$\mathbb{E}\left[\widehat{\Delta G}^p \widehat{\nabla G}^q\right] = O\left(\frac{1}{n}\right) \tag{1.51}$$

对于任意整数 $p,\ q \geq 0$ 满足 $p + q > 0$，因此在主阶上，我们可以忽略这些有限-n修正。[10]

让我们逐步推导出该表达式，共分八步（方程后附有冗长的文字说明）：

$$\begin{aligned}
&\mathbb{E}\left[\widehat{\Delta G}^p \widehat{\nabla G}^q\right] \qquad (1.52)\\
=&\mathbb{E}\left\{\left[\sum_{r=0}^{p}\binom{p}{r}\frac{(-G)^{p-r}}{n^r}\sum_{i_1,\ldots,i_r=1}^{n} z_{i_1}^2\cdots z_{i_r}^2\right]\left[\frac{1}{n^{2q}}\sum_{j_1,\ldots,j_{2q}=1}^{n} z_{j_1}\cdots z_{j_{2q}}\right]\right\}\\
=&\sum_{r=0}^{p}\binom{p}{r}(-G)^{p-r}\frac{1}{n^{r+2q}}\sum_{i_1,\ldots,i_r=1}^{n}\sum_{j_1,\ldots,j_{2q}=1}^{n}\mathbb{E}\left[z_{i_1}^2\cdots z_{i_r}^2 z_{j_1}\cdots z_{j_{2q}}\right]\\
=&\sum_{r=0}^{p}\binom{p}{r}(-G)^{p-r}\frac{n(n-1)\cdots(n-r-2q+1)}{n^{r+2q}}\mathbb{E}\left[z_1^2\cdots z_r^2 z_{r+1}\cdots z_{r+2q}\right] + O\left(\frac{1}{n}\right)\\
=&\sum_{r=0}^{p}\binom{p}{r}(-G)^{p-r}\,\mathbb{E}\left[z_1^2\right]\cdots\mathbb{E}\left[z_r^2\right]\mathbb{E}\left[z_{r+1}\right]\cdots\mathbb{E}\left[z_{r+2q}\right] + O\left(\frac{1}{n}\right)\\
=&\sum_{r=0}^{p}\binom{p}{r}(-G)^{p-r}\,G^r 0^q + O\left(\frac{1}{n}\right)\\
=&G^p\sum_{r=0}^{p}\binom{p}{r}(-1)^{p-r}\,0^q + O\left(\frac{1}{n}\right)\\
=&G^p(1-1)^p 0^q + O\left(\frac{1}{n}\right)\\
=&O\left(\frac{1}{n}\right)\,.
\end{aligned}$$

第一步中，我们用二项式公式明确写出了 ΔG – 和 ∇G 的表达式；第二步中，我们将非随机系数和求和符号移到了期望之外；第三步中，我们将求和拆分为嵌入索引 $i_1, \ldots, i_r, j_1, \ldots, j_{2q}$ 互不相同的项（这些项产生了其中所写的、推定为主阶的贡献），以及至少有一对嵌入索引重合的其余项；从组合计数上看，后者至多产生 $O(1/n)$ 个贡献；第四步中，我们提取了第一项中的主导贡献，并将其余部分归入 (ΔG)；

[10] 我们将推导的结果曾在参考文献 [17] 中凭直觉使用；下面将给出稍为严谨一些的分析。

第五步中，我们直接计算了各项期望；第六步中，我们将公因子 G^p 提到了求和号之外；第七步中，我们反向使用了二项式公式；第八步中，我们利用了如下事实：对于任意 $p, q \geq 0$，都有 $0^p 0^q = 0$ $p+q>0$。

尘埃落定后，谨慎地放回样本–词元–区块索引，我们得到

$$F^{(\ell)}_{(\alpha_1;t_1)(\alpha_2;t_2)} = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} s^{(\ell)}_{\alpha_1;t_1;i} s^{(\ell)}_{\alpha_2;t_2;i}\right] = \frac{G^{(\ell)}_{(\alpha_1;t_1)(\alpha_2;t_2)}}{\sqrt{G^{(\ell)}_{(\alpha_1;t_1)(\alpha_1;t_1)}+\epsilon}\sqrt{G^{(\ell)}_{(\alpha_2;t_2)(\alpha_2;t_2)}+\epsilon}} + O\left(\frac{1}{n}\right). \tag{1.53}$$

作为快速合理性检查，当 $(\alpha_1;\, t_1) = (\alpha_2;\, t_2)$ 且 $\epsilon = 0$ 时，我们有 $F^{(\ell)}{}_{(\alpha;\, t)\ (\alpha;\, t)} = 1$，这正如预期，因为 $\frac{1}{n}\sum_{i=1}^{n} \underline{s}_{\alpha;t;i} s_{\alpha;t;i} = 1$ 是按构造得到的。最重要的是，即使我们得到了指数级大的核 $G^{(\ell)} \sim e^{\text{large}}$，它们也会在分子和分母之间相互抵消，从而得到数量级为 1 的数值，如 $F^{(\ell)} \sim \frac{e^{\text{large}}}{\sqrt{e^{\text{large}}+\epsilon}\sqrt{e^{\text{large}}+\epsilon}} = O(1)$ 所示。这样，归一化层便能避免信号问题呈指数级爆炸，至少不会在块与块之间发生。

C. 多头自注意力模块

如 §I 0 C 所述，多头自注意力块的残差路径输出

$$r_{\alpha;t;i} = \sum_{h=1}^{H}\sum_{t'=1}^{T}\sum_{c=1}^{C}\sum_{j=1}^{n} \Omega^h_{\alpha;tt'} U^h_{ic} V^h_{cj} s_{\alpha;t';j}\,, \tag{1.54}$$

其中，自注意力矩阵 $\Omega^h_{\alpha;tt'} = \Omega^h_{\alpha;tt'}\left[\tilde{\Omega}\,(Q,\ \ K;\, s)\right]$ 是查询–键点积矩阵 $\tilde{\Omega}^h_{\alpha\,;\ \bar{t}\bar{t}'}$ (1.16) 的一个一阶函数，其定义如下

$$\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'} \equiv \frac{1}{\sqrt{C}}\sum_{c=1}^{C} q^h_{\alpha;\tilde{t};c} k^h_{\alpha;\tilde{t}';c} = \frac{1}{\sqrt{C}}\sum_{c=1}^{C}\sum_{i_1,i_2=1}^{n} Q^h_{ci_1} K^h_{ci_2} s_{\alpha;\tilde{t};i_1} s_{\alpha;\tilde{t}';i_2}\,. \tag{1.55}$$

[11] 更详细地说，这第四步包含两个小步骤：一个是简单的代数运算，$n(n-1)\cdots(n-r-2q+1)/n^{r+2q} = 1 + O(1/n)$；另一个是将不同神经元之间的相关性视为次要项而忽略，即
$\mathbb{E}\left[z_1^2\cdots z_r^2 z_{r+1}\cdots z_{r+2q}\right] = \mathbb{E}\left[z_1^2\right]\cdots\mathbb{E}\left[z_r^2\right]\mathbb{E}\left[z_{r+1}\right]\cdots\mathbb{E}\left[z_{r+2q}\right] + O(1/n)$。关于后者的更多内容，请参见参考文献 [15]。

我们通过从均值为零、协方差为以下形式的分布中抽样，来初始化相关权重 Q^h_{ci}、 K^h_{ci}、 V^h_{ci} 和 U^h_{ic}

$$\mathbb{E}\left[Q^{h_1}_{c_1 i_1} Q^{h_2}_{c_2 i_2}\right] = \left(\frac{C_Q}{n}\right) \delta_{c_1 c_2} \delta_{i_1 i_2} \delta^{h_1 h_2} \,, \tag{1.56}$$

$$\mathbb{E}\left[K^{h_1}_{c_1 i_1} K^{h_2}_{c_2 i_2}\right] = \left(\frac{C_K}{n}\right) \delta_{c_1 c_2} \delta_{i_1 i_2} \delta^{h_1 h_2} \,, \tag{1.57}$$

$$\mathbb{E}\left[V^{h_1}_{c_1 i_1} V^{h_2}_{c_2 i_2}\right] = \left(\frac{C_V}{n}\right) \delta_{c_1 c_2} \delta_{i_1 i_2} \delta^{h_1 h_2} \,, \tag{1.58}$$

$$\mathbb{E}\left[U^{h_1}_{i_1 c_1} U^{h_2}_{i_2 c_2}\right] = \left(\frac{C_U}{n}\right) \delta_{i_1 i_2} \delta_{c_1 c_2} \delta^{h_1 h_2} \,, \tag{1.59}$$

使用阶数为 1 的初始化超参数 C_Q、 C_K、 C_V 和 C_U。（存在偏置参数时，通常将其初始化为零。）为了了解为何这些缩放是恰当的，我们来计算查询–键点积 (1.55) 和残差路径输出 (1.54) 的统计量。

首先，查询–键点积在初始化时的均值为零，因为

$$\begin{aligned}\mathbb{E}\left[\widetilde{\Omega}^h_{\alpha;tt'}\right] =& \mathbb{E}\left[\frac{1}{\sqrt{C}} \sum_{c=1}^{C} \sum_{i,j=1}^{n} Q^h_{ci} K^h_{cj} s_{\alpha;t;i} s_{\alpha;t';j}\right] \\ =& \frac{1}{\sqrt{C}} \sum_{c=1}^{C} \sum_{i,j=1}^{n} \mathbb{E}\left[Q^h_{ci}\right] \mathbb{E}\left[K^h_{cj}\right] \mathbb{E}\left[s_{\alpha;t;i} s_{\alpha;t';j}\right] = 0 \,,\end{aligned} \tag{1.60}$$

因为这些权重 是从零均值分布中独立抽取的。至于协方差 ，

$$\begin{aligned}\mathbb{E}\left[\widetilde{\Omega}^{h_1}_{\alpha_1;t_1 t'_1} \widetilde{\Omega}^{h_2}_{\alpha_2;t_2 t'_2}\right] =& \frac{1}{C} \sum_{c_1,c_2=1}^{C} \sum_{i_1,i_2,j_1,j_2=1}^{n} \mathbb{E}\left[Q^{h_1}_{c_1 i_1} K^{h_1}_{c_1 j_1} s_{\alpha_1;t_1;i_1} s_{\alpha_1;t'_1;j_1} Q^{h_2}_{c_2 i_2} K^{h_2}_{c_2 j_2} s_{\alpha_2;t_2;i_2} s_{\alpha_2;t'_2;j_2}\right] \\ =& \frac{1}{C} \sum_{c_1,c_2=1}^{C} \sum_{i_1,i_2,j_1,j_2=1}^{n} \frac{C_Q}{n} \frac{C_K}{n} \delta_{c_1 c_2} \delta_{i_1 i_2} \delta_{j_1 j_2} \delta^{h_1 h_2} \mathbb{E}\left[s_{\alpha_1;t_1;i_1} s_{\alpha_1;t'_1;j_1} s_{\alpha_2;t_2;i_2} s_{\alpha_2;t'_2;j_2}\right] \\ =& \delta^{h_1 h_2} C_Q C_K \mathbb{E}\left[\left(\frac{1}{n} \sum_{i=1}^{n} s_{\alpha_1;t_1;i} s_{\alpha_2;t_2;i}\right) \left(\frac{1}{n} \sum_{j=1}^{n} s_{\alpha_1;t'_1;j} s_{\alpha_2;t'_2;j}\right)\right] \\ =& \delta^{h_1 h_2} C_Q C_K F_{(\alpha_1;t_1)(\alpha_2;t_2)} F_{(\alpha_1;t'_1)(\alpha_2;t'_2)} + O\left(\frac{1}{n}\right) \,,\end{aligned} \tag{1.61}$$

其中在最后一步，我们使用八倍结果（1.52）截去了 $1/n$ 修正项。综合而言，我们预期 $\widetilde{\Omega}^h_{tt'}$ 具有零均值和一阶协方差。这里，对查询–键点积进行重缩放至关重要，否则其协方差会随宽度 n 和/或注意力头数 H 以非平凡的方式缩放。[12] 因此，我们预期

[12] 这段冗长数学推导背后的直觉—— C 多个$\sqrt{C}$ 零均值、阶数为 1 的随机数相加后，其尺度仍为 [13]——已在原始 Transformer 论文的脚注 4 中解释。相比之下，Tensor Program V [25] 提出了另一种缩放策略： $1/C$ 采用 $1/\sqrt{C}$ ，其依据是这样的启发式假设：查询向量和键向量在训练过程中某个阶段可能会产生相关性。

自注意力矩阵为一阶量。[13]

其次，将 我们的注意力转向残差路径输出，它在初始化时具有消失的均值 ，

$$\mathbb{E}\left[r_{\alpha;t;i}\right]=\sum_{h=1}^{H}\sum_{t'=1}^{T}\sum_{c=1}^{C}\sum_{j=1}^{n}\mathbb{E}\left[U_{ic}^{h}\right]\mathbb{E}\left[V_{cj}^{h}\right]\mathbb{E}\left[\Omega_{\alpha;tt'}^{h}s_{\alpha;j;t'}\right]=0\,, \tag{1.62}$$

以及一阶协方差，

$$\begin{aligned}
&\mathbb{E}\left[r_{\alpha_1;t_1;i_1}r_{\alpha_2;t_2;i_2}\right] \\
=&\sum_{h_1,h_2=1}^{H}\sum_{t_1',t_2'=1}^{T}\sum_{c_1,c_2=1}^{C}\sum_{j_1,j_2=1}^{n}\mathbb{E}\left[\Omega_{\alpha_1;t_1t_1'}^{h_1}U_{i_1c_1}^{h_1}V_{c_1j_1}^{h_1}s_{\alpha_1;t_1';j_1}\Omega_{\alpha_2;t_2t_2'}^{h_2}U_{i_2c_2}^{h_2}V_{c_2j_2}^{h_2}s_{\alpha_2;t_2';j_2}\right] \\
=&\sum_{h_1,h_2=1}^{H}\sum_{t_1',t_2'=1}^{T}\sum_{c_1,c_2=1}^{C}\sum_{j_1,j_2=1}^{n}\frac{C_U}{n}\frac{C_V}{n}\delta_{c_1c_2}\delta_{i_1i_2}\delta_{j_1j_2}\delta^{h_1h_2}\mathbb{E}\left[\Omega_{\alpha_1;t_1t_1'}^{h_1}s_{\alpha_1;t_1';j_1}\Omega_{\alpha_2;t_2t_2'}^{h_2}s_{\alpha_2;t_2';j_2}\right] \\
=&\delta_{i_1i_2}C_UC_V\frac{C}{n^2}\sum_{h=1}^{H}\sum_{t_1',t_2'=1}^{T}\sum_{j=1}^{n}\mathbb{E}\left[\Omega_{\alpha_1;t_1t_1'}^{h}\Omega_{\alpha_2;t_2t_2'}^{h}s_{\alpha_1;t_1';j}s_{\alpha_2;t_2';j}\right] \\
=&\delta_{i_1i_2}C_UC_V\sum_{t_1',t_2'=1}^{T}\mathbb{E}\left[\left(\frac{1}{H}\sum_{h=1}^{H}\Omega_{\alpha_1;t_1t_1'}^{h}\Omega_{\alpha_2;t_2t_2'}^{h}\right)\left(\frac{1}{n}\sum_{i=1}^{n}s_{\alpha_1;t_1';i}s_{\alpha_2;t_2';i}\right)\right]\,.
\end{aligned} \tag{1.63}$$

这里，每对括号中的项都预期为一阶量级。[14]

总的来说，将跳跃路径加回去并对块指标进行求和，我们得到通过多头自注意力块的如下核变换：

$$\begin{aligned}
G_{(\alpha_1;t_1)(\alpha_2;t_2)}^{(\ell+1)}=&G_{(\alpha_1;t_1)(\alpha_2;t_2)}^{(\ell)} \\
&+C_UC_V\sum_{t_1',t_2'=1}^{T}\mathbb{E}\left[\left(\frac{1}{H}\sum_{h=1}^{H}\Omega_{\alpha_1;t_1t_1'}^{h}\Omega_{\alpha_2;t_2t_2'}^{h}\right)\left(\frac{1}{n}\sum_{i=1}^{n}s_{\alpha_1;t_1';i}^{(\ell)}s_{\alpha_1;t_2';i}^{(\ell)}\right)\right]\,.
\end{aligned} \tag{1.65}$$

特别是，我们预期第二个加法项是关于一阶层归一化核 $F^{(\ell)}$ (1.53)的一阶泛函，因此预期整体核 $G^{(\ell)}$ 线性增长。

[13] 有关自注意力统计量的更多有趣内容，请参见附录 A。[14] 更具体地说，我们可以利用附录 A 的结果将期望分解为

$$\begin{aligned}
&\mathbb{E}\left[\left(\frac{1}{H}\sum_{h=1}^{H}\Omega_{\alpha_1;t_1t_1'}^{h}\Omega_{\alpha_2;t_2t_2'}^{h}\right)\left(\frac{1}{n}\sum_{i=1}^{n}s_{\alpha_1;t_1';i}s_{\alpha_2;t_2';i}\right)\right] \\
=&\frac{1}{H}\sum_{h=1}^{H}\left(\mathbb{E}\left[\Omega_{\alpha_1;t_1t_1'}^{h}\Omega_{\alpha_2;t_2t_2'}^{h}\right]\right)F_{(\alpha_1;t_1')(\alpha_2;t_2')}+O\left(\frac{1}{C}\right)\,,
\end{aligned} \tag{1.64}$$

并进一步将剩余的期望表示为一个 $\left(|\mathcal{D}|T^2\right)$ 维高斯积分。

D. 多层感知机块

如 §I 0 D 中所述，多层感知机块的残差路径输出

$$r_{\alpha;t;i}=\sum_{j=1}^{Mn}X_{ij}\sigma\left(w_{\alpha;t;j}\right)=\sum_{j=1}^{Mn}X_{ij}\sigma\left(\sum_{k=1}^{n}W_{jk}s_{\alpha;t;k}\right). \tag{1.66}$$

我们通过从零均值分布中抽样来初始化相关权重 W_{ij} 和 X_{ij}，其协方差为

$$\mathbb{E}\left[W_{i_1j_1}W_{i_2j_2}\right]=\left(\frac{C_W}{n}\right)\delta_{i_1i_2}\delta_{j_1j_2}\,, \tag{1.67}$$

$$\mathbb{E}\left[X_{i_1j_1}X_{i_2j_2}\right]=\left(\frac{C_X}{Mn}\right)\delta_{i_1i_2}\delta_{j_1j_2}\,. \tag{1.68}$$

下面说明这些缩放为何合适。

首先，在中间层中，我们得到趋于零的均值和一阶协方差：

$$\mathbb{E}\left[w_{\alpha;t;i}\right]=\sum_{j=1}^{n}\mathbb{E}\left[W_{ij}\right]\mathbb{E}\left[s_{\alpha;t;j}\right]=0\,, \tag{1.69}$$

$$\begin{aligned}\mathbb{E}\left[w_{\alpha_1;t_1;i_1}w_{\alpha_2;t_2;i_2}\right]=&\sum_{j_1,j_2=1}^{n}\mathbb{E}\left[W_{i_1j_1}W_{i_2j_2}s_{\alpha_1;t_1;j_1}s_{\alpha_2;t_2;j_2}\right]\\ =&C_W\delta_{i_1i_2}\mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n}s_{\alpha_1;t_1;j}s_{\alpha_2;t_2;j}\right]\\ =&\delta_{i_1i_2}C_WF_{(\alpha_1;t_1)(\alpha_2;t_2)}\,.\end{aligned} \tag{1.70}$$

事实上，在 $1/n$ 展开的主阶上，我们还可以证明，$w_{\alpha;t;i}$ 服从均值为零的高斯分布，其协方差为 $\delta_{i_1i_2}C_WF_{(\alpha_1;t_1)(\alpha_2;t_2)}$ [15]。[15] 特别是，为了计算各种期望值，我们可以使用因式分解公式

$$\mathbb{E}\left[f_1(w_{\alpha_1;t_1;i_1})\cdots f_p(w_{\alpha_p;t_p;i_p})\right]=\left\langle f_1\left(\widetilde{w}_{\alpha_1;t_1}\right)\right\rangle_{C_WF}\cdots\left\langle f_p\left(\widetilde{w}_{\alpha_p;t_p}\right)\right\rangle_{C_WF}+O\left(\frac{1}{n}\right), \tag{1.73}$$

对于不同的嵌入索引 $i_1,\ldots,i_p$ 以及任意函数 $f_1,\ldots,f_p$。这里，$\langle\cdot\rangle_K$ 一般表示一个以核 $K_{(\alpha_1;t_1)(\alpha_2;t_2)}$ 为参数的（$|\mathcal{D}|T$）维高斯积分

$$\begin{aligned}\left\langle f\left(\widetilde{w}\right)\right\rangle_K\equiv&\frac{1}{\sqrt{\det\left(2\pi K\right)}}\int\left[\prod_{\alpha\in\mathcal{D}}\prod_{t=1}^{T}d\widetilde{w}_{\alpha;t}\right]f\left(\widetilde{w}\right)\\ &\times\exp\left[-\frac{1}{2}\sum_{\alpha_1,\alpha_2\in\mathcal{D}}\sum_{t_1,t_2=1}^{T}\widetilde{w}_{\alpha_1;t_1}\left(K^{-1}\right)^{(\alpha_1;t_1)(\alpha_2;t_2)}\widetilde{w}_{\alpha_2;t_2}\right].\end{aligned} \tag{1.74}$$

[15] 感兴趣的读者可参阅文献 [15]，了解多层感知机块的详细介绍，其中涵盖了无限宽度和有限宽度两种情形，远超本文所需的细节。尽管如此，为了让读者对（非）高斯性有所了解，我们来考察

其次，转到残差路径输出，我们同样有均值为零且协方差为一阶的性质：

$$\mathbb{E}\left[r_{\alpha;t;i}\right] = \sum_{j=1}^{Mn} \mathbb{E}\left[X_{ij}\right] \mathbb{E}\left[\sigma\left(w_{\alpha;t;j}\right)\right] = 0\,, \tag{1.75}$$

$$\begin{aligned}\mathbb{E}\left[r_{\alpha_1;t_1;i_1} r_{\alpha_2;t_2;i_2}\right] =& C_X \mathbb{E}\left[\frac{1}{Mn}\sum_{j=1}^{Mn} \sigma\left(w_{\alpha_1;t_1;j}\right)\sigma\left(w_{\alpha_2;t_2;j}\right)\right] \\ =& C_X \left\langle \sigma\left(\widetilde{w}_{\alpha_1;t_1}\right)\sigma\left(\widetilde{w}_{\alpha_2;t_2}\right)\right\rangle_{C_W F} + O\left(\frac{1}{n}\right)\,,\end{aligned} \tag{1.76}$$

其中，最后一步使用了刚刚提到的因式分解公式 (1.73)。

结合这两个操作，加入跳跃路径，并在块指标上取迹，我们得到多层感知机块中的核变换。

$$G^{(\ell+1)}_{(\alpha_1;t_1)(\alpha_2;t_2)} = G^{(\ell)}_{(\alpha_1;t_1)(\alpha_2;t_2)} + C_X \left\langle \sigma\left(\widetilde{w}_{\alpha_1;t_1}\right)\sigma\left(\widetilde{w}_{\alpha_2;t_2}\right)\right\rangle_{C_W F^{(\ell)}} + O\left(\frac{1}{n}\right)\,, \tag{1.77}$$

其中，层归一化核 $F^{(\ell)}$ (1.53) 用前一块的核 $G^{(\ell)}$ 表示。具体而言，与多头自注意力块的情况一样，我们预期第二个加性项由于层归一化而为一阶量，因此核会线性增长。[16]

中间层预激活的四阶矩，

$$\begin{aligned}&\mathbb{E}\left[w_{\alpha_1;t_1;i_1} w_{\alpha_2;t_2;i_2} w_{\alpha_3;t_3;i_3} w_{\alpha_4;t_4;i_4}\right] \\ =& \sum_{j_1,j_2,j_3,j_4=1}^{n} \mathbb{E}\left[W_{i_1 j_1} W_{i_2 j_2} W_{i_3 j_3} W_{i_4 j_4} s_{\alpha_1;t_1;j_1} s_{\alpha_2;t_2;j_2} s_{\alpha_3;t_3;j_3} s_{\alpha_4;t_4;j_4}\right] \\ =& C_W^2 \left\{ \delta_{i_1 i_2}\delta_{i_3 i_4} \mathbb{E}\left[\left(\frac{1}{n}\sum_{j=1}^{n} s_{\alpha_1;t_1;j} s_{\alpha_2;t_2;j}\right)\left(\frac{1}{n}\sum_{k=1}^{n} s_{\alpha_3;t_3;k} s_{\alpha_4;t_4;k}\right)\right]\right. \\ &\quad + \delta_{i_1 i_3}\delta_{i_2 i_4} \mathbb{E}\left[\left(\frac{1}{n}\sum_{j=1}^{n} s_{\alpha_1;t_1;j} s_{\alpha_3;t_3;j}\right)\left(\frac{1}{n}\sum_{k=1}^{n} s_{\alpha_2;t_2;k} s_{\alpha_4;t_4;k}\right)\right] \\ &\quad \left. + \delta_{i_1 i_4}\delta_{i_2 i_3} \mathbb{E}\left[\left(\frac{1}{n}\sum_{j=1}^{n} s_{\alpha_1;t_1;j} s_{\alpha_4;t_4;j}\right)\left(\frac{1}{n}\sum_{k=1}^{n} s_{\alpha_2;t_2;k} s_{\alpha_3;t_3;k}\right)\right]\right\} + O\left(\frac{1}{n}\right) \\ =& \mathbb{E}\left[w_{\alpha_1;t_1;i_1} w_{\alpha_2;t_2;i_2}\right] \mathbb{E}\left[w_{\alpha_3;t_3;i_3} w_{\alpha_4;t_4;i_4}\right] + \mathbb{E}\left[w_{\alpha_1;t_1;i_1} w_{\alpha_3;t_3;i_3}\right] \mathbb{E}\left[w_{\alpha_2;t_2;i_2} w_{\alpha_4;t_4;i_4}\right] \\ &+ \mathbb{E}\left[w_{\alpha_1;t_1;i_1} w_{\alpha_4;t_4;i_4}\right] \mathbb{E}\left[w_{\alpha_2;t_2;i_2} w_{\alpha_3;t_3;i_3}\right] + O\left(\frac{1}{n}\right)\,.\end{aligned} \tag{1.71}$$

在第二个等式中，我们利用了初始化分布的独立同分布性；具体来说，第一项来自 $(i_1,\ j_1) = (i_2,\ j_2)$ 且 $(i_3,\ j_3) = (i_4,\ j_4)$ 的情况，另外两项也类似。 [对于正态分布，这一步是精确的，是维克定理的一个特例；对于非正态分布，当 $i_1 = i_2 = i_3 = i_4$ 且 $j_1 = j_2 = j_3 = j_4$ 时，会缺少一个非高斯项，因此产生与 $(1/n)\delta_{i_1 i_2 i_3 i_4} = (1/n)\delta_{i_1 i_2}\delta_{i_1 i_3}\delta_{i_1 i_4}$ 成正比的贡献。] 在最后一个等式中，我们利用八倍结果 (1.52) 截去了 $1/n$ 修正项。总的来说，关于四阶矩的这一结果可以简洁地概括为

$$\mathbb{E}\left[w_{\alpha_1;t_1;i_1} w_{\alpha_2;t_2;i_2} w_{\alpha_3;t_3;i_3} w_{\alpha_4;t_4;i_4}\right]\big|_{\text{connected}} = O\left(\frac{1}{n}\right)\,. \tag{1.72}$$

[16]相比之下，如果没有归一化层，那么我们得到的结果就会是——例如，对于 ReLU

E. 头部块

视觉：线性分类层

在视觉 Transformer 的头部块中，该块输出最后一个块的预激活值 $z^{(L)}_{\alpha\, t\, ;\, ;\, i} = b^{\text{head}}_i + \sum_{j=1}^{n} W^{\text{head}}_{ij} s^{(L-1)}_{\alpha\, t\, j\, ;\, ;\, 。}$ 我们通常将头部偏置初始化为零 $b^{\text{head}}_i = 0$，而将头部权重 W^{head}_{ij} 从均值为零、协方差为

$$\mathbb{E}\left[W^{\text{head}}_{i_1 j_1} W^{\text{head}}_{i_2 j_2}\right] = \left(\frac{C_{\text{head}}}{n}\right) \delta_{i_1 i_2} \delta_{j_1 j_2} \,. \tag{1.78}$$

和往常一样，输出的均值为零，而其协方差由 b 给出 y

$$\mathbb{E}\left[z^{(L)}_{\alpha_1;t_1;i_1} z^{(L)}_{\alpha_2;t_2;i_2}\right] = \delta_{i_1 i_2} C_{\text{head}} \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} s^{(L-1)}_{\alpha_1;t_1;i} s^{(L-1)}_{\alpha_2;t_2;i}\right] = \delta_{i_1 i_2} C_{\text{head}} F^{(L-1)}_{(\alpha_1;t_1)(\alpha_2;t_2)} \,. \tag{1.79}$$

也就是说，输出核

$$G^{(L)}_{(\alpha_1;t_1)(\alpha_2;t_2)} = C_{\text{head}} F^{(L-1)}_{(\alpha_1;t_1)(\alpha_2;t_2)} \,, \tag{1.80}$$

显然为一阶量。[17] 请注意，由于最后的归一化层，核随 ℓ深度 $G^{(\ell)}$ 线性增长的效应在这最后一步也被抵消了。

语言：词嵌入，转置（并重新缩放）

在语言 Transformer 的头部块中，它输出最后一块的预激活 $z^{(L)}_{\alpha\, t\, ;;i} = \mathcal{N}_{\text{rescale}} \sum_{j=1}^{n} W^{\text{WE}}_{ji} s^{(L-1)}_{\alpha\, t\, j\, ;;。}$ 我们已经将词嵌入参数的协方差指定为数量级为一，即 $\mathbb{E}\left[W^{\text{WE}}_{i_1 j_1} W^{\text{WE}}_{i_2 j_2}\right] = (C_{\text{WE}})\, \delta_{i_1 i_2} \delta_{j_1 j_2}$，因此接下来关注重缩放因子 $\mathcal{N}_{\text{rescale}}$。直观地说，需要这个重缩放因子 $\mathcal{N}_{\text{rescale}}$ 显而易见：由于我们要对 n 个均值为零、数量级为一的随机数求和，因此需要按以下方式重缩放：

$$\mathcal{N}_{\text{rescale}} = \sqrt{\frac{1}{n}} \,, \tag{1.82}$$

激活值——类似于 $C_X \langle \sigma(\widetilde{w})\, \sigma(\widetilde{w}) \rangle_{C_W G^{(\ell)}} \sim \sharp G^{(\ell)}$，这将导致 $G^{(\ell+1)} \sim (1+\sharp) G^{(\ell)}$，进而引发指数级爆炸 $\sim (1+\sharp)^{\ell}$。[17] 如果我们沿词元方向进行均值池化，如 $\widetilde{z}^{(L)}_{\alpha\, ;i} = \frac{1}{T}\sum_{t=1}^{T} z^{(L)}_{\alpha\, t\, ;;i}$ (1.23)，那么实际的输出核为

$$\widetilde{G}^{(L)}_{\alpha_1\alpha_2} = \frac{1}{T^2} \sum_{t_1,t_2=1}^{T} G^{(L)}_{(\alpha_1;t_1)(\alpha_2;t_2)} \,. \tag{1.81}$$

如果在 $t = 0$ 处附加一个类别词元，并将 $\widetilde{z}^{(L)}_{\alpha\, ;\, i} = z^{(L)}_{\alpha\, 0\, ;\, ;\, i}$ 作为输出，那么输出核由 $\widetilde{G}^{(L)}_{\alpha_1\alpha_2} = G^{(L)}_{(\alpha_1;\, 0)(\alpha_2;\, 0)}$ 给出。

从而使输出保持为一阶量。[18] 出乎直觉的是，我们有

$$\begin{aligned}\mathbb{E}\left[z^{(L)}_{\alpha_1;t_1;i_1}z^{(L)}_{\alpha_2;t_2;i_2}\right] =&\mathcal{N}^2_{\text{rescale}}\delta_{i_1i_2}C_{\text{WE}}\sum_{j=1}^{n}\mathbb{E}\left[s^{(L-1)}_{\alpha_1;t_1;j}s^{(L-1)}_{\alpha_2;t_2;j}\right] \\ =&\delta_{i_1i_2}C_{\text{WE}}\mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n}s^{(L-1)}_{\alpha_1;t_1;j}s^{(L-1)}_{\alpha_2;t_2;j}\right] = \delta_{i_1i_2}C_{\text{WE}}F^{(L-1)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\,,\end{aligned} \tag{1.83}$$

其中，在第二个等式中，我们使用了重缩放方案 (1.82)。因此，我们再次得到量级为一的输出核

$$G^{(L)}_{(\alpha_1;t_1)(\alpha_2;t_2)} = C_{\text{WE}}F^{(L-1)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\,. \tag{1.84}$$

实际上，整个讨论都忽略了词干与头部模块之间因权重绑定而产生的微妙层间相关性，相关内容见此处的脚注。[19]

[18] 尽管如此，原始论文 [22] 中并没有这个重缩放因子。例如，参考文献 [4] （不过我们无法追溯到最初的出处）采用了这一 $1/\sqrt{n}$ 输出重缩放因子，而最大更新版本—— $\mathcal{N}_{\text{rescale}} = 1/n$ ——则首次见于参考文献 [25]。 [19] 下一段将简要讨论下面的一般情形，但内容不会特别有启发性，因此我们建议先自行推导一个简单的玩具模型：由三个带有跳跃连接的线性层组成，$z^{(3)}_{\alpha\ t}$；；$_i = \mathcal{N}_{\text{rescale}}\sum_j^n$，$_k\sum_{m=1}^{n_{\text{vocab}}}W^{\text{WE}}_{ji}\left(W^{(2)}_{jk}+\delta_{jk}\right)W^{\text{WE}}_{km}x_{\alpha;t;m}$ 其中，服从正态分布的词嵌入权重 W^{WE}_{ij}的协方差为 C_{WE}，中间块权重 $W^{(2)}_{ij}$ 的协方差为 $C_{W^{(2)}}/n$。于是我们得到$\mathbb{E}\left[z^{(3)}_{\alpha_1}$；$_{t_1}$；$_{i_1}z^{(3)}_{\alpha_2}$；$_{t_2}$；$_{i_2}\right]$
$=\delta_{i_1i_2}C^2_{\text{WE}}\ (C_{W^{(2)}}+1)\ G^{(0)}\ (\alpha_1;\ _{t_1})\ (\alpha_2;\ _{t_2})\ + C^2_{\text{WE}}\left(n+\dfrac{C_{W^{(2)}}}{n}\right)x_{\alpha_1;\ _{t_1};\ _{i_1}x_{\alpha_2;\ _{t_2};\ _{i_2}}}+\dfrac{1}{n}\ (C_{W^{(2)}}+1)$
$(x_{\alpha_1;\ _{t_1};\ _{i_2}x_{\alpha_2;\ _{t_2};\ _{i_1}})$。 (1.85)

这里第一项就是正文中讨论的那一项，其余部分大多可以忽略，除了这一项 $nC^2_{\text{WE}}x_{\alpha_1;\ _{t_1};\ _{i_1}x_{\alpha_2;\ _{t_2};\ _{i_2}}}$ 它来自跳跃路径 $\mathcal{N}_{\text{rescale}}\sum_{j=1}^{n}\sum_{m=1}^{n_{\text{vocab}}}W^{\text{WE}}_{ji}W^{\text{WE}}_{jm}x_{\alpha;t;m}$。乍一看，这个接触项似乎是最主要的贡献，但请注意，主要贡献中有 $\delta_{i_1i_2}$ 个 n_{vocab}非零分量，且量级为一；而这个接触项只有一个非零分量，量级为 n。因此，只要 $n \lesssim n_{\text{vocab}}$，它就不明显构成问题。尽管如此，如果我们想去掉这个接触项，最简单的办法就是直接舍弃第二块中的跳跃路径。

为处理更一般的情形，我们可以先研究关联函数 $\mathbb{E}\left[W^{\text{WE}}_{j_1i_1}W_{\text{WE}j_2i_2}z^{(1)}_{\alpha_1\ ;\ t_1;\ k_1}\cdots z^{(1)}_{\alpha_p\ ;\ t_p;\ k_p}\right]$，并证明对于任意函数 $\mathcal{F}$，只要它对 W^{WE}_{km} 的依赖仅通过第一块预激活 $z^{(1)}_{\alpha\ t\ ;\ ;\ k}$，

$$\mathbb{E}\left[W^{\text{WE}}_{j_1i_1}W^{\text{WE}}_{j_2i_2}\mathcal{F}\right] =\frac{1}{n}C_{\text{WE}}\delta_{i_1i_2}\delta_{j_1j_2}\mathbb{E}\left[\mathcal{F}\right]+\frac{1}{n}C^2_{\text{WE}}\sum_{\widetilde{\alpha}_1,\widetilde{\alpha}_2\in\mathcal{D}}\sum_{\tilde{t}_1,\tilde{t}_2=1}^{T}x_{\widetilde{\alpha}_1;\tilde{t}_1;i_1}x_{\widetilde{\alpha}_2;\tilde{t}_2;i_2}\mathbb{E}\left[\frac{\partial^2\mathcal{F}}{\partial z^{(1)}_{\widetilde{\alpha}_1;\tilde{t}_1;j_1}\partial z^{(1)}_{\widetilde{\alpha}_2;\tilde{t}_2;j_2}}\right].$$

因此，代入 $\mathcal{F} = s^{(L-1)}_{\alpha_1\ t_1\ ;;j_1}s^{(L-1)}_{\alpha_2\ t_2\ ;;j_2}$ 后，第一项给出了正文中讨论的朴素贡献（1.83）

而第二项则产生另外四个贡献

$$\frac{1}{n}C_{\mathrm{WE}}^{2}\sum_{\tilde{t}_1,\tilde{t}_2=1}^{T}x_{\alpha_1;\tilde{t}_1;i_1}x_{\alpha_2;\tilde{t}_2;i_2}\mathbb{E}\left[\left(\sum_{j_1=1}^{n}\frac{\partial s_{\alpha_1;t_1;j_1}^{(L-1)}}{\partial z_{\alpha_1;\tilde{t}_1;j_1}^{(1)}}\right)\left(\sum_{j_2=1}^{n}\frac{\partial s_{\alpha_2;t_2;j_2}^{(L-1)}}{\partial z_{\alpha_2;\tilde{t}_2;j_2}^{(1)}}\right)\right] \tag{1.86}$$

$$+\frac{1}{n}C_{\mathrm{WE}}^{2}\sum_{j_1,j_2=1}^{n}\sum_{\tilde{t}_1,\tilde{t}_2=1}^{T}x_{\alpha_1;\tilde{t}_1;i_2}x_{\alpha_2;\tilde{t}_2;i_1}\mathbb{E}\left[\frac{\partial s_{\alpha_1;t_1;j_2}^{(L-1)}}{\partial z_{\alpha_1;\tilde{t}_1;j_1}^{(1)}}\frac{\partial s_{\alpha_2;t_2;j_2}^{(L-1)}}{\partial z_{\alpha_2;\tilde{t}_2;j_1}^{(1)}}\right]$$

$$+\frac{1}{n}C_{\mathrm{WE}}^{2}\sum_{j_1,j_2=1}^{n}\sum_{\tilde{t}_1,\tilde{t}_2=1}^{T}x_{\alpha_1;\tilde{t}_1;i_1}x_{\alpha_1;\tilde{t}_2;i_2}\mathbb{E}\left[\frac{\partial^2 s_{\alpha_1;t_1;j_1}^{(L-1)}}{\partial z_{\alpha_1;\tilde{t}_1;j_1}^{(1)}\partial z_{\alpha_1;\tilde{t}_2;j_2}^{(1)}}s_{\alpha_2;t_2;j_2}^{(L-1)}\right]$$

$$+\frac{1}{n}C_{\mathrm{WE}}^{2}\sum_{j_1,j_2=1}^{n}\sum_{\tilde{t}_1,\tilde{t}_2=1}^{T}x_{\alpha_2;\tilde{t}_1;i_1}x_{\alpha_2;\tilde{t}_2;i_2}\mathbb{E}\left[s_{\alpha_1;t_1;j_1}^{(L-1)}\frac{\partial^2 s_{\alpha_2;t_2;j_2}^{(L-1)}}{\partial z_{\alpha_2;\tilde{t}_1;j_1}^{(1)}\partial z_{\alpha_2;\tilde{t}_2;j_2}^{(1)}}\right].$$

在这四项中，第一项包含接触项。具体来说，要看出这一点，我们使用链式法则$\partial {s^{(L-1)}_{\alpha\ t}}_{;;j}\ /\partial {z^{(1)}_{\alpha\ \tilde{t}}}_{;;j} = \sum_{k=1}^{n}\left(\partial {s^{(L-1)}_{\alpha\ t}}_{;;j}\ /\partial {z^{(L-1)}_{\alpha\ t}}_{;;k}\right)\left(\partial {z^{(L-1)}_{\alpha\ t}}_{;;k}\ /\partial {z^{(1)}_{\alpha\ \tilde{t}}}_{;;j}\right)$，使用未来助记规则（1.140）提取主导项 $\partial {s^{(L-1)}_{\alpha\ t}}_{;;j}\ /\partial {z^{(L-1)}_{\alpha\ t}}_{;;k} \approx \delta_{jk}/\sqrt{{G^{(L-1)}_{\alpha\ t}}_{\alpha(;)(;t)}+\epsilon}$，最后注意到，$\partial {z^{(L-1)}_{\alpha\ t}}_{;;j}\ /\partial {z^{(1)}_{\alpha\ \tilde{t}}}_{;;j}$ 的主导贡献来自跳跃路径（本质上是因为，残差路径输出 ${\mathrm{R}^{(\ell)}_{t}}_{;j}$ 第 j 个分量中的“j”与第一块预激活 ${z^{(1)}_{\alpha\ t}}_{;;j}$ 第 j 个分量中的“j”没有有意义的关联，因此其贡献被稀释，并按 $1/n$ 的幂次递减）。

2. 神经切核速成入门

在这篇简短的速成入门中，我们将简要介绍神经切线核的概念 [9],，既包括普通随机梯度下降（SGD）优化器（§I 2 A），也包括 AdamW 优化器（§I 2B）；若想快速了解相关内容，可参阅例如参考文献 [15]。本介绍为我们在 §I 3 中对反向路径进行的有效理论分析铺平道路。在那里，我们将探究如何为 Transformer 中每组模型参数缩放相对学习率因子。

A. 普通随机梯度下降

随机梯度下降的更新方程为

$$\theta_\mu(t) = \theta_\mu(t-1) - \eta_t \frac{\partial \mathcal{L}_{\mathcal{A}_t}}{\partial \theta_\mu}\bigg|_{\theta=\theta(t-1)}, \tag{1.87}$$

其中，模型参数索引 μ 遍历该架构中的所有 P 模型参数 θ_μ ； η_t 表示第 t次迭代时的学习率； $\mathcal{L}_{\mathcal{A}_t}$ 表示在第 t次迭代时针对一个小批量 $\mathcal{A}_t$ 计算的损失函数； θ_μ(0)则取自初始化分布，该分布已在 §I 1 中作了详尽讨论。[20] 在这种标准形式中，我们为所有模型参数赋予同一个学习率 η_t ；但从理论上讲，我们很快会发现，随着 Transformer 的规模不断扩大，每组 G 模型参数的学习率都必须采用不同的缩放方式。 [要了解 Transformer 中的模型参数如何划分为不同组，可回看 §I1 中的方程 (1.25)–(1.33)，或预览 §I3 中的方程 (1.99)–(1.116)。] 为此，我们将随机梯度下降更新方程 (1.87) 推广为

$$\theta_\mu(t) = \theta_\mu(t-1) - \eta_t \lambda_{G(\mu)} \frac{\partial \mathcal{L}_{\mathcal{A}_t}}{\partial \theta_\mu}\bigg|_{\theta=\theta(t-1)}, \tag{1.88}$$

其中， $G(\mu)$ 表示第 μ 个模型参数所属的参数组 G ，相对学习率因子 λ_G 可用于平衡架构中各个参数组的更新幅度。具体来说，标准随机梯度下降会简单地将所有 G 的 $\lambda_G = 1$ 设为相同的值，也就是说，对所有模型参数使用统一的学习率；而我们将探讨如何根据宽度对 Transformer 中各个模型参数组的学习率进行精细缩放。

为此，我们考察网络函数 $f_{\alpha;i} \equiv z^{(L)}_{\alpha\, i}$ ；如何更新（这里暂时省略词元索引 t ，以避免符号冲突）。也就是说，对其进行泰勒展开，

[20] 对于我们用字母 t 同时表示迭代索引 t 和词元索引 t，我们深表歉意。

更改 i 将函数按模型参数更新（1.88）展开至一阶，可得

$$
\begin{aligned}
f_{\alpha;i}(t) =& f_{\alpha;i}(t-1) - \eta_t \sum_{\mu=1}^{P} \lambda_{G(\mu)} \left(\frac{\partial \mathcal{L}_{\mathcal{A}_t}}{\partial \theta_\mu} \frac{\partial f_{\alpha;i}}{\partial \theta_\mu} \right) \Big|_{\theta=\theta(t-1)} + \dots \\
=& f_{\alpha;i}(t-1) - \eta_t \sum_{\mu=1}^{P} \lambda_{G(\mu)} \left[\sum_{\widetilde{\alpha} \in \mathcal{A}_t} \sum_{j=1}^{n_{\text{out}}} \frac{\partial \mathcal{L}_{\mathcal{A}_t}}{\partial f_{\widetilde{\alpha};j}} \left(\frac{\partial f_{\widetilde{\alpha};j}}{\partial \theta_\mu} \frac{\partial f_{\alpha;i}}{\partial \theta_\mu} \right) \right] \Big|_{\theta=\theta(t-1)} + \dots \\
=& f_{\alpha;i}(t-1) - \eta_t \sum_{\widetilde{\alpha} \in \mathcal{A}_t} \sum_{j=1}^{n_{\text{out}}} \left(\frac{\partial \mathcal{L}_{\mathcal{A}_t}}{\partial f_{\widetilde{\alpha};j}} H_{(\widetilde{\alpha};j)(\alpha;i)} \right) \Big|_{\theta=\theta(t-1)} + \dots \,.
\end{aligned}
\tag{1.89}
$$

这里，在最后一行中，我们将神经切线核定义为

$$
H_{(\alpha_1;i_1)(\alpha_2;i_2)} \equiv \sum_{\mu=1}^{P} \lambda_{G(\mu)} \frac{\partial f_{\alpha_1;i_1}}{\partial \theta_\mu} \frac{\partial f_{\alpha_2;i_2}}{\partial \theta_\mu} \,. \tag{1.90}
$$

特别地，由于对于一般的损失函数，相对于网络输出 $\partial \mathcal{L}_{\mathcal{A}_t} / \partial f_{\widetilde{\alpha};j}$ 的损失函数导数预期为一阶量，因此在 §I3 中，我们的目标是缩放相对学习率因子 λ_G，使神经切线核 $H_{(\alpha_1;i_1)(\alpha_2;i_2)}$ ——以及由此产生的网络函数变化——在 Transformer 宽度增大时仍保持为一阶量。

顺带一提，原则上，泰勒展开式 (1.89) 中的高阶项“…”可能会产生影响，而且神经切线核本身也可能发生动态变化。不过，这些效应都受到 $1/n$ 抑制，因此在本文所讨论的（主导）阶上可以忽略。[21]

B. AdamW

AdamW [28, 29] 的更新方程——这里再次通过相对学习-~率因子 λ_G's 进行了推广——可以写作

$$
g^\mu(t) \equiv \frac{\partial \mathcal{L}_{\mathcal{A}_t}}{\partial \theta_\mu} \Big|_{\theta=\theta(t-1)} \,, \tag{1.91}
$$

$$
v^\mu(t) = \beta_1 v^\mu(t-1) + (1-\beta_1) g^\mu(t) \,, \tag{1.92}
$$

$$
u^{\mu\mu}(t) = \beta_2 u^{\mu\mu}(t-1) + (1-\beta_2) \left[g^\mu(t) \right]^2 \,, \tag{1.93}
$$

$$
\theta_\mu(t) = (1 - \eta_t \cdot \texttt{wd}) \theta_\mu(t-1) - \widetilde{\lambda}_{G(\mu)} \frac{\eta_t}{\sqrt{\frac{u^{\mu\mu}(t)}{(1-\beta_2^t)}} + \epsilon} \frac{v^\mu(t)}{(1-\beta_1^t)} \,, \tag{1.94}
$$

[21] 关于这些 $1/n$ 效应的更全面讨论，见参考文献 [15]（一些早期工作另见参考文献 [26, 27] ）。更一般地，主导微扰修正随宽度按 $1/n^{1-s}$ [23] 缩放；对于 $s=0$ 处的神经切线缩放策略，该结果简化为 $1/n$ ，而对于 $s=1$ 处的最大更新缩放策略，它始终不属于微扰范围。

其中 $v^\mu(0) = u^{\mu\mu}(0) = 0$，而 $\theta_\mu(0)$ 同样取自初始化分布。这里，优化器超参数 β_1 和 β_2 分别设定逐参数梯度一阶和二阶矩运行平均值的衰减率——大致相当于对过去 $1/(1-\beta_\#)$ 次迭代取平均——而超参数 η_t 和 wd 分别是全局学习率和权重衰减。具体而言，最后一个更新方程 (1.94) 可以理解为根据 $\eta\widetilde{\lambda}_{G(\mu)}/\sqrt{u^{\mu\mu}}$ 调整逐参数学习率，即依据逐参数梯度平方运行平均值的平方根倒数进行调整。[22]

——

让我们对函数变化关于模型参数更新（1.94）作一阶思想泰勒展开，就像之前处理随机梯度下降时那样。接着我们注意到，随机梯度下降与 AdamW 之间的本质区别在于参数更新分母中的 $\sqrt{u^{\mu\mu}} \sim |g^\mu|$ 因子，也就是说，我们得到形式如下的（经 AdamW 修正的）神经切线核

$$\widetilde{H}_{(\alpha_1;i_1)(\alpha_2;i_2)} \equiv \sum_{\mu=1}^{P} \frac{\widetilde{\lambda}_{G(\mu)}}{|g^\mu|} \frac{\partial f_{\alpha_1;i_1}}{\partial \theta_\mu} \frac{\partial f_{\alpha_2;i_2}}{\partial \theta_\mu} \,. \tag{1.95}$$

因此，我们需要对 λ_G 进行缩放，使这个修正后的神经切线核保持为一阶量，从而再次确保网络函数的变化保持为一阶量。

♦♦♦♦

在 §I3 的分析中，为了简化处理，我们将采用以下策略，同时处理 λ_G 随机梯度下降和 λ_G AdamW：对于每一组 G 模型参数，我们将（*i*）首先确定 λ_G 随机梯度下降的相对学习率因子的适当缩放方式，使神经切线核 (1.90) 保持在 1 的量级；（*ii*）然后估计该组 G模型参数的预期逐参数梯度幅值 $|g^\mu|$ 的宽度缩放；（*iii*）据此调整 AdamW 的相对学习率因子 $\lambda_{G(\mu)} \sim \lambda_{G(\mu)}|g^\mu|$，使修正后的神经切线核 (1.95) 保持在 1 的量级。

[22]让我们更详细地解释 AdamW 最后一个更新方程（1.94）中的其他因子：除数 $(1-\beta_1^t)$ 和 $(1-\beta_2^t)$ 分别是矩估计量 v^μ 和 $u^{\mu\mu}$ 的偏差校正项 [28], ；分母中的正则化因子 ϵ 则设定了梯度的截断尺度。也就是说，对于满足 $|g^\mu| \gg \epsilon$ 的模型参数分量 μ，更新基本上是 $\eta \cdot \text{sign}(g^\mu)$；而对于 $|g^\mu| \ll \epsilon$，更新的幅度开始按梯度的大小成比例减小，不过学习率 η/ϵ 会有所放大。谈到 AdamW 优化器的超参数，直观来说，采用本文中的神经切线缩放时，梯度信息在主阶上不会变化，因此我们可以将梯度视为在初始化时固定，并且在主阶理论分析中忽略 β_1 和 β_2 的影响。至于权重衰减，按照参考文献 [15], 第 §10 节脚注中提出的正则化方法完整推导后，我们发现全局学习率与权重衰减的乘积 $\eta_t \cdot$wd 保持为一阶量：粗略地说，用该文的记号，学习率张量带有下标 $\lambda_{\mu\nu}$，而权重衰减张量以 $\sum_{\mu,\,\nu=1}^{P} a^{\mu\nu}\theta_\mu\theta_\nu$ 形式带有上标；$\lambda_{\mu\mu}$ 按 fan in 缩放，而 $a^{\mu\mu}$ 按 fan in 缩放，因此它们的乘积 $\lambda_{\mu\mu}a^{\mu\mu}$ 为一阶量。

3. 神经切线核的统计性质

在前几页中，我们已经确定了神经切线核，

$$\widehat{H}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2;i_2)} \equiv \sum_{\mu=1}^{P} \lambda_{G(\mu)} \frac{\partial z^{(L)}_{\alpha_1;t_1;i_1}}{\partial \theta_\mu} \frac{\partial z^{(L)}_{\alpha_2;t_2;i_2}}{\partial \theta_\mu} \,, \tag{1.96}$$

作为确定每组学习率 λ_G's 合适缩放方式的核心研究对象。也就是说，神经切线核决定了网络函数的更新，而我们希望它保持在 1 的量级，从而在模型规模扩大时不会爆炸或消失。（这里，我们在神经切线核上加了 帽子 [15]，表示我们将研究这个对象在初始化时的统计性质——或者说，在本文中，只研究其均值；同时也恢复了词元指标t_1，t_2，因为此时它们不会与迭代指标 t 冲突。[23])

为了进行分析，我们引入第 ℓ个块的神经切线核

$$\widehat{H}^{(\ell)}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2;i_2)} \equiv \sum_{\mu=1}^{P} \lambda_{G(\mu)} \frac{\partial z^{(\ell)}_{\alpha_1;t_1;i_1}}{\partial \theta_\mu} \frac{\partial z^{(\ell)}_{\alpha_2;t_2;i_2}}{\partial \theta_\mu} \,, \tag{1.97}$$

当 $\ell = L$ 时，它等于实际的神经切线核（1.96）。特别是，这些对象使我们能够进行前向递归分析，其思路与我们在 §I1 中对预激活统计性质的分析非常相似。具体而言，回顾主要的前向方程（1.3），并利用链式法则展开导数，我们得到

$$\begin{aligned}
&\widehat{H}^{(\ell+1)}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2;i_2)} \qquad\qquad (1.98)\\
&= \sum_{\mu\in(\ell+1)\text{-th block}} \lambda_{G(\mu)} \frac{\partial \mathcal{R}^{(\ell+1)}_{\alpha_1;t_1;i_1}}{\partial \theta^{(\ell+1)}_\mu} \frac{\partial \mathcal{R}^{(\ell+1)}_{\alpha_2;t_2;i_2}}{\partial \theta^{(\ell+1)}_\mu}\\
&\quad + \sum_{t_1',t_2'=1}^{T} \sum_{j_1,j_2=1}^{n} \sum_{k_1,k_2=1}^{n} \left(\frac{\partial \mathcal{R}^{(\ell+1)}_{\alpha_1;t_1;i_1}}{\partial s^{(\ell)}_{\alpha_1;t_1';j_1}} \frac{\partial \mathcal{R}^{(\ell+1)}_{\alpha_2;t_2;i_2}}{\partial s^{(\ell)}_{\alpha_2;t_2';j_2}} \right) \left(\frac{\partial s^{(\ell)}_{\alpha_1;t_1';j_1}}{\partial z^{(\ell)}_{\alpha_1;t_1';k_1}} \frac{\partial s^{(\ell)}_{\alpha_2;t_2';j_2}}{\partial z^{(\ell)}_{\alpha_2;t_2';k_2}} \right) \widehat{H}^{(\ell)}_{(\alpha_1;t_1';k_1)(\alpha_2;t_2';k_2)}\\
&\quad + \widehat{H}^{(\ell)}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2;i_2)}\\
&\quad + \sum_{t_1'=1}^{T} \sum_{j_1=1}^{n} \sum_{k_1=1}^{n} \left(\frac{\partial \mathcal{R}^{(\ell+1)}_{\alpha_1;t_1;i_1}}{\partial s^{(\ell)}_{\alpha_1;t_1';j_1}} \right) \left(\frac{\partial s^{(\ell)}_{\alpha_1;t_1';j_1}}{\partial z^{(\ell)}_{\alpha_1;t_1';k_1}} \right) \widehat{H}^{(\ell)}_{(\alpha_1;t_1';k_1)(\alpha_2;t_2;i_2)}\\
&\quad + \sum_{t_2'=1}^{T} \sum_{j_2=1}^{n} \sum_{k_2=1}^{n} \left(\frac{\partial \mathcal{R}^{(\ell+1)}_{\alpha_2;t_2;i_2}}{\partial s^{(\ell)}_{\alpha_2;t_2';j_2}} \right) \left(\frac{\partial s^{(\ell)}_{\alpha_2;t_2';j_2}}{\partial z^{(\ell)}_{\alpha_2;t_2';k_2}} \right) \widehat{H}^{(\ell)}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2';k_2)} \,.
\end{aligned}$$

用文字来说，第一项是来自第（$\ell+1$）个块的模型参数的加性贡献，第二项是残差路径的累积贡献，第三项是经过复制的

[23] 如§I 2所述，神经切线核在初始化时于主阶保持固定 [9, 15]。

项，它来自跳跃路径；最后两项是跳跃路径与残差路径之间的交叉项。（对于 $\ell+1=L$，由于不存在跳跃路径，最后三项不存在。）进一步来说，在累积项中，第一组括号里的乘法因子描述了信号通过第（$\ell+1$）个块时梯度如何变化，而第二组括号里的乘法因子描述了归一化层对梯度的影响。[24]

接下来，我们将逐块采用以下策略：*(i)* 选择随机梯度下降的相对学习率因子 λ_G，使第一个加法项为一阶量；*(ii)* 估计组 $G,\widetilde{}$ 中模型参数的期望逐参数梯度幅值 $|g^\mu|$ 的宽度缩放；*(iii)* 将 AdamW 的相对学习率因子调整为 $\lambda_{G(\mu)} \sim \lambda_{G(\mu)}|g^\mu|$；此外，*(iv)* 递归地证明，我们选择的初始化超参数 (1.25)–(1.33) 会使第二个累积项中的乘法因子为一阶量，同时交叉项在期望下消失。

对于随机梯度下降，对梯度进行这样的神经切线分析，将得到以下相对学习率因子的缩放关系；这与我们对预激活的分析得出初始化超参数的缩放关系 (1.25)–(1.33) 的方式类似：

$$\lambda_{W^{\text{patch}}} = \left(\frac{1}{n_{\text{patch}}}\right)\Lambda_{W^{\text{patch}}} \quad \Big|\Big| \quad \lambda_{W^{\text{WE}}} = (1)\cdot\Lambda_{\text{WE}}\,, \tag{1.99}$$

$$\lambda_{b^{\text{PE}}} = (1)\cdot\Lambda_{\text{PE}}\,, \tag{1.100}$$

$$\lambda_{Q} = \left(\frac{1}{n}\right)\Lambda_{Q}\,, \tag{1.101}$$

$$\lambda_{K} = \left(\frac{1}{n}\right)\Lambda_{K}\,, \tag{1.102}$$

$$\lambda_{V} = \left(\frac{1}{n}\right)\Lambda_{V}\,, \tag{1.103}$$

$$\lambda_{U} = \left(\frac{1}{n}\right)\Lambda_{U}\,, \tag{1.104}$$

$$\lambda_{W} = \left(\frac{1}{n}\right)\Lambda_{W}\,, \tag{1.105}$$

$$\lambda_{X} = \left(\frac{1}{Mn}\right)\Lambda_{X}\,, \tag{1.106}$$

$$\lambda_{W^{\text{head}}} = \left(\frac{1}{n}\right)\Lambda_{W^{\text{head}}}\,, \quad \lambda_{b^{\text{head}}} = (1)\cdot\Lambda_{b^{\text{head}}}\,, \tag{1.107}$$

其中，与 C_G's 和 Λ_G's 一样属于一阶量的初始化超参数，是原则上可以调节、但我们不会调节的一阶量训练超参数。[25] 对于 AdamW，考虑到因子

[24] 这些乘法因子被认定为参考文献 [17] 中的部分雅可比矩阵——此处对其进行了平方和分解。[25] 如果我们采用参考文献 [15, 23] 中的 $\lambda_{\mu\nu}$ 记法，而不是本文使用的 λ_G 记法，那么例如，我们会将第一个方程（1.99）记作 $\lambda_{W^{\text{patch}}_{i_1 j_1} W^{\text{补丁}}_{i_2 j_2}} = \lambda_{W^{\text{patch}}}\delta_{i_1 i_2}\delta_{j_1 j_2} = (\Lambda_{W^{\text{patch}}}/n_{\text{patch}})\delta_{i_1 i_2}\delta_{j_1 j_2}$。稍微更多

$\sim |g^{\mu}|$ 会将上述相对学习率因子修改为

$$\widetilde{\lambda}_{W^{\text{patch}}} = \left(\frac{1}{n_{\text{patch}}\sqrt{n}}\right)\widetilde{\Lambda}_{W^{\text{patch}}} \quad \Big|\Big| \quad \widetilde{\lambda}_{W^{\text{WE}}} = \left(\frac{1}{\sqrt{n}}\right)\widetilde{\Lambda}_{W^{\text{WE}}}\,, \tag{1.108}$$

$$\widetilde{\lambda}_{b^{\text{PE}}} = \left(\frac{1}{\sqrt{n}}\right)\widetilde{\Lambda}_{b^{\text{PE}}}\,, \tag{1.109}$$

$$\widetilde{\lambda}_{Q} = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_{Q}\,, \tag{1.110}$$

$$\widetilde{\lambda}_{K} = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_{K}\,, \tag{1.111}$$

$$\widetilde{\lambda}_{V} = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_{V}\,, \tag{1.112}$$

$$\widetilde{\lambda}_{U} = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_{U}\,, \tag{1.113}$$

$$\widetilde{\lambda}_{W} = \left(\frac{1}{n\sqrt{Mn}}\right)\widetilde{\Lambda}_{W}\,, \tag{1.114}$$

$$\widetilde{\lambda}_{X} = \left(\frac{1}{Mn\sqrt{n}}\right)\widetilde{\Lambda}_{X}\,, \tag{1.115}$$

$$\widetilde{\lambda}_{W^{\text{head}}} = \left(\frac{1}{n\sqrt{n_{\text{out}}}}\right)\widetilde{\Lambda}_{W^{\text{head}}}\,, \quad \widetilde{\lambda}_{b^{\text{head}}} = \left(\frac{1}{\sqrt{n_{\text{out}}}}\right)\widetilde{\Lambda}_{b^{\text{head}}}\,, \tag{1.116}$$

从而确保对于 AdamW，网络更新保持为一阶量。[26] 采用这些选择后，在本节余下部分，我们将递归地证明，初始化时神经切线核的均值具有如下形式

$$\mathbb{E}\left[\widehat{H}^{(\ell)}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2;i_2)}\right] = \delta_{i_1 i_2}\Theta^{(\ell)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\,, \tag{1.117}$$

其中核为一阶量 $\Theta^{(\ell)}{}_{(\alpha_1;t_1)(\alpha_2;t_2)}$。

明确了这些目标后，让我们继续逐块对梯度进行有效理论分析。

A. 词干模块

T为阻止递归分析，我们先分析第一个块的神经切线核 el,

$$\widehat{H}^{(1)}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2;i_2)} = \sum_{\mu\in 1\text{st block}} \lambda_{G(\mu)} \frac{\partial z^{(1)}_{\alpha_1;t_1;i_1}}{\partial\theta^{(1)}_{\mu}} \frac{\partial z^{(1)}_{\alpha_2;t_2;i_2}}{\partial\theta^{(1)}_{\mu}}\,, \tag{1.118}$$

翻译时容易令人困惑的一点是，参考文献 [15] 中的一阶训练超参数（例如 $\lambda^{(\ell)}_{W}$）在本文中会记作 $\Lambda_{W^{(\ell)}}$。[26] 让我们回顾脚注 8，并讨论一般的元原理缩放策略 [23]。对于学习率，存在规范冗余 [23, 24],，但一种实现方式是：对于随机梯度下降，将式 (1.99)–(1.106) 右侧乘以 n^{s} *except*头部一 (1.107)；对于 AdamW，将式 (1.108)–(1.115) 右侧乘以 $n^{\frac{s}{2}}$ *except*头部一 (1.116)。这里，随机梯度下降与 AdamW 之间相差 $n^{\frac{s}{2}}$ 的因子，是因为非头部梯度幅值 $|g^{\mu}|$ 会因头部初始化缩放 (1.34) 而带上一个 $\sqrt{1/n^{s}}$ 因子。

首先分析视觉 Transformer，然后分析语言 Transformer。

视觉：分块嵌入与位置嵌入

对词干块预激活值求导

$$z^{(1)}_{\alpha;t;i} = b^{\text{PE}}_{t;i} + \sum_{j=1}^{n_{\text{patch}}} W^{\text{patch}}_{ij} x_{\alpha;t;j} \tag{1.119}$$

关于词干块模型参数，我们有

$$\frac{\partial z^{(1)}_{\alpha;t;i}}{\partial W^{\text{patch}}_{jk}} = \delta_{ij} x_{\alpha;t;k}\,, \tag{1.120}$$

$$\frac{\partial z^{(1)}_{\alpha;t;i}}{\partial b^{\text{PE}}_{t';j}} = \delta_{tt'}\delta_{ij}\,. \tag{1.121}$$

这些梯度的平方范数为

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n_{\text{patch}}} \frac{\partial z^{(1)}_{\alpha_1;t_1;i_1}}{\partial W^{\text{patch}}_{ij}} \frac{\partial z^{(1)}_{\alpha_2;t_2;i_2}}{\partial W^{\text{patch}}_{ij}} &= \delta_{i_1 i_2} n_{\text{patch}} \left(\frac{1}{n_{\text{patch}}} \sum_{j=1}^{n_{\text{patch}}} x_{\alpha_1;t_1;j} x_{\alpha_2;t_2;j} \right) \\ &= \delta_{i_1 i_2} \left[(n_{\text{patch}}) \cdot G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)} \right]\,,\end{aligned} \tag{1.122}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{n} \frac{\partial z^{(1)}_{\alpha_1;t_1;i_1}}{\partial b^{\text{PE}}_{t;i}} \frac{\partial z^{(1)}_{\alpha_2;t_2;i_2}}{\partial b^{\text{PE}}_{t;i}} = \delta_{i_1 i_2} \left[(1) \cdot \delta_{t_1 t_2} \right]\,, \tag{1.123}$$

其中，中间一步用到了我们回顾的一阶输入核定义 (1.40)。[27] 注意，正如之前所述 (1.117)，这些加性贡献在嵌入方向上是对角的。

对于随机梯度下降，这些梯度对词干块神经切线核 (1.118) 的加性贡献 (1.122) 和 (1.123) 分别乘以 $\lambda_{W^{\text{patch}}}$ 和 $\lambda_{b^{\text{PE}}}$。因此，为了使这些贡献保持为一阶量，我们需要按如下方式缩放相对学习率因子

$$\lambda_{W^{\text{patch}}} = \left(\frac{1}{n_{\text{patch}}} \right) \Lambda_{W^{\text{patch}}}\,, \quad \lambda_{b^{\text{PE}}} = (1) \cdot \Lambda_{b^{\text{PE}}} \quad \text{for} \quad \text{SGD}\,, \tag{1.124}$$

采用数量级为一的超参数 $\Lambda_{W^{\text{patch}}}$ 和 $\Lambda_{b^{\text{PE}}}$。对于 AdamW，我们需要考虑逐参数梯度 $|g^\mu|$ 的幅度。为了估计分块权重的该数值，我们注意到共有 $n \cdot n_{\text{patch}}$ 个分量相加得到数量级为 n_{patch} 的平方范数，因此梯度的每个分量平均按 $|g^\mu| \sim \sqrt{n_{\text{patch}}}/(n \cdot n_{\text{patch}}) = 1/\sqrt{n}$ 的比例缩放。同样，对于

[27] 顺带一提，我们注意到，即使不取期望值，这些关系仍然成立，也就是说，首模块神经切线核是确定性的。

位置嵌入参数，我们估计 $|g^{\mu}| \sim \sqrt{1/n} = 1/\sqrt{n}$。[28] 因此，为抵消 AdamW 优化器中的这些因素，我们设置

$$\widetilde{\lambda}_{W^{\text{patch}}} = \left(\frac{1}{n_{\text{patch}}\sqrt{n}}\right)\widetilde{\Lambda}_{W^{\text{patch}}}\,, \quad \widetilde{\lambda}_{b^{\text{PE}}} = \left(\frac{1}{\sqrt{n}}\right)\widetilde{\Lambda}_{b^{\text{PE}}} \quad \text{for} \quad \text{AdamW}\,, \tag{1.125}$$

$\widetilde{\Lambda}_{W^{\text{patch}}}$ 和 $\widetilde{\Lambda}_{b^{\text{PE}}}$ 采用数量级为一的超参数。

语言：词嵌入和位置嵌入

与视觉情形非常相似，对词干块预激活值

$$z^{(1)}_{\alpha;t;i} = b^{\text{PE}}_{t;i} + \sum_{j=1}^{n_{\text{vocab}}} W^{\text{WE}}_{ij} x_{\alpha;t;j}\,, \tag{1.126}$$

关于词干块模型参数求导，可得

$$\left.\frac{\partial z^{(1)}_{\alpha;t;i}}{\partial W^{\text{WE}}_{jk}}\right|_{\text{stem}} = \delta_{ij} x_{\alpha;t;k}\,, \tag{1.127}$$

$$\frac{\partial z^{(1)}_{\alpha;t;i}}{\partial b^{\text{PE}}_{t';j}} = \delta_{tt'}\delta_{ij}\,. \tag{1.128}$$

无论如何，对于关于词嵌入参数的导数，我们特别指出，这些梯度来自词干模块，因为还会有来自头部块的另一项贡献，将在 §I 3E 中讨论。

那么，这些梯度的平方范数为

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n_{\text{vocab}}} \left.\frac{\partial z^{(1)}_{\alpha_1;t_1;i_1}}{\partial W^{\text{WE}}_{ij}}\right|_{\text{stem}} \left.\frac{\partial z^{(1)}_{\alpha_2;t_2;i_2}}{\partial W^{\text{WE}}_{ij}}\right|_{\text{stem}} &= \delta_{i_1 i_2}\left(\sum_{j=1}^{n_{\text{vocab}}} x_{\alpha_1;t_1;j} x_{\alpha_2;t_2;j}\right) \\ &= \delta_{i_1 i_2}\left[(1)\cdot G^{(0)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\right]\,,\end{aligned} \tag{1.129}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{n} \frac{\partial z^{(1)}_{\alpha_1;t_1;i_1}}{\partial b^{\text{PE}}_{t;i}} \frac{\partial z^{(1)}_{\alpha_2;t_2;i_2}}{\partial b^{\text{PE}}_{t;i}} = \delta_{i_1 i_2}\left[(1)\cdot \delta_{t_1 t_2}\right]\,, \tag{1.130}$$

其中我们回顾了一阶输入核（1.44）的定义。注意，正如所承诺的（1.117），这些加性贡献在嵌入方向上是对角的。

对于随机梯度下降，为确保对词干块神经切线核 (1.118) 的加性贡献为一阶量，我们将相对学习率因子缩放为

$$\lambda_{W^{\text{WE}}} = (1)\cdot\Lambda_{W^{\text{WE}}}\,, \quad \lambda_{b^{\text{PE}}} = (1)\cdot\Lambda_{b^{\text{PE}}} \quad \text{for} \quad \text{SGD}\,, \tag{1.131}$$

[28] 在本文中，我们隐含地关注超参数随宽度变化的缩放，而 *not* 考察其随批量大小 $|\mathcal{A}_t|$、序列长度 T等变化的缩放。

也就是说，我们使它们二者都保持为一阶量。对于 AdamW，我们根据每个参数的梯度幅度进行调整 $|g^{\mu}| \sim \sqrt{1/n} = 1/\sqrt{n}$ ，—这同时适用于词嵌入参数和位置嵌入参数，从而得到

$$\widetilde{\lambda}_{W^{\mathrm{WE}}} = \left(\frac{1}{\sqrt{n}}\right) \widetilde{\Lambda}_{W^{\mathrm{WE}}}, \quad \widetilde{\lambda}_{b^{\mathrm{PE}}} = \left(\frac{1}{\sqrt{n}}\right) \widetilde{\Lambda}_{b^{\mathrm{PE}}} \quad \text{for} \quad \mathrm{AdamW}. \tag{1.132}$$

在这里，估算词嵌入参数各分量的梯度幅值时，我们没有除以因子 n_{vocab}，原因如下：与视觉情形不同，在视觉情形中，梯度分散到所有 n_{patch} 输入分量上；而在这里的语言情形中，由于输入具有独热结构，梯度大多为零，即 $x_{\alpha;t;j} = \delta_{jj_\star}\,(\alpha;\ t)$ 。[29]

B. 层归一化

仿照 §I 1B ，在尘埃落定之前，我们暂时省略样本–词元–区块索引。

为了分析归一化层如何影响梯度，回看神经切线核的前向方程（1.98），似乎我们需要分析乘法因子

$$\frac{\partial s_{j_1}}{\partial z_{k_1}} \frac{\partial s_{j_2}}{\partial z_{k_2}}, \tag{1.133}$$

对于嵌入索引的一般组合（j_1，j_2，k_1，k_2）。幸运的是，由于嵌入方向上的神经切线核具有递归预期的对角形式 (1.117)，我们可以专注于满足 $k_1 = k_2$ 的那些组合。接下来，我们将在下文 §§I 3 C–I 3 E[中递归地证明] ，（推测的）主阶贡献总是呈现以下两种特定线性组合的形式：

$$\frac{1}{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \frac{\partial s_j}{\partial z_k} \frac{\partial s_j}{\partial z_k}, \tag{1.134}$$

and

$$\frac{1}{n^2} \sum_{j_1,j_2=1}^{n} \sum_{k=1}^{n} s_{j_1} s_{j_2} \frac{\partial s_{j_1}}{\partial z_k} \frac{\partial s_{j_2}}{\partial z_k}. \tag{1.135}$$

[29] 然而，随着批量大小 $|\mathcal{A}_t|$ 增大，这一推理就变得愈发站不住脚：当批量大小足够大时，原则上所有梯度都会变为非零，尽管由于典型词汇表中的词元分布不均匀，梯度会呈现不均匀分布 [30]（顺带一提，这使我们无法简单地用 n_{vocab} 的某个幂对 $\widetilde{\lambda}_{W^{\mathrm{WE}}}$ 进行缩放）。头部块中还存在另一个微妙之处：参见脚注 33。总之，我们将批量大小 $|\mathcal{A}_t|$、序列长度 T、词汇表大小 n_{vocab}与词元分布之间相互作用的分析留待未来工作。

我们先来看第一个 (1.134)。机械地展开后，得到

$$\begin{aligned}
&\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{\partial s_j}{\partial z_k}\frac{\partial s_j}{\partial z_k} \qquad (1.136)\\
=&\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left\{\frac{\partial}{\partial z_k}\left[\frac{z_j-\frac{1}{n}\sum_{i=1}^{n}z_i}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}z_i^2-\left(\frac{1}{n}\sum_{i=1}^{n}z_i\right)^2+\epsilon}}\right]\right\}^2\\
=&\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[\frac{1}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}z_i^2-\left(\frac{1}{n}\sum_{i=1}^{n}z_i\right)^2+\epsilon}}\left(\delta_{jk}-\frac{1}{n}-\frac{1}{n}s_js_k\right)\right]^2\\
=&\frac{1}{\left[\sqrt{\frac{1}{n}\sum_{i=1}^{n}z_i^2-\left(\frac{1}{n}\sum_{i=1}^{n}z_i\right)^2+\epsilon}\right]^2}\left[1-\frac{1}{n}-\frac{2}{n}\left(\frac{1}{n}\sum_{j=1}^{n}s_j^2\right)+\frac{2}{n}\left(\frac{1}{n}\sum_{j=1}^{n}s_j\right)^2+\frac{1}{n}\left(\frac{1}{n}\sum_{j=1}^{n}s_j^2\right)^2\right].
\end{aligned}$$

回顾 §I 0 B 中的内容，由于 $\frac{1}{n}\sum_{i=1}^{n}z_i^2=G+\Delta G$ 和 $\left(\frac{1}{n}\sum_{i=1}^{n}z_i\right)^2=\nabla G$ 以及 ΔG 和 ∇G 在主阶上都可以忽略，我们得到

$$\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{\partial s_j}{\partial z_k}\frac{\partial s_j}{\partial z_k}=\frac{1}{\left(\sqrt{G+\epsilon}\right)^2}+O\left(\frac{1}{n}\right). \qquad (1.137)$$

接下来，直接展开第二个式子 (1.135)，我们得到

$$\begin{aligned}
&\frac{1}{n^2}\sum_{j_1,j_2=1}^{n}\sum_{k=1}^{n}s_{j_1}s_{j_2}\frac{\partial s_{j_1}}{\partial z_k}\frac{\partial s_{j_2}}{\partial z_k} \qquad (1.138)\\
=&\sum_{k=1}^{n}\left\{\frac{1}{n}\sum_{j=1}^{n}s_j\frac{\partial}{\partial z_k}\left[\frac{z_j-\frac{1}{n}\sum_{i=1}^{n}z_i}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}z_i^2-\left(\frac{1}{n}\sum_{i=1}^{n}z_i\right)^2+\epsilon}}\right]\right\}^2\\
=&\sum_{k=1}^{n}\left\{\frac{1}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}z_i^2-\left(\frac{1}{n}\sum_{i=1}^{n}z_i\right)^2+\epsilon}}\left[\frac{s_k}{n}-\frac{1}{n}\left(\frac{1}{n}\sum_{j=1}^{n}s_j\right)-\frac{1}{n}\left(\frac{1}{n}\sum_{j=1}^{n}s_j^2\right)s_k\right]\right\}^2\\
=&O\left(\frac{1}{n}\right).
\end{aligned}$$

也就是说，在主阶上，我们可以忽略这一类贡献。

尘埃落定后，将所有指标仔细代回，我们得到

$$\frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{\partial s^{(\ell)}_{\alpha_1;t_1;j}}{\partial z^{(\ell)}_{\alpha_1;t_1;k}}\frac{\partial s^{(\ell)}_{\alpha_2;t_2;j}}{\partial z^{(\ell)}_{\alpha_2;t_2;k}}=\frac{1}{\sqrt{G^{(\ell)}_{(\alpha_1;t_1)(\alpha_1;t_1)}+\epsilon}\sqrt{G^{(\ell)}_{(\alpha_2;t_2)(\alpha_2;t_2)}+\epsilon}}+O\left(\frac{1}{n}\right). \qquad (1.139)$$

请注意，由于我们预计核 $G^{(\ell)}$ 会随深度 ℓ线性增长，因此预计这个乘法因子会受到线性抑制 [17]。

总而言之，回顾过去并展望未来，我们只需使用简单直观的记忆口诀

$$s_{\alpha;t;j}\approx\frac{z_{\alpha;t;j}}{\sqrt{G_{(\alpha;t)(\alpha;t)}+\epsilon}}\quad\text{and}\quad\frac{\partial s_{\alpha;t;j}}{\partial z_{\alpha;t;k}}\approx\frac{1}{\sqrt{G_{(\alpha;t)(\alpha;t)}+\epsilon}}\delta_{jk}\,, \qquad (1.140)$$

由此，我们在 I3 B 中的所有结果以及在 I 1 B 中的结果都可立即得到。

C. 多头自注意力模块

对残差 MHSA 块输出求导

$$r_{\alpha;t;i} = \sum_{h=1}^{H}\sum_{t'=1}^{T}\sum_{c=1}^{C}\sum_{j=1}^{n} \Omega^h_{\alpha;tt'}\left[\widetilde{\Omega}\left(Q,K;s\right)\right] U^h_{ic}V^h_{cj}s_{\alpha;t';j} \tag{1.141}$$

对于 MHSA 块模型参数和输入信号，我们有

$$\frac{\partial r_{\alpha;t;i}}{\partial U^h_{jc}} = \delta_{ij}\sum_{t'=1}^{T}\sum_{k=1}^{n} V^h_{ck}\Omega^h_{\alpha;tt'}s_{\alpha;t';k}\,, \tag{1.142}$$

$$\frac{\partial r_{\alpha;t;i}}{\partial V^h_{cj}} = U^h_{ic}\sum_{t'=1}^{T} \Omega^h_{\alpha;tt'}s_{\alpha;t';\ j}\,, \tag{1.143}$$

$$\frac{\partial r_{\alpha;t;i}}{\partial K^h_{cj}} = \sum_{t'=1}^{T}\sum_{c'=1}^{C}\sum_{k=1}^{n} U^h_{ic'}V^h_{c'k}\left(\sum_{\tilde{t},\tilde{t}'=1}^{T}\frac{\partial\Omega^h_{\alpha;tt'}}{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}\frac{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}{\partial K^h_{cj}}\right)s_{\alpha;t';k}\,, \tag{1.144}$$

$$\frac{\partial r_{\alpha;t;i}}{\partial Q^h_{cj}} = \sum_{t'=1}^{T}\sum_{c'=1}^{C}\sum_{k=1}^{n} U^h_{ic'}V^h_{c'k}\left(\sum_{\tilde{t},\tilde{t}'=1}^{T}\frac{\partial\Omega^h_{\alpha;tt'}}{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}\frac{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}{\partial Q^h_{cj}}\right)s_{\alpha;t';k}\,, \tag{1.145}$$

$$\frac{\partial r_{\alpha;t;i}}{\partial s_{\alpha;t'';k}} = \sum_{t'=1}^{T}\sum_{h=1}^{H}\sum_{c=1}^{C} U^h_{ic}\sum_{j=1}^{n}V^h_{cj}\left[\delta_{jk}\Omega^h_{\alpha;tt''} + \left(\sum_{\tilde{t},\tilde{t}'=1}^{T}\frac{\partial\Omega^h_{\alpha;tt'}}{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}\frac{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}{\partial s_{\alpha;t'';k}}\right)s_{\alpha;t';j}\right]\,, \tag{1.146}$$

其中，对于最后三项，我们使用链式法则展开了自注意力矩阵 $\Omega^h_{\alpha;tt'}$ 的导数。更具体地说，查询–键点积 (1.16) 的导数,

$$\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'} = \frac{1}{\sqrt{C}}\sum_{c=1}^{C} q^h_{\alpha;\tilde{t};c}k^h_{\alpha;\tilde{t}';c} = \frac{1}{\sqrt{C}}\sum_{c=1}^{C}\sum_{i_1,i_2=1}^{n} Q^h_{ci_1}K^h_{ci_2}s_{\alpha;\tilde{t};i_1}s_{\alpha;\tilde{t}';i_2}\,, \tag{1.147}$$

由下式给出

$$\frac{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}{\partial K^h_{cj}} = \frac{1}{\sqrt{C}}\sum_{m=1}^{n} Q^h_{cm}s_{\alpha;\tilde{t};m}s_{\alpha;\tilde{t}';j} = \frac{1}{\sqrt{C}}q^h_{\alpha;\tilde{t};c}s_{\alpha;\tilde{t}';j}\,, \tag{1.148}$$

$$\frac{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}{\partial Q^h_{cj}} = \frac{1}{\sqrt{C}}\sum_{m=1}^{n} K^h_{cm}s_{\alpha;\tilde{t};j}s_{\alpha;\tilde{t}';m} = \frac{1}{\sqrt{C}}s_{\alpha;\tilde{t};j}k^h_{\alpha;\tilde{t}';c}\,, \tag{1.149}$$

$$\begin{aligned}\frac{\partial\widetilde{\Omega}^h_{\alpha;\tilde{t}\tilde{t}'}}{\partial s_{\alpha;t'';k}} &= \frac{1}{\sqrt{C}}\sum_{c=1}^{C}\sum_{m=1}^{n}\left(\delta_{t''\tilde{t}}Q^h_{ck}K^h_{cm}s_{\alpha;\tilde{t}';m} + \delta_{t''\tilde{t}'}Q^h_{cm}K^h_{ck}s_{\alpha;\tilde{t};m}\right) \\ &= \frac{1}{\sqrt{C}}\sum_{c=1}^{C}\left(\delta_{t''\tilde{t}}Q^h_{ck}k^h_{\alpha;\tilde{t}';c} + \delta_{t''\tilde{t}'}q^h_{\alpha;\tilde{t};c}K^h_{ck}\right)\,.\end{aligned} \tag{1.150}$$

期望　　梯度范数平方的值随后由下式给出——回顾 $n = HC$ ——

$$\mathbb{E}\left[\sum_{h=1}^{H}\sum_{c=1}^{C}\sum_{j=1}^{n}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial U_{jc}^{h}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial U_{jc}^{h}}\right] \tag{1.151}$$
$$=\delta_{i_1 i_2}\left\{(n)\cdot C_V\sum_{t_1',t_2'=1}^{T}\mathbb{E}\left[\left(\frac{1}{H}\sum_{h=1}^{H}\Omega_{\alpha_1;t_1t_1'}^{h}\Omega_{\alpha_2;t_2t_2'}^{h}\right)\left(\frac{1}{n}\sum_{j=1}^{n}s_{\alpha_1;t_1';j}s_{\alpha_2;t_2';j}\right)\right]\right\},$$

$$\mathbb{E}\left[\sum_{h=1}^{H}\sum_{c=1}^{C}\sum_{j=1}^{n}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial V_{cj}^{h}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial V_{cj}^{h}}\right] \tag{1.152}$$
$$=\delta_{i_1 i_2}\left\{(n)\cdot C_U\sum_{t_1',t_2'=1}^{T}\mathbb{E}\left[\left(\frac{1}{H}\sum_{h=1}^{H}\Omega_{\alpha_1;t_1t_1'}^{h}\Omega_{\alpha_2;t_2t_2'}^{h}\right)\left(\frac{1}{n}\sum_{j=1}^{n}s_{\alpha_1;t_1';j}s_{\alpha_2;t_2';j}\right)\right]\right\},$$

$$\mathbb{E}\left[\sum_{h=1}^{H}\sum_{c=1}^{C}\sum_{j=1}^{n}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial K_{cj}^{h}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial K_{cj}^{h}}\right] \tag{1.153}$$
$$=\delta_{i_1 i_2}\Bigg\{(n)\cdot C_U C_V\sum_{t_1',t_2',\tilde{t}_1,\tilde{t}_2,\tilde{t}_1',\tilde{t}_2'=1}^{T}\mathbb{E}\Bigg[\left(\frac{1}{H}\sum_{h=1}^{H}\frac{\partial\Omega_{\alpha_1;t_1t_1'}^{h}}{\partial\widetilde{\Omega}_{\alpha_1;\tilde{t}_1\tilde{t}_1'}^{h}}\frac{\partial\Omega_{\alpha_2;t_2t_2'}^{h}}{\partial\widetilde{\Omega}_{\alpha_2;\tilde{t}_2\tilde{t}_2'}^{h}}\right)\left(\frac{1}{C}\sum_{c=1}^{C}q_{\alpha_1;\tilde{t}_1;c}^{h}q_{\alpha_2;\tilde{t}_2;c}^{h}\right)$$
$$\times\left(\frac{1}{n}\sum_{j=1}^{n}s_{\alpha_1;\tilde{t}_1';j}s_{\alpha_2;\tilde{t}_2';j}\right)\left(\frac{1}{n}\sum_{k=1}^{n}s_{\alpha_1;t_1';k}s_{\alpha_2;t_2';k}\right)\Bigg]\Bigg\},$$

$$\mathbb{E}\left[\sum_{h=1}^{H}\sum_{c=1}^{C}\sum_{j=1}^{n}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial Q_{cj}^{h}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial Q_{cj}^{h}}\right] \tag{1.154}$$
$$=\delta_{i_1 i_2}\Bigg\{(n)\cdot C_U C_V\sum_{t_1',t_2',\tilde{t}_1,\tilde{t}_2,\tilde{t}_1',\tilde{t}_2'=1}^{T}\mathbb{E}\Bigg[\left(\frac{1}{H}\sum_{h=1}^{H}\frac{\partial\Omega_{\alpha_1;t_1t_1'}^{h}}{\partial\widetilde{\Omega}_{\alpha_1;\tilde{t}_1\tilde{t}_1'}^{h}}\frac{\partial\Omega_{\alpha_2;t_2t_2'}^{h}}{\partial\widetilde{\Omega}_{\alpha_2;\tilde{t}_2\tilde{t}_2'}^{h}}\right)\left(\frac{1}{C}\sum_{c=1}^{C}k_{\alpha_1;\tilde{t}_1';c}^{h}k_{\alpha_2;\tilde{t}_2';c}^{h}\right)$$
$$\times\left(\frac{1}{n}\sum_{j=1}^{n}s_{\alpha_1;\tilde{t}_1;j}s_{\alpha_2;\tilde{t}_2;j}\right)\left(\frac{1}{n}\sum_{k=1}^{n}s_{\alpha_1;t_1';k}s_{\alpha_2;t_2';k}\right)\Bigg]\Bigg\}.$$

这里，期望中的每个括号对内的项都预期为一阶量，因此梯度范数平方均按 n.[30] 缩放。请注意，正如承诺的那样 (1.117)，这些加性

[30] 具体而言，例如，我们可以利用附录 A 的结果对期望进行因式分解 as

$$\mathbb{E}\left[\left(\frac{1}{H}\sum_{h=1}^{H}\frac{\partial\Omega_{\alpha_1;t_1t_1'}^{h}}{\partial\widetilde{\Omega}_{\alpha_1;\tilde{t}_1\tilde{t}_1'}^{h}}\frac{\partial\Omega_{\alpha_2;t_2t_2'}^{h}}{\partial\widetilde{\Omega}_{\alpha_2;\tilde{t}_2\tilde{t}_2'}^{h}}\right)\left(\frac{1}{C}\sum_{c=1}^{C}k_{\alpha_1;\tilde{t}_1c}^{h}k_{\alpha_2;\tilde{t}_2c}^{h}\right)\left(\frac{1}{n}\sum_{j=1}^{n}s_{\alpha_1;\tilde{t}_1';j}s_{\alpha_2;\tilde{t}_2';j}\right)\left(\frac{1}{n}\sum_{k=1}^{n}s_{\alpha_1;t_1';k}s_{\alpha_2;t_2';k}\right)\right]$$
$$=\mathbb{E}\left[\left(\frac{1}{H}\sum_{h=1}^{H}\frac{\partial\Omega_{\alpha_1;t_1t_1'}^{h}}{\partial\widetilde{\Omega}_{\alpha_1;\tilde{t}_1\tilde{t}_1'}^{h}}\frac{\partial\Omega_{\alpha_2;t_2t_2'}^{h}}{\partial\widetilde{\Omega}_{\alpha_2;\tilde{t}_2\tilde{t}_2'}^{h}}\right)\right]\mathbb{E}\left[\frac{1}{C}\sum_{c=1}^{C}k_{\alpha_1;\tilde{t}_1c}^{h}k_{\alpha_2;\tilde{t}_2c}^{h}\right]F_{(\alpha_1;\tilde{t}_1')(\alpha_2;\tilde{t}_2')}F_{(\alpha_1;t_1')(\alpha_2;t_2')}+O\left(\frac{1}{C}\right) \tag{1.155}$$

其中

$$\mathbb{E}\left[\frac{1}{C}\sum_{c=1}^{C}k_{\alpha_1;\tilde{t}_1c}^{h}k_{\alpha_2;\tilde{t}_2c}^{h}\right]=\mathbb{E}\left[\frac{1}{C}\sum_{c=1}^{C}\sum_{i_1,i_2=1}^{n}K_{ci_1}^{h}K_{ci_2}^{h}s_{\alpha_1;\tilde{t}_1;i_1}s_{\alpha_2;\tilde{t}_2;i_2}\right]=C_K F_{(\alpha_1;\tilde{t}_1)(\alpha_2;\tilde{t}_2)}\,, \tag{1.156}$$

并进一步将剩余的期望表示为一个 $\left(|\mathcal{D}|T^2\right)$ 维高斯积分。

各项贡献在嵌入方向上是对角的。

对于随机梯度下降，为了确保神经切线核中有阶为一的加性贡献，我们因此应将相对学习率因子 λ_G 缩放为

$$\lambda_Q = \left(\frac{1}{n}\right)\Lambda_Q\,, \quad \lambda_K = \left(\frac{1}{n}\right)\Lambda_K\,, \quad \lambda_V = \left(\frac{1}{n}\right)\Lambda_V\,, \quad \lambda_U = \left(\frac{1}{n}\right)\Lambda_U \quad \text{for} \quad \text{SGD}\,. \tag{1.157}$$

对于 AdamW，我们需要根据逐参数梯度的大小进行调整。由于所有这些权重都有 n^2 个分量，其平方范数之和为 n 阶，因此梯度的每个分量按 $|g^\mu| \sim \sqrt{n}/(n^2) = 1/\sqrt{n}$ 缩放。为抵消 AdamW 优化器中的这些因子，我们设定

$$\widetilde{\lambda}_Q = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_Q\,, \quad \widetilde{\lambda}_K = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_K\,, \quad \widetilde{\lambda}_V = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_V\,, \quad \widetilde{\lambda}_U = \left(\frac{1}{n\sqrt{n}}\right)\widetilde{\Lambda}_U \quad \text{for} \quad \text{AdamW}\,. \tag{1.158}$$

最后，转到递归贡献，交叉项按以下方式消失

$$\mathbb{E}\left[\frac{\partial r_{\alpha;t;i}}{\partial s_{\alpha;t'';k}}\mathcal{F}^{(\text{previous})}\right] = 0\,, \tag{1.159}$$

由于信号导数 (1.146) 中残留的零均值权重 U 和 V，其中这里的$\mathcal{F}^{(\text{previous})}$ 是前序块的可观测量函数。对于累积贡献，我们有

$$\begin{aligned}
&\mathbb{E}\left[\sum_{k_1,k_2=1}^{n}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial s_{\alpha_1;t_1'';k_1}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial s_{\alpha_2;t_2'';k_2}}\mathcal{F}^{(\text{previous})}_{k_1k_2}\right] \\
=&\delta_{i_1i_2}C_UC_V\frac{1}{H}\sum_{h=1}^{H}\sum_{t_1',t_2'=1}^{T}\frac{1}{n}\sum_{k_1,k_2=1}^{n} \\
&\times\mathbb{E}\Bigg\{\Bigg[\delta_{k_1k_2}\Omega^h_{\alpha_1;t_1t_1''}\Omega^h_{\alpha_2;t_2t_2''} \\
&\quad+s_{\alpha_2;t_2';k_1}\Omega^h_{\alpha_1;t_1t_1''}\sum_{\tilde{t}_2,\tilde{t}_2'=1}^{T}\frac{\partial\Omega^h_{\alpha_2;t_2t_2'}}{\partial\widetilde{\Omega}^h_{\alpha_2;\tilde{t}_2\tilde{t}_2'}}\frac{1}{\sqrt{C}}\sum_{c_2=1}^{C}\left(\delta_{t_2''\tilde{t}_2}Q^h_{c_2k_2}k^h_{\alpha_2;\tilde{t}_2';c_2}+\delta_{t_2''\tilde{t}_2'}K^h_{c_2k_2}q^h_{\alpha_2;\tilde{t}_2;c_2}\right) \\
&\quad+s_{\alpha_1;t_1';k_2}\Omega^h_{\alpha_2;t_2t_2''}\sum_{\tilde{t}_1,\tilde{t}_1'=1}^{T}\frac{\partial\Omega^h_{\alpha_1;t_1t_1'}}{\partial\widetilde{\Omega}^h_{\alpha_1;\tilde{t}_1\tilde{t}_1'}}\frac{1}{\sqrt{C}}\sum_{c_1=1}^{C}\left(\delta_{t_1''\tilde{t}_1}Q^h_{c_1k_1}k^h_{\alpha_1;\tilde{t}_1';c_1}+\delta_{t_1''\tilde{t}_1'}K^h_{c_1k_1}q^h_{\alpha_1;\tilde{t}_1;c_1}\right) \\
&\quad+n\left(\frac{1}{n}\sum_{j=1}^{n}s_{\alpha_1;t_1';j}s_{\alpha_2;t_2';j}\right)\sum_{\tilde{t}_1,\tilde{t}_1',\tilde{t}_2,\tilde{t}_2'=1}^{T}\frac{\partial\Omega^h_{\alpha_1;t_1t_1'}}{\partial\widetilde{\Omega}^h_{\alpha_1;\tilde{t}_1\tilde{t}_1'}}\frac{\partial\Omega^h_{\alpha_2;t_2t_2'}}{\partial\widetilde{\Omega}^h_{\alpha_2;\tilde{t}_2\tilde{t}_2'}}\frac{1}{C}\sum_{c_1,c_2=1}^{C}\sum_{m_1,m_2=1}^{n} \\
&\qquad\times\left(\delta_{t_1''\tilde{t}_1}Q^h_{c_1k_1}K^h_{c_1m_1}s_{\alpha_1;\tilde{t}_1';m_1}+\delta_{t_1''\tilde{t}_1'}Q^h_{c_1m_1}K^h_{c_1k_1}s_{\alpha_1;\tilde{t}_1;m_1}\right) \\
&\qquad\times\left(\delta_{t_2''\tilde{t}_2}Q^h_{c_2k_2}K^h_{c_2m_2}s_{\alpha_2;\tilde{t}_2';m_2}+\delta_{t_2''\tilde{t}_2'}Q^h_{c_2m_2}K^h_{c_2k_2}s_{\alpha_2;\tilde{t}_2;m_2}\right)\Bigg]\mathcal{F}^{(\text{previous})}_{k_1k_2}\Bigg\}\,.
\end{aligned} \tag{1.160}$$

现在 t 这里的期望中有四项。第一项产生了贡献 on

$$\delta_{i_1 i_2} C_U C_V \frac{1}{H} \sum_{h=1}^{H} \sum_{t_1', t_2'=1}^{T} \mathbb{E}\left[\Omega^h_{\alpha_1; t_1 t_1''} \Omega^h_{\alpha_2; t_2 t_2''} \frac{1}{n} \left(\sum_{k=1}^{n} \mathcal{F}^{(\text{previous})}_{kk} \right) \right], \tag{1.161}$$

它具有前面承诺的层归一化乘法因子形式 (1.134)，并且在嵌入方向上也是对角的 (1.117)。接下来的两个交叉项在这一阶实际上为零。

阶。[31] 对于最后一项，对 Q 和 K 权重进行积分后，我们得到

$$
\begin{aligned}
&\delta_{i_1 i_2} C_U C_V C_Q C_K \frac{1}{H} \sum_{h=1}^{H} \sum_{t_1',\ t_2',\ \tilde{t}_1,\ \tilde{t}_1',\ \tilde{t}_2,\ \tilde{t}_2'=1}^{T} \qquad (1.163)\\
&\times \mathbb{E}\Bigg[\frac{\partial \Omega^h_{\alpha_1;\ t_1 t_1'}}{\partial \widetilde{\Omega}^h_{\alpha_1;\tilde{t}_1\tilde{t}_1'}} \frac{\partial \Omega^h_{\alpha_2;\ t_2 t_2'}}{\partial \widetilde{\Omega}^h_{\alpha_2;\tilde{t}_2\tilde{t}_2'}} \left(\frac{1}{n} \sum_{j=1}^{n} s_{\alpha_1;t_1';j} s_{\alpha_2;t_2';j} \right) \frac{1}{n^2} \sum_{k_1,k_2,m_1,m_2=1}^{n} \\
&\quad \times \Bigg(\delta_{t_1''\tilde{t}_1} \delta_{t_2''\tilde{t}_2} \delta_{k_1 k_2} \delta_{m_1 m_2} s_{\alpha_1;\tilde{t}_1';m_1} s_{\alpha_2;\tilde{t}_2';m_2} + \delta_{t_1''\tilde{t}_1} \delta_{t_2''\tilde{t}_2'} \delta_{k_1 m_2} \delta_{k_2 m_1} s_{\alpha_1;\tilde{t}_1';m_1} s_{\alpha_2;\tilde{t}_2;m_2} \\
&\qquad + \delta_{t_1''\tilde{t}_1'} \delta_{t_2''\tilde{t}_2} \delta_{k_1 m_2} \delta_{k_2 m_1} s_{\alpha_1;\tilde{t}_1;m_1} s_{\alpha_2;\tilde{t}_2';m_2} + \delta_{t_1''\tilde{t}_1'} \delta_{t_2''\tilde{t}_2'} \delta_{k_1 k_2} \delta_{m_1 m_2} s_{\alpha_1;\tilde{t}_1;m_1} s_{\alpha_2;\tilde{t}_2;m_2} \Bigg) \mathcal{F}^{(\text{previous})}_{k_1 k_2} \Bigg] \\
=&\delta_{i_1 i_2} C_U C_V C_Q C_K \frac{1}{H} \sum_{h=1}^{H} \sum_{t_1',t_2',\tilde{t}_1,\tilde{t}_1',\tilde{t}_2,\tilde{t}_2'=1}^{T} \\
&\times \mathbb{E}\Bigg\{ \frac{\partial \Omega^h_{\alpha_1;t_1 t_1'}}{\partial \widetilde{\Omega}^h_{\alpha_1;\tilde{t}_1\tilde{t}_1'}} \frac{\partial \Omega^h_{\alpha_2;t_2 t_2'}}{\partial \widetilde{\Omega}^h_{\alpha_2;\tilde{t}_2\tilde{t}_2'}} \left(\frac{1}{n} \sum_{j=1}^{n} s_{\alpha_1;t_1';j} s_{\alpha_2;t_2';j} \right) \\
&\quad \times \left[\delta_{t_1''\tilde{t}_1} \delta_{t_2''\tilde{t}_2} \left(\frac{1}{n} \sum_{m=1}^{n} s_{\alpha_1;\tilde{t}_1';m} s_{\alpha_2;\tilde{t}_2';m} \right) + \delta_{t_1''\tilde{t}_1'} \delta_{t_2''\tilde{t}_2'} \left(\frac{1}{n} \sum_{m=1}^{n} s_{\alpha_1;\tilde{t}_1;m} s_{\alpha_2;\tilde{t}_2;m} \right) \right] \left(\frac{1}{n} \sum_{k=1}^{n} \mathcal{F}^{(\text{previous})}_{kk} \right) \Bigg\} \\
+&\delta_{i_1 i_2} C_U C_V C_Q C_K \frac{1}{H} \sum_{h=1}^{H} \sum_{t_1',t_2',\tilde{t}_1,\tilde{t}_1',\tilde{t}_2,\tilde{t}_2'=1}^{T} \\
&\times \mathbb{E}\Bigg\{ \frac{\partial \Omega^h_{\alpha_1;t_1 t_1'}}{\partial \widetilde{\Omega}^h_{\alpha_1;\tilde{t}_1\tilde{t}_1'}} \frac{\partial \Omega^h_{\alpha_2;t_2 t_2'}}{\partial \widetilde{\Omega}^h_{\alpha_2;\tilde{t}_2\tilde{t}_2'}} \left(\frac{1}{n} \sum_{j=1}^{n} s_{\alpha_1;t_1';j} s_{\alpha_2;t_2';j} \right) \\
&\quad \times \Bigg[\delta_{t_1''\tilde{t}_1} \delta_{t_2''\tilde{t}_2'} \left(\frac{1}{n^2} \sum_{m_1,m_2=1}^{n} s_{\alpha_1;\tilde{t}_1';m_1} s_{\alpha_2;\tilde{t}_2;m_2} \mathcal{F}^{(\text{previous})}_{m_2 m_1} \right) \\
&\qquad + \delta_{t_1''\tilde{t}_1'} \delta_{t_2''\tilde{t}_2} \left(\frac{1}{n^2} \sum_{m_1,m_2=1}^{n} s_{\alpha_1;\tilde{t}_1;m_1} s_{\alpha_2;\tilde{t}_2';m_2} \mathcal{F}^{(\text{previous})}_{m_2 m_1} \right) \Bigg] \Bigg\}.
\end{aligned}
$$

[31] The argument essentially boils down to doing algebra of the form

$$
\begin{aligned}
&\frac{1}{n} \sum_{k_1,k_2=1}^{n} \mathbb{E}\left[\left(\widetilde{\Omega}^h_{\alpha;tt'} \right) s_{\alpha_2;t_2';k_1} \frac{1}{\sqrt{C}} Q^h_{c_2 k_2} k^h_{\alpha;\tilde{t}_2';c_2} \mathcal{F}^{(\text{previous})}_{k_1 k_2} \right] \qquad (1.162)\\
=&\frac{1}{n} \sum_{k_1,k_2=1}^{n} \frac{1}{C} \sum_{c_2=1}^{C} \mathbb{E}\left[\left(\sum_{c=1}^{C} \sum_{i_1,i_2=1}^{n} Q^h_{ci_1} K^h_{ci_2} s_{\alpha;t;i_1} s_{\alpha;t';i_2} \right) s_{\alpha_2;t_2';k_1} Q^h_{c_2 k_2} \sum_{m_2=1}^{n} K^h_{c_2 m_2} s_{\alpha_2;\tilde{t}_2';m_2} \mathcal{F}^{(\text{previous})}_{k_1 k_2} \right] \\
=&C_Q C_K \frac{1}{n^3} \sum_{i_1,i_2,k_1,k_2,m_2=1}^{n} \mathbb{E}\left[\delta_{i_1 k_2} \delta_{i_2 m_2} s_{\alpha;t;i_1} s_{\alpha;t';i_2} s_{\alpha_2;t_2';k_1} s_{\alpha_2;\tilde{t}_2';m_2} \mathcal{F}^{(\text{previous})}_{k_1 k_2} \right] \\
=&C_Q C_K \mathbb{E}\left[\left(\frac{1}{n} \sum_{m_2=1}^{n} s_{\alpha;t';m_2} s_{\alpha_2;\tilde{t}_2';m_2} \right) \left(\frac{1}{n^2} \sum_{k_1,k_2=1}^{n} s_{\alpha;t;k_2} s_{\alpha_2;t_2';k_1} \mathcal{F}^{(\text{previous})}_{k_1 k_2} \right) \right] = O\left(\frac{1}{n} \right),
\end{aligned}
$$

在最后一个等式中，我们注意到它得出了层归一化乘法因子所承诺的第二种形式 (1.135)，因此可以将其舍去，如 $O(1/n)$。接下来的论证与附录 A 中的类似，只需进行相同的代数运算，不过这里涉及的不是一个查询–键点积 $\widetilde{\Omega}^h_{\alpha;\ tt'}$，而是任意奇数个查询–键点积。

第一项具有层归一化乘法因子所承诺的第一种形式 (1.134)，并且在嵌入方向上也是对角的 (1.117)。与此同时，第二项具有所承诺的第二种形式 (1.135)，因此可以将其舍去，如 $O(1/n)$。

D. 多层感知机块

对残差多层感知机模块输出求导

$$r_{\alpha;t;i} = \sum_{j=1}^{Mn} X_{ij}\sigma\left(w_{\alpha;t;j}\right) = \sum_{j=1}^{Mn} X_{ij}\sigma\left(\sum_{k=1}^{n} W_{jk}s_{\alpha;t;k}\right), \tag{1.164}$$

关于多层感知机模块模型参数和输入信号，我们有

$$\frac{\partial r_{\alpha;t;i}}{\partial X_{jk}} = \delta_{ij}\sigma\left(w_{\alpha;t;k}\right), \tag{1.165}$$

$$\frac{\partial r_{\alpha;t;i}}{\partial W_{jk}} = X_{ij}\sigma'\left(w_{\alpha;t;j}\right)s_{\alpha;t;k}, \tag{1.166}$$

$$\frac{\partial r_{\alpha;t;i}}{\partial s_{\alpha;t;k}} = \sum_{j=1}^{Mn} X_{ij}\sigma'\left(w_{\alpha;t;j}\right)W_{jk}. \tag{1.167}$$

梯度范数平方的期望值为

$$\begin{aligned}\mathbb{E}\left[\sum_{j=1}^{n}\sum_{k=1}^{Mn}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial X_{jk}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial X_{jk}}\right] &= Mn\delta_{i_1i_2}\mathbb{E}\left[\frac{1}{Mn}\sum_{k=1}^{Mn}\sigma\left(w_{\alpha_1;t_1;k}\right)\sigma\left(w_{\alpha_2;t_2;k}\right)\right] \\ &= \delta_{i_1i_2}\left\{(Mn)\cdot\left[\left\langle\sigma\left(\widetilde{w}_{\alpha_1;t_1}\right)\sigma\left(\widetilde{w}_{\alpha_2;t_2}\right)\right\rangle_{C_WF} + O\left(\frac{1}{n}\right)\right]\right\},\end{aligned} \tag{1.168}$$

$$\begin{aligned}\mathbb{E}\left[\sum_{j=1}^{Mn}\sum_{k=1}^{n}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial W_{jk}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial W_{jk}}\right] &= nC_X\delta_{i_1i_2}\mathbb{E}\left\{\left[\frac{1}{Mn}\sum_{j=1}^{Mn}\sigma'\left(w_{\alpha_1;t_1;j}\right)\sigma'\left(w_{\alpha_2;t_2;j}\right)\right]\right. \\ &\qquad\left.\times\left(\frac{1}{n}\sum_{k=1}^{n}s_{\alpha_1;t_1;k}s_{\alpha_2;t_2;k}\right)\right\} \\ &= \delta_{i_1i_2}\left\{(n)\cdot\left[C_X\left\langle\sigma'\left(\widetilde{w}_{\alpha_1;t_1}\right)\sigma'\left(\widetilde{w}_{\alpha_2;t_2}\right)\right\rangle_{C_WF}F_{(\alpha_1;t_1)(\alpha_2;t_2)} + O\left(\frac{1}{n}\right)\right]\right\},\end{aligned} \tag{1.169}$$

其中，我们特别使用因式分解公式 (1.73) 来简化表达式。请注意 意到 ，正如（1.117）所承诺的，这些加性贡献在嵌入方向上是对角的。

对于随机梯度下降，为了确保神经切线核中存在一阶量级的加性贡献，我们因此应将相对学习率因子 λ_G 缩放为

$$\lambda_W = \left(\frac{1}{n}\right)\Lambda_W, \quad \lambda_X = \left(\frac{1}{Mn}\right)\Lambda_X \quad \text{for} \quad \text{SGD}. \tag{1.170}$$

对于 AdamW，我们需要根据逐参数梯度的大小进行调整。对于 W权重，梯度的每个分量按 $|g^\mu| \sim \sqrt{n}/\left(Mn\cdot n\right) = 1/\sqrt{Mn}$ 缩放；而对于 X

权重，$|g^{\mu}| \sim \sqrt{Mn/(n \cdot Mn)} = 1/\sqrt{n_{\circ}}$因此，为了抵消 AdamW 优化器中的这些因素，我们设定

$$\widetilde{\lambda}_W = \left(\frac{1}{n\sqrt{Mn}}\right)\widetilde{\Lambda}_W\,, \quad \widetilde{\lambda}_X = \left(\frac{1}{Mn\sqrt{n}}\right)\widetilde{\Lambda}_X \quad \text{for} \quad \text{AdamW}\,. \tag{1.171}$$

最后，转到递归贡献，交叉项消失，因为

$$\mathbb{E}\left[\frac{\partial r_{\alpha;t;i}}{\partial s_{\alpha;t;j}}\mathcal{F}^{(\text{previous})}\right] = 0\,, \tag{1.172}$$

信号导数 (1.167) 中存在游离的零均值权重 X，其中 $\mathcal{F}^{(\text{previous})}$ 是前序块中可观测量的函数。对于累积贡献，我们有

$$\begin{aligned}
&\mathbb{E}\left[\sum_{k_1,k_2=1}^{n}\frac{\partial r_{\alpha_1;t_1;i_1}}{\partial s_{\alpha_1;t_1;k_1}}\frac{\partial r_{\alpha_2;t_2;i_2}}{\partial s_{\alpha_2;t_2;k_2}}\mathcal{F}^{(\text{previous})}_{k_1k_2}\right] \qquad (1.173)\\
=&C_X\delta_{i_1i_2}\mathbb{E}\left\{\left[\frac{1}{Mn}\sum_{j=1}^{Mn}\sigma'\left(w_{\alpha_1;t_1;j}\right)\sigma'\left(w_{\alpha_2;t_2;j}\right)\right]\left[\sum_{k_1,k_2=1}^{n}W_{jk_1}W_{jk_2}\mathcal{F}^{(\text{previous})}_{k_1k_2}\right]\right\}\\
=&C_XC_W\delta_{i_1i_2}\mathbb{E}\left\{\left[\frac{1}{Mn}\sum_{j=1}^{Mn}\sigma'\left(w_{\alpha_1;t_1;j}\right)\sigma'\left(w_{\alpha_2;t_2;j}\right)\right]\left[\frac{1}{n}\sum_{k=1}^{n}\mathcal{F}^{(\text{previous})}_{kk}\right]\right\} + O\left(\frac{1}{n}\right)\\
=&\delta_{i_1i_2}\left[C_XC_W\left\langle\sigma'\left(\widetilde{w}_{\alpha_1;t_1}\right)\sigma'\left(\widetilde{w}_{\alpha_2;t_2}\right)\right\rangle_{C_WF}\right]\mathbb{E}\left[\frac{1}{n}\sum_{k=1}^{n}\mathcal{F}^{(\text{previous})}_{kk}\right] + O\left(\frac{1}{n}\right)\,.
\end{aligned}$$

这里，在第一个等号中，我们对 X 权重进行了积分；在第二个不等式中，我们对 W 权重进行了积分（其中忽略了层间相关性——参见文献 [15]——将其视为次要的$1/n$ 修正）；在最后一个等号中，我们再次使用因式分解公式 (1.73) 来简化表达式。由此得到所承诺的第一种形式 (1.134) 的层归一化乘法因子，并且该因子在嵌入方向上也是对角的 (1.117)。

E. 头部块

Vision: linear classification layer

对网络输出求导

$$z^{(L)}_{\alpha;t;i} = b^{\text{head}}_i + \sum_{j=1}^{n}W^{\text{head}}_{ij}s^{(L-1)}_{\alpha;t;j} \tag{1.174}$$

with 关于头块模型参数和输入信号，我们有

$$\frac{\partial z^{(L)}_{\alpha;t;i}}{\partial b^{\text{head}}_j} = \delta_{ij}\,, \tag{1.175}$$

$$\frac{\partial z^{(L)}_{\alpha;t;i}}{\partial W^{\text{head}}_{jk}} = \delta_{ij} s^{(L-1)}_{\alpha;t;k}\,, \tag{1.176}$$

$$\frac{\partial z^{(L)}_{\alpha;t;i}}{\partial s^{(L-1)}_{\alpha;t;j}} = W^{\text{head}}_{ij}\,. \tag{1.177}$$

梯度范数平方的期望值因此为

$$\mathbb{E}\left[\sum_{j=1}^{n_{\text{out}}} \frac{\partial z^{(L)}_{\alpha_1;t_1;i_1}}{\partial b^{\text{head}}_j} \frac{\partial z^{(L)}_{\alpha_2;t_2;i_2}}{\partial b^{\text{head}}_j}\right] = \delta_{i_1 i_2} \cdot (1)\,, \tag{1.178}$$

$$\begin{aligned}\mathbb{E}\left[\sum_{j=1}^{n_{\text{out}}}\sum_{k=1}^{n} \frac{\partial z^{(L)}_{\alpha_1;t_1;i_1}}{\partial W^{\text{head}}_{jk}} \frac{\partial z^{(L)}_{\alpha_2;t_2;i_2}}{\partial W^{\text{head}}_{jk}}\right] &= \delta_{i_1 i_2}\mathbb{E}\left[\sum_{k=1}^{n} s^{(L-1)}_{\alpha_1;t_1;k} s^{(L-1)}_{\alpha_2;t_2;k}\right] = \delta_{i_1 i_2} n\ \mathbb{E}\left[\frac{1}{n}\sum_{k=1}^{n} s^{(L-1)}_{\alpha_1;t_1;k} s^{(L-1)}_{\alpha_2;t_2;k}\right] \\ &= \delta_{i_1 i_2}\left[(n)\cdot F^{(L-1)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\right]\,. \end{aligned} \tag{1.179}$$

请注意，正如承诺的那样（1.117），这些加性贡献在嵌入方向上是对角的。

对于随机梯度下降，为了确保对神经切线核的加性贡献为 1 阶，我们因此应将相对学习率因子缩放为

$$\lambda_{b^{\text{head}}} = (1)\cdot\Lambda_{b^{\text{head}}}\,, \quad \lambda_{W^{\text{head}}} = \left(\frac{1}{n}\right)\Lambda_{W^{\text{head}}} \quad \text{for} \quad \text{SGD}\,. \tag{1.180}$$

对于 AdamW，我们需要根据逐参数梯度的大小进行调整。对于偏置，梯度的每个分量按 $|g^\mu| \sim 1/\sqrt{n_{\text{out}}}$ 缩放；类似地，对于权重，$|g^\mu| \sim \sqrt{n}/\sqrt{n \cdot n_{\text{out}}} = 1/\sqrt{n_{\text{out}}}$。因此，为了抵消 AdamW 优化器中的这些因子，我们设定

$$\widetilde{\lambda}_{b^{\text{head}}} = \left(\frac{1}{\sqrt{n_{\text{out}}}}\right)\cdot\widetilde{\Lambda}_{b^{\text{head}}}\,, \quad \widetilde{\lambda}_{W^{\text{head}}} = \left(\frac{1}{n\sqrt{n_{\text{out}}}}\right)\widetilde{\Lambda}_{W^{\text{head}}} \quad \text{for} \quad \text{AdamW}\,. \tag{1.181}$$

最后讨论递归贡献。正如我们在神经切线核的块间递归公式 (1.98) 之后括号中提到的，最后一块中没有跳跃连接—残差交叉项。对于累积贡献，我们有

$$\begin{aligned}\mathbb{E}\left[\sum_{j_1,j_2=1}^{n} \frac{\partial z^{(L)}_{\alpha_1;t_1;i_1}}{\partial s^{(L-1)}_{\alpha_1;t_1;j_1}} \frac{\partial z^{(L)}_{\alpha_2;t_2;i_2}}{\partial s^{(L-1)}_{\alpha_2;t_2;j_2}} \mathcal{F}^{(L-1)}_{j_1 j_2}\right] &= \sum_{j_1,j_2=1}^{n} \mathbb{E}\left[W^{\text{head}}_{i_1 j_1} W^{\text{head}}_{i_2 j_2} \mathcal{F}^{(L-1)}_{j_1 j_2}\right] \\ &= \delta_{i_1 i_2} C_{\text{head}} \mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n} \mathcal{F}^{(L-1)}_{jj}\right]\,. \end{aligned} \tag{1.182}$$

这得到了层归一化乘法因子的第一种预期形式 (1.134)，并且在嵌入方向上也是对角的 (1.117)。

语言：词嵌入，转置（并重新缩放）

对网络输出求导

$$z^{(L)}_{\alpha;t;i} = \mathcal{N}_{\text{rescale}} \sum_{j=1}^{n} W^{\text{WE}}_{ji} s^{(L-1)}_{\alpha;t;j} \,, \tag{1.183}$$

wit对于词嵌入参数和输入信号，我们有 e

$$\left.\frac{\partial z^{(L)}_{\alpha;t;i}}{\partial W^{\text{WE}}_{jk}}\right|_{\text{head}} = \mathcal{N}_{\text{rescale}} \delta_{ik} s^{(L-1)}_{\alpha;t;j} \,, \tag{1.184}$$

$$\frac{\partial z^{(L)}_{\alpha;t;i}}{\partial s^{(L-1)}_{\alpha;t;j}} = \mathcal{N}_{\text{rescale}} W^{\text{WE}}_{ji} \,. \tag{1.185}$$

请注意，词干模块（1.127）也产生了另一项贡献；此处我们将其与头部产生的贡献区分开来。

接着，回顾 $\mathcal{N}_{\text{rescale}} = \sqrt{1/n}$ (1.82)，可得梯度范数平方的期望值为

$$\begin{aligned}\mathbb{E}\left[\sum_{i=1}^{n_{\text{vocab}}}\sum_{j=1}^{n} \left.\frac{\partial z^{(L)}_{\alpha_1;t_1;i_1}}{\partial W^{\text{WE}}_{ji}}\right|_{\text{head}} \left.\frac{\partial z^{(L)}_{\alpha_2;t_2;i_2}}{\partial W^{\text{WE}}_{ji}}\right|_{\text{head}}\right] &= \delta_{i_1 i_2} \mathbb{E}\left[\left(\frac{1}{n}\sum_{j=1}^{n} s^{(L-1)}_{\alpha_1;t_1;j} s^{(L-1)}_{\alpha_2;t_2;j}\right)\right] \\ &= \delta_{i_1 i_2}\left[(1)\cdot F^{(L-1)}_{(\alpha_1;t_1)(\alpha_2;t_2)}\right] ,\end{aligned} \tag{1.186}$$

请注意，正如承诺的那样（1.117），这些加性贡献在嵌入方向上是对角的。[32]

对于随机梯度下降，为了确保神经切线核中阶为一的加性贡献，我们因此应将相对学习率因子缩放为

$$\lambda_{W^{\text{WE}}} = (1)\cdot \Lambda_{W^{\text{WE}}} \quad \text{for} \quad \text{SGD} \,. \tag{1.188}$$

[32] 为完整起见，除了这个加性项（1.186）和累积项之外，我们还需要考虑词干–头部交叉梯度项。正如长脚注 19 所述，要抓住问题的关键，最高效的方法是推导玩具模型 $z^{(3)}_{\alpha\ t\ ;;i} = \mathcal{N}_{\text{rescale}} \sum_{j}^{n}{}_{,k} \sum_{m=1}^{n_{\text{vocab}}} W^{\text{WE}}_{ji} \left(W^{(2)}_{jk} + \delta_{jk}\right) W_{\text{WE}_{km}}\, x_{\alpha;t;m}$，对此我们得到

$$\mathbb{E}\left[\sum_{i=1}^{n_{\text{vocab}}}\sum_{j=1}^{n} \left.\frac{\partial z^{(3)}_{\alpha_1;t_1;i_1}}{\partial W^{\text{WE}}_{ji}}\right|_{\text{head}} \left.\frac{\partial z^{(3)}_{\alpha_2;t_2;i_2}}{\partial W^{\text{WE}}_{ji}}\right|_{\text{stem}}\right] = C_{\text{WE}}\left(1 + \frac{C_{W^{(2)}}}{n}\right) x_{\alpha_1;t_1;i_2} x_{\alpha_2;t_2;i_1} \,, \tag{1.187}$$

其中，首项来自跳跃路径。与我们在脚注 19 中的观察类似，除一个特定分量外，接触贡献均为零；在更一般的情形下也会出现相同的贡献。

对于 AdamW，我们需要根据逐参数梯度 $|g^{\mu}| \sim 1/\sqrt{n}$ 的大小进行调整。因此，为了抵消这些$^{-}$因素，我们设置 AdamW 优化器为。

$$\widetilde{\lambda}_{W^{\mathrm{WE}}} = \left(\frac{1}{\sqrt{n}}\right) \widetilde{\Lambda}_{W^{\mathrm{WE}}} \quad \text{for} \quad \mathrm{AdamW}\,. \tag{1.189}$$

重要的是，这两种缩放都与我们通过分析词干模块的梯度贡献而决定为其设定的缩放（1.131）和（1.132）一致。[33]

最后，转到递归贡献部分，正如我们在神经切线核的块间递归公式（1.98）之后括号中提到的那样，最后一层没有跳跃连接—残差交叉项。对于累积贡献，我们有

$$\begin{aligned}\mathbb{E}\left[\sum_{j_1,j_2=1}^{n} \frac{\partial z^{(L)}_{\alpha_1;t_1;i_1}}{\partial s^{(L-1)}_{\alpha_1;t_1;j_1}} \frac{\partial z^{(L)}_{\alpha_2;t_2;i_2}}{\partial s^{(L-1)}_{\alpha_2;t_2;j_2}} \mathcal{F}^{(L-1)}_{j_1 j_2}\right] &= \mathcal{N}^2_{\mathrm{rescale}} \sum_{j_1,j_2=1}^{n} \mathbb{E}\left[W^{\mathrm{WE}}_{j_1 i_1} W^{\mathrm{WE}}_{j_2 i_2} \mathcal{F}^{(L-1)}_{j_1 j_2}\right] \\ &= \delta_{i_1 i_2} C_{\mathrm{WE}} \mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n} \mathcal{F}^{(L-1)}_{jj}\right]\,.\end{aligned} \tag{1.190}$$

这就得到了所承诺的第一种层归一化乘法因子形式 (1.134)，并且它在嵌入方向上也是对角的 (1.117)。[34]

[33] 同样，正如脚注29中讨论词干模块时所述，相对 AdamW 学习率因子 $\widetilde{\lambda}_{W^{\mathrm{WE}}}$ 随词汇表大小 n_{vocab} 变化的缩放，对于头部块而言颇为微妙，甚至更加微妙。具体来说，使用交叉熵损失时，词嵌入参数 W^{WE}_{ij}的梯度会乘上因子 $p_{\alpha;t;j} - q_{\alpha;t;j}$，其中 $p_{\alpha;t;j}$ 是输出 $z^{(L)}_{\alpha\, t}$ 在嵌入方向 j上的 Softmax 分布$_{;;j}$，而 $q_{\alpha;t;j}$ 是当前任务的目标分布（通常为独热分布）。即使批量大小 $|\mathcal{A}_t|$ 较小，这里仍有一个微妙之处：这些梯度大多虽小但非零（初始化时大致为 $\sim 1/n_{\mathrm{vocab}}$ 量级），其带噪信号会在 AdamW 优化器中被放大——除非选择显式关注 j 方向上排名靠前的少数分量，或通过适当选择 ϵ进行隐式正则化。[34] 我们通常会掩去词干模块与输出头模块的权重绑定所导致的微妙层间相关性；现在，我们以此作为惯常做法的收尾，只需指出，首模块神经切线核 $\widehat{H}^{(1)}{}_{(\alpha_1;t_1;i_1)(\alpha_2;t_2;i_2)}$ 不依赖词嵌入参数（1.129）；并且使用主导阶的层归一化简记（1.140）后，这里无需担心。

II. 实际应用

真是荒唐至极。

Megatron

在本文第一部分中，我们建立了 Transformer 在主阶下的有效理论。具体而言，这项分析提出了初始化和训练超参数随宽度变化的适当缩放方式，这些方式共同确保 Transformer 扩宽时具有良好的极限行为。然而，如果这些建议不能在某些方面改善实践，它们就只是毫无意义的理论。在本文第二部分中，我们将对这些理论建议进行实践检验，训练视觉与语言 Transformer。

在 §II 1 中，我们将训练视觉 Transformer 来完成图像分类任务。对于这类模型，初始化超参数通常（但并非总是）会随宽度进行适当缩放，因此我们将重点比较$\widetilde{\text{标准}}$的统一学习率——也就是 $\lambda_G = 1$ 用我们的术语来说——与学习率的非统一神经切线缩放 (1.108)–(1.116)。在我们的训练设置中，我们将发现，与统一学习率相比，神经切线缩放策略不仅可以提升模型性能，还能降低训练中期尖峰出现的频率。[35]

在 §II 2 中，我们将预训练语言 Transformer，以完成跨度去噪任务。对于它们，我们将对超参数的标准统一（不）缩放方式做出三项调整：*(i)* 将词嵌入参数的初始化超参数从标准值 $C_{\text{WE}} =$(0.02)^2 提高到 $C_{\text{WE}} = 1$；*(ii)* 相应地，网络输出将采用 $\mathcal{N}_{\text{rescale}} = 1/\sqrt{n}$ 进行重新缩放，而非$\widetilde{\text{标准}}$的不缩放方式 $\mathcal{N}_{\text{rescale}} = 1$；以及 *(iii)* 再次按照神经切线缩放策略设定相对学习$\underline{\text{率}}$因子 λ_G。这里，我们的结果比 §II 1 中的结果稍显复杂，但我们仍将其作为支持更大规模的鼓舞性轶事证据。

编程说明 *1*：与本说明第一部分的理论处理方式一致，对于所有模型，我们将移除多头自注意力和多层感知机块中的偏置参数（例如，一些大型语言模型如 PaLM [4] 也采用这种做法），并将归一化层中的逐元素仿射参数固定。编程说明 *2*：对于所有实验，我们将使用 PyTorch 的自动混合精度 [31], ，以更好地反映当前实践并加快训练。

[35] 我们还将针对视觉 Transformer 测试最大更新缩放策略 [24] ，以及结合神经切线缩放策略与最大更新缩放策略的混合方案 [23] 。我们将发现，这两种方案都能减少训练中期尖峰的出现频率；而在本文测试的四种缩放策略中，混合缩放策略的性能最佳，最大更新缩放策略的性能最差。

1. 使用仅编码器 Transformer 进行图像分类

在这里，我们将以图像分类目标训练仅含编码器的视觉 Transformer。[36] 我们将在 §II 1 A 中介绍数据集和任务，在 §II 1 B 中介绍架构超参数，在 §II 1 C 中介绍初始化超参数，并在 §II 1 D 中介绍优化器与训练超参数。随后，我们将在 §II 1 E 中比较学习率的标准均匀缩放与神经切线缩放。

A. 数据集与任务

数据集方面，我们使用 ImageNet-1k 数据集 [18],的 ILSVRC 挑战赛版本 [33] ，其中包含 1000 个不同类别的 RGB 图像。数据集分为训练集和验证集，分别包含 1,281,167 张图像和 50,000 张图像。按照惯例，训练集和验证集中的每张图像都经过预处理：减去均值 $= [0.485, 0.456, 0.406]$ ，再除以标准差 std $= [0.229, 0.224, 0.225]$。

训练目标方面，我们首先通过标准的随机缩放和裁剪将图像调整为 224×224，并进行随机水平翻转；然后使用 RandAugment [34] ($n = 2$,$m = 9$)，并以相同的 50% 概率选择 Mixup [35] ($\alpha = 0.8$) 或 CutMix [36]($\alpha = 1.0$)；最后以 25% 的概率应用一个随机擦除 [37] 矩形，并在块内为每个像素随机赋值。[37] 随后，我们优化模型输出与目标标签之间的交叉熵损失，并采用 0.1 的标签平滑 [39] 。

评估时，验证集中的图像会缩放至 256×256，并进行中心裁剪至 224×224，不使用任何数据增强。随后，我们评估最高类别预测准确率。

B. 架构超参数

总体而言，我们的架构设计遵循原始视觉 Transformer [14]，只有一处不同：如本部分 II 的引言所述，我们关闭了多头自注意力和多层感知机块 [4]中的所有偏置参数，并且不训练归一化层中的逐元素仿射参数。

有了这一差　　鉴于此，我们的视觉 Transformer 架构正是 §I0 中

[36]按照参考文献 [32], ，所有模型均使用以下代码库进行训练：https://github.com/ facebookresearch/convnext；我们在其中加入了自己的视觉 Transformer 模型代码，并将相对学习率因子 $\tilde{\lambda}_G$'s 引入 AdamW 优化器代码。[37] 作为 RandAugment 的完整配置，我们在 timm 中使用 rand-m9-mstd0.5-inc1 [38], ，该配置会施加标准差为 std $= 0.5$ 的幅度噪声。

所描述的 视觉 分支，其图像块维度为 $n_{\text{patch}} = 16 \cdot 16 \cdot 3 = 768$，分词后的图像块数量为 $T = 196$，嵌入维度为 $n = 768$，归一化层正则化为 $\epsilon = 10^{-6}$，多头自注意力头的数量为 $H = 12$，多层感知机倍数为 $M = 4$，多层感知机模块中采用 GELU 激活函数，输出维度为 $n_{\text{out}} = 1000$；在主体部分，堆叠了 12 个编码器，每个编码器由一个双向多头自注意力模块和一个多层感知机块依次组成。

为确保我们的基础一致，请检查这些架构选择是否得到 $P = 768 \cdot 768 + 196 \cdot 768 + 768^2 \cdot (4 + 2 \cdot 4) \cdot 12 + 768 \cdot 1000 + 1000 \approx 86 \cdot 10^6$ 模型参数。

C. 初始化超参数

关于初始化，文献中的做法各不相同。在我们的实验中，对于标准运行和神经切线运行，我们决定在某些方面或多或少遵循 PyTorch 的默认设置；具体而言，在主体部分，对于多头自注意力模块中的权重，我们使用具有 $C_Q = C_K = C_V = 2/(3 + 1) = 1/2$ 和 $C_U = 1/3$ 的零均值均匀分布；对于多层感知机模块中的权重，我们使用具有 $C_W = 2/(4 + 1) = 2/5$，以及 $C_X = 2 \cdot 4/(4 + 1) = 8/5$的零均值均匀分布。[38] 这些作为数量级为一的数值是可以接受的。至于词干与头部模块，我们简单地使用具有 $C_{\text{patch}} = C_{\text{head}} = 1$ 和 $b_i^{\text{head}} = 0$ 的零均值正态分布，同时按照有时采用的做法，将$C_{\text{PE}} = (0.02)^2$ 用于位置嵌入参数。[39]

D. 优化器和训练超参数

总体而言，我们的训练方案几乎完全遵循文献 [32] –中采用的方案，唯一的区别是：为了使针对各自超参数的比较受到的混杂因素略少一些，我们在此消融了 (*i*) 随机深度 [40] 和 (*ii*) 指数移动平均 [41], 。

至于优化器，我们使用 AdamW [28, 29] (1.91)–(1.94)，其参数为（β_1，β_2，ϵ）=(0.9，0.999，10^{-8})；另外请注意，头部块 [32]中的偏置参数不应用权重衰减。

学习率计划方面，我们采用 4096 的批量大小，每轮迭代 312 次，并在前 20 个训练轮次进行线性预热 [42]，随后 [43]采用余弦学习率衰减

[38] 其中大部分直接源自应用标准 Xavier 初始化；我们特别指出，Q–K–V 权重通常被编码为一个线性层，而不是三个线性层；对于 U，协方差中的因子 $1/3$ 源自将权重均匀分布在区间 $[-1/\sqrt{n}, 1/\sqrt{n}]$中。[39] 经过我们的理论阐述后，这一最后选择可能看起来小得不自然，但由于位置嵌入参数的作用类似于偏置参数（即不是乘法性的，而是加法性的），因此原则上我们可以将这一超参数设为任意一阶数量级的数值，包括 $C_{\text{PE}} = 0$；我们认为采用该值是可以接受的。总体而言，我们尽量将相对于文献的改动控制在最低限度。

在接下来的 280 个训练轮次中，也就是说，

$$\eta_t = \begin{cases} \mathtt{lr} \cdot \left(\frac{t}{312\cdot 20}\right) & \text{for} \quad t \le 312\cdot 20\,, \\ \mathtt{lr}_{\min} + (\mathtt{lr} - \mathtt{lr}_{\min}) \cdot \left\{\frac{1}{2} + \frac{1}{2}\cos\left[\frac{(t-312\cdot 20)\pi}{312\cdot 280}\right]\right\} & \text{for} \quad t > 312\cdot 20\,, \end{cases} \tag{2.1}$$

与 $\mathtt{lr}_{\min} = 10^{-6}$.[40] 我们对模型进行完整的 300 个训练轮次；稍后将在 §II 1 E 中讨论整体学习率 lr 和权重衰减 wd 的设置。

~关于分组学习率因子 λ_G，对于标准运行，我们采用标准的~均匀缩放 $\lambda_G = 1$；对于神经切线运行，我们遵循理论建议 (1.108)–(1.116)，但忽略因子 $M = 4$：见表 II1 E。[41] 总体而言，在 $n = 768 \sim 1000$ 的情况下，我们的神经切线缩放实质上就是将位置嵌入参数（以及头部块中的偏置参数）的学习率相对于其他模型参数提高$n \sim 1000$ 倍。

E. 缩放策略比较

这里我们比较使用标准统一学习率训练的模型与使用神经切线缩放学习率训练的模型的性能：下文表 II 1E 给出了简明概要。

AdamW–标准运行		
	初始标准差	相对学习率因子
分块化 W_{ij}^{patch}	$768^{-\frac{1}{2}}$	1
位置嵌入 $b_{t;i}^{\text{PE}}$	0.02	1
Q–K–V–U–W–X 权重	$\sharp n^{-\frac{1}{2}}$	1
头部权重 W_{ij}^{head}	$n^{-\frac{1}{2}}$	1
头部偏置 b_i^{head}	0	1

AdamW–神经切线运行		
	初始标准差	相对学习率因子
分块化 W_{ij}^{patch}	$768^{-\frac{1}{2}}$	$768^{-1}\cdot n^{-\frac{1}{2}}$
位置嵌入 $b_{t;i}^{\text{PE}}$	0.02	$n^{-\frac{1}{2}}$
Q–K–V–U–W–X 权重	$\sharp n^{-\frac{1}{2}}$	$n^{-1}\cdot n^{-\frac{1}{2}}$
头部权重 W_{ij}^{head}	$n^{-\frac{1}{2}}$	$n^{-1}\cdot 1000^{-\frac{1}{2}}$
头部偏置 b_i^{head}	0	$1000^{-\frac{1}{2}}$

AdamW–$\sqrt{\text{神经切线}}$ ·最大更新运行		
	初始标准差	相对学习率因子
分块化 W_{ij}^{patch}	$768^{-\frac{1}{2}}$	$768^{-1}\cdot n^{-\frac{1}{4}}$
位置嵌入 $b_{t;i}^{\text{PE}}$	0.02	$n^{-\frac{1}{4}}$
Q–K–V–U–W–X 权重	$\sharp n^{-\frac{1}{2}}$	$n^{-1}\cdot n^{-\frac{1}{4}}$
头部权重 W_{ij}^{head}	$n^{-\frac{3}{4}}$	$n^{-1}\cdot 1000^{-\frac{1}{2}}$
头部偏置 b_i^{head}	0	$1000^{-\frac{1}{2}}$

AdamW–最大更新运行		
	初始标准差	相对学习率因子
分块化 W_{ij}^{patch}	$768^{-\frac{1}{2}}$	768^{-1}
位置嵌入 $b_{t;i}^{\text{PE}}$	0.02	1.0
Q–K–V–U–W–X 权重	$\sharp n^{-\frac{1}{2}}$	n^{-1}
头部权重 W_{ij}^{head}	n^{-1}	$n^{-1}\cdot 1000^{-\frac{1}{2}}$
头部偏置 b_i^{head}	0	$1000^{-\frac{1}{2}}$

[40] 回过头来看，对于标准缩放运行（$\mathtt{lr}^\star = 0.001$）而言，$\mathtt{lr}_{\min} = 10^{-6}$ 的影响大约只会在最后 ~ 5 个训练轮次中显现；而对于神经切线缩放运行（$\mathtt{lr}^\star = 20$）而言，则只会在最后 ~ 10 次迭代中显现。[41]为了紧密遵循理论建议，我们本应设置 $\widetilde{\lambda}_W = 1/(n\sqrt{Mn})$ 和 $\widetilde{\lambda}_X = 1/(Mn\sqrt{n})$，但我们~ ~最终还是妥协采用了 $\lambda_W = \lambda_X = 1/(n\sqrt{n})$，因为对于更大的语言模型来说，这种方式更容易实现——分片和展平有时会使编码变得复杂。不过，如果将来有人把 M 扩展到很大规模，那么就应该用 M对 $\widetilde{\lambda}_W$ 和 $\widetilde{\lambda}_X$ 进行适当缩放。

为进行尽可能严谨的比较，我们针对每种缩放策略调整整体学习率 lr 和权重衰减 wd：见图 1。具体而言，我们在 $\log_2$ 网格空间中搜索最优的 (lr, wd)；找到候选最优值 $(\text{lr}^\star, \text{wd}^\star)$ 后，我们确保 $(\text{lr}, \text{wd}) = (2 \cdot \text{lr}^\star, \text{wd}^\star)$、$(\text{lr}^\star, 2 \cdot \text{wd}^\star)$、$(2 \cdot \text{lr}^\star, 0.5 \cdot \text{wd}^\star)$、$(0.5 \cdot \text{lr}^\star, 2 \cdot \text{wd}^\star)$、$(\text{lr}^\star, 0.5 \cdot \text{wd}^\star)$、$(0.5 \cdot \text{lr}^\star, \text{wd}^\star)$ 对应的 top-one 验证准确率都更低。[42] 为每种缩放策略找到最优组合 $(\text{lr}^\star, \text{wd}^\star)$ 后，我们使用三个不同的随机种子运行实验：见图 2。

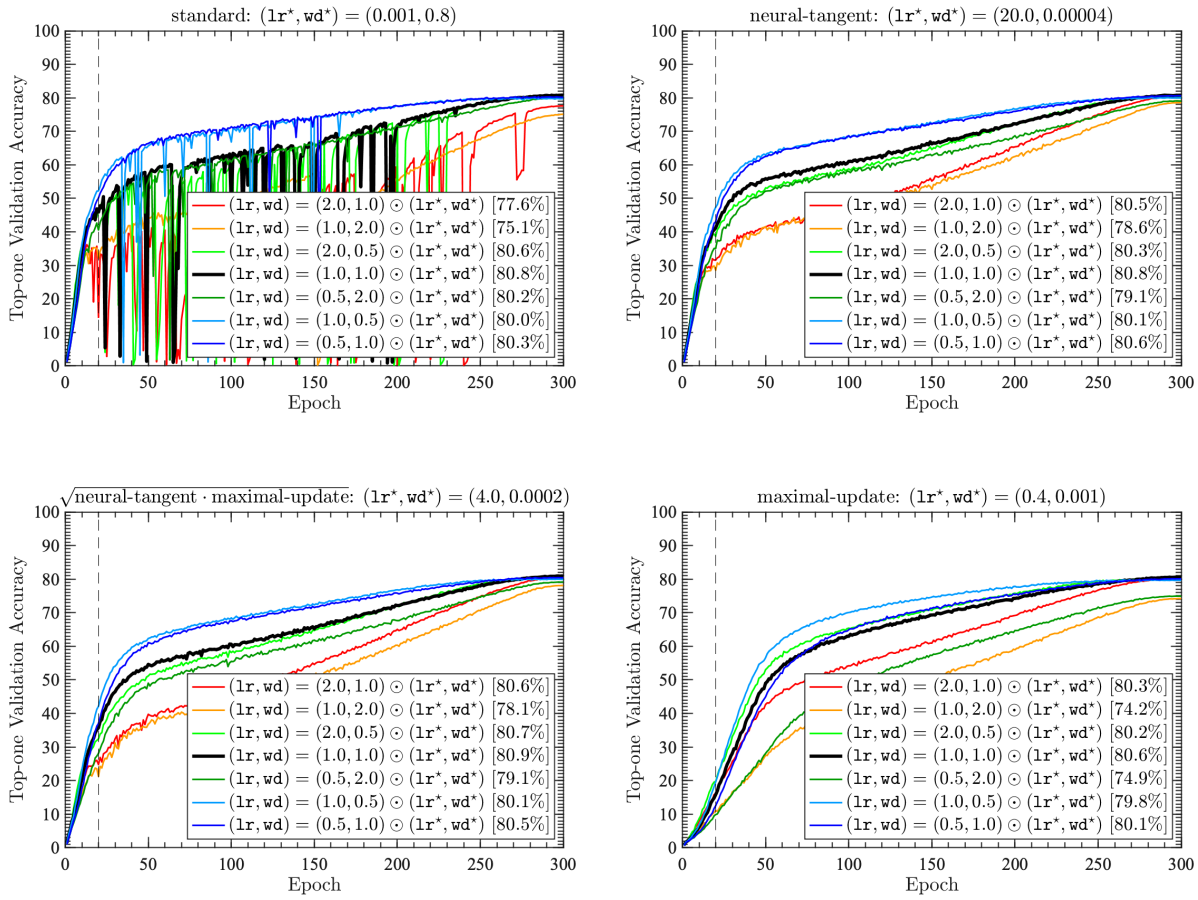


图 1：使用 AdamW$_{(\beta_1,\beta_2,\epsilon)=(0.9,0.999,1\text{e-}8)}$训练的视觉 Transformer 在标准统一缩放策略（左上）、神经切线缩放策略（右上）、混合神经切线—最大更新缩放策略（左下）和最大更新缩放策略（右下）下的超参数搜索。每种策略都绘制了 ImageNet-1k 数据集上的最高验证准确率随训练轮次变化的曲线。图例中记录了各条训练轨迹上最高验证准确率对应的训练超参数对（lr,wd） []。

[42] 为节省计算资源，这里省略 $(\text{lr}, \text{wd}) = (2 \cdot \text{lr}^\star, 2 \cdot \text{wd}^\star), (0.5 \cdot \text{lr}^\star, 0.5 \cdot \text{wd}^\star)$，因为如果将 lr 和 wd *each individually* 加倍或减半会降低性能，那么我们预计将 *both together* 加倍或减半也会降低性能。

总体而言，在我们的实验设置中，与标准缩放策略相比，神经切线缩放策略不仅在统计意义上提升了性能，还降低了训练中期尖峰的出现频率。[43]

此外我们指出，对于神经切线运行，在最优状态下， Q–K–V–U–W–X 权重的有效整体学习率 $\mathrm{lr}\cdot\widetilde{\lambda}_G$ 由 $20\cdot 768^{-3/2}\approx 0.94\cdot 10^{-4}\sim 0.001$ 给出，这基本等于标准运行的最优学习率；我们还指出，两种缩放策略下 $\mathrm{lr}^\star\cdot\mathrm{wd}^\star = 8\cdot 10^{-4}$。因此我们得出结论，神经切线缩放策略带来的稳定性和性能提升，本质上源于提高 位置嵌入参数的学习率，幅度为 $n\sim 1000$ 倍。[44]

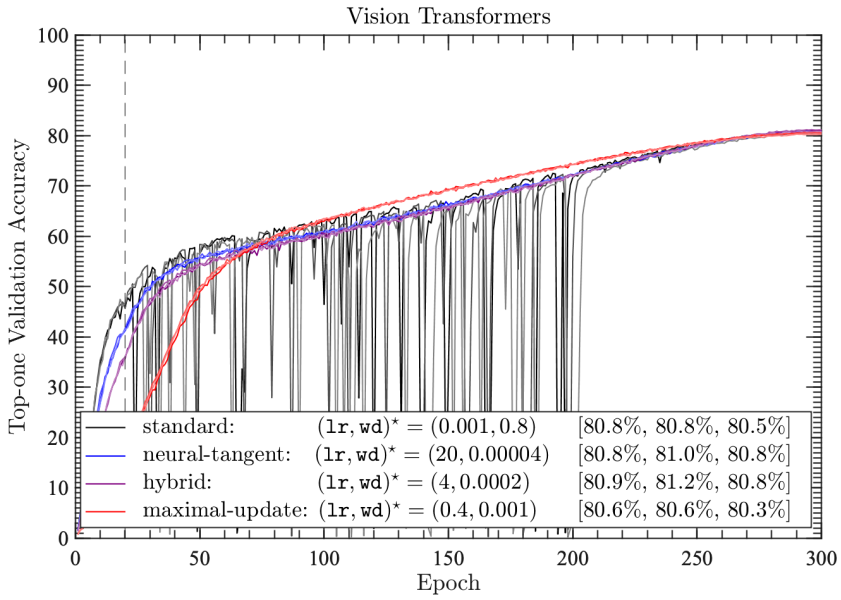


图 2：采用 AdamW 训练的视觉 Transformer 在标准（黑色）、神经切线（蓝色）、神经切线–最大更新混合（紫色）和最大更新（红色）缩放策略下的比较$_{(\beta_1,\beta_2,\epsilon)=(0.9,0.999,1\text{e-}8)}$。图中绘出了每种策略下随训练轮次变化的验证准确率，分别对应三个不同的随机种子（颜色深浅不同）。图例中记录了每种缩放策略对应的最优训练超参数对（$\mathrm{lr}^\star$,$\mathrm{wd}^\star$） [，其数值取各自轨迹中的最高验证准确率]。

[43] 我们还观察到，神经切线–最大更新混合策略和最大更新缩放策略同样能够减少尖峰：参见表 II 1 E 以及图 1 和图 2。顺便一提，在我们的实验设置中，在四种缩放策略中，混合缩放策略的性能最佳，而最大更新缩放策略的性能最差。[44] 严格来说，我们还提高了头部块中偏置的学习率。不过，我们也进行了关闭头部块偏置的实验，观察到的结果在定性上相似，因此我们认为，所声称的益处源于我们对位置嵌入参数的恰当处理。

2. 使用编码器–解码器 Transformer 进行跨度去噪

这里我们将使用跨度去噪目标对编码器—解码器 Language Transformer 进行预训练。[45]我们将在 §II 2 A 中介绍数据集和任务，在 §II 2 B 中介绍架构超参数，在 §II 2 C 中介绍初始化超参数，并在 §II 2 D 中介绍优化器与训练超参数。随后，我们将在 §II 2 E 中比较超参数的标准均匀缩放与神经切线缩放。

A. 数据集和任务

关于数据集，我们遵循参考文献 [45], ，使用由 RoBERTa [47] 所用语料库组合而成的数据集——这些语料库本身由 BookCorpus [48] 、英文维基百科 [49], 、CommonCrawl News [50], 、OpenWebText [51, 52], 和 Stories [53] 汇集而成——并加入 CC-100 语料库 [54] 的英文子集，以及由第三方整理的 pushshift.io Reddit 数据转储 [55, 56]。我们采用与 GPT-2 [52] 相同的字节对编码分词器。对于训练集和验证集，每个输入样本 α 都是从这些文档中抽取的一组句子，总序列长度最多为 512；每组句子始终从某个句子的开头开始，并在某个句子的末尾结束（这两个句子可以相同，也可以不同），同时分别添加一个 [start]词元和一个 [end] 词元；此外，我们还补入 [pad] 词元，使完整序列长度始终为 514。

关于训练目标，遵循文献 [44], ，我们首先通过随机打乱句子来破坏文档，然后使用一个 [MASK]词元反复填充一段标记——其中每段长度均从均值和方差为 3 的泊松分布中随机采样——直到输入序列中 30% 的标记被破坏。随后，我们优化模型输出与原始未破坏样本之间的交叉熵损失，并丢弃 [pad]。

在评估时，我们以相同方式破坏文档并计算交叉熵，不过这次使用的是验证集。

B. 架构超参数

总体而言，我们的架构设计遵循 BART-large [44] 和 R2C2 [45], ，但有一点不同：正如本论文第二部分的引言所述，我们关闭了多头自注意力中的所有偏置参数

[45]遵循文献 [44, 45], ，所有模型均使用 Fairseq [46] 进行训练，Fairseq 可从 https://github.com/facebookresearch/fairseq 获取。

和多层感知机块 cks [4]并且不训练归一化层中的逐元素仿射参数。[4] [6]

考虑到这一差异，我们的 BART-large (R2C2) 架构与 §I0 的语言部分中所述的架构几乎 相同——见下一段——其词汇表大小为 $n_{\text{vocab}} = 50265$，序列长度为 $T = 514$，宽度为 $n = 1024$ (2048)，归一化层正则化为$\epsilon = 10^{-5}$，注意力头数量为 $H = 16$ (32)，多层感知机倍数为 $M = 4$，且多层感知机模块中采用 GELU 激活函数；整体上，首先堆叠 12 (22) 个编码器——每个编码器由一个双向多头自注意力模块和一个多层感知机块组成——然后堆叠 12 (22) 个解码器——每个解码器由一个双向多头自注意力模块、一个即将描述的编码器–解码器多头混合注意力块，以及一个多层感知机块组成。

与 §I0 的 语言 部分所述内容相比，BART-large 和 R2C2 架构还包含一些额外组件：

• 在编码器块堆栈之后、解码器块堆栈之前，还设置了一组额外的位置嵌入参数，通常将其视为解码器堆栈中的词干模块。

• 除了应用于每条残差路径起始处的归一化层外，还额外设置了三处归一化层*(1)* 编码器堆栈词干模块中的第一组位置嵌入参数之后，*(2)* 所有编码器块堆栈之后，以及*(3)* 刚才提到的连接处第二组位置嵌入参数之后，也就是解码器堆栈的词干模块中。

• 在解码器中的编码器–解码器多头混合注意力模块里，我们使用.....而不是通常用于双向/掩码多头自注意力模块的查询–键–值向量 (1.13)–(1.15)。

$$q^{h}_{\alpha;t;c} \equiv \sum_{i=1}^{n} Q^{h}_{ci} s_{\alpha;t;i} \,, \tag{2.2}$$

$$k^{h}_{\alpha;t;c} \equiv \sum_{i=1}^{n} K^{h}_{ci} \widetilde{s}_{\alpha;t;i} \,, \tag{2.3}$$

$$v^{h}_{\alpha;t;c} \equiv \sum_{i=1}^{n} V^{h}_{ci} \widetilde{s}_{\alpha;t;i} \,, \tag{2.4}$$

[46] 与原始 Language Transformer [13], 和原始 BART-large [44] 不同，后者将归一化层置于每个块的末尾；但 R2C2 [45] 并非如此。正如 §I0 中所述，按照如今的标准，我们在 BART-large 和 R2C2 中都将归一化层置于每条残差路径的起始处。

其中，与之前一样，$s_{\alpha;t;i}$ 是来自前一个（掩码多头自注意力）块的层归一化信号；但与之前不同的是，$\widetilde{s}_{\alpha;t;i}$ 是来自进入解码器堆栈之前的层归一化信号，也就是在添加第二组位置嵌入参数之前的信号。[47]然后，与双向多头自注意力类似，我们对所有词元应用 Softmax (1.17)，但不考虑被掩码的 [填充]词元。

为确保我们的比较基准一致，请检查这些架构选择是否使 BART-large 的模型参数数量为 $P = 50265 \cdot 1024 + 514 \cdot 1024 + 1024^2 \cdot (4 + 2 \cdot 4) \cdot 12 + 514 \cdot 1024 + 1024^2 \cdot (4 + 4 + 2 \cdot 4) \cdot 12 \approx 405 \cdot 10^6$，以及 R2C2 的模型参数数量为 $P = 50265 \cdot 2048 + 514 \cdot 2048 + 2048^2 \cdot (4 + 2 \cdot 4) \cdot 22 + 514 \cdot 2048 + 2048^2 \cdot (4 + 4 + 2 \cdot 4) \cdot 22 \approx 2.7 \cdot 10^9$。

C. 初始化超参数

关于初始化，在标准运行中，按照参考文献 [44, 45], ，我们对所有组中的全部模型参数统一采用标准差为 std $= 0.02$ 的零均值正态分布进行初始化。[48] 对于取值为 $n = 1024$ 和 $M = 4$ 的 BART-large，用我们的语言来说，这意味着 $C_Q = C_K = C_V = C_U = C_W = 0.4096$、$C_X = 1.6384$ 和 $C_{\mathrm{PE}} = C_{\mathrm{WE}} = 0.02$；对于取值为 $n = 2048$ 和 $M = 4$ 的 R2C2，则意味着 $C_Q = C_K = C_V = C_U = C_W = 0.8192$、$C_X = 3.2768$ 和 $C_{\mathrm{PE}} = C_{\mathrm{WE}} = (0.02)^2$。这些数值大多可以接受，属于数量级为一的数值——尽管从审美上看并不理想——但词嵌入参数的协方差 $C_{\mathrm{WE}} = (0.02)^2 = 0.0004$ 远非数量级为一，因此对于神经切线运行，我们将其调整为 $C_{\mathrm{WE}} = 1$，并相应地将标准运行中的 $\mathcal{N}_{\mathrm{rescale}} = 1$ 改为神经切线运行中的$\mathcal{N}_{\mathrm{rescale}} = \sqrt{1/n}$：请注意表 II 2 E 中的绿色部分。[49]

——

D. 优化器与训练超参数

总体而言，我们的训练方案基本遵循参考文献 [44, 45] 中采用的方案。[50]

[47] 理论上，缩放分析保持不变，内积 $(1/n)\sum_{i=1}^{n} s_{\alpha_1;t_1;i} s_{\alpha_2;t_2;i}$ 只需在某些地方替换为 $(1/n)\sum_{i=1}^{n} s_{\alpha_1;t_1;i} \widetilde{s}_{\alpha_2;t_2;i}$，在另一些地方替换为 $(1/n)\sum_{i=1}^{n} \widetilde{s}_{\alpha_1;t_1;i} \widetilde{s}_{\alpha_2;t_2;i}$。 [48] 具体而言，当 j 对应于 [pad] 词元时，我们将词嵌入参数 W_{ij}^{WE}初始化为零。[49] 相比之下，出于脚注 39 中所述的相同原因，我们认为 $C_{\mathrm{PE}} = (0.02)^2$ 可以接受：位置嵌入参数的作用类似于偏置参数（即，不是乘法性的，而是加法性的），因此原则上我们可以将该超参数设为任何数量级为 1 的数值，包括 $C_{\mathrm{PE}} = 0$。[50] 最显著的差异是，出于本研究在计算资源方面的考虑，我们的预热时长以及 *target* 训练时长分别为 5,000 次迭代和 200,000 次迭代，短于参考文献 [44]（参考文献 [45]）中采用的 10,000（15,000）次迭代和 500,000（500,000）次迭代。

至于优化器，我们使用 AdamW [28, 29] (1.91)–(1.94)，参数为（β_1，β_2，ϵ）=(0.9，0.998，10^{-6})；我们还将 --dropout 和 --attention-dropout 设为 0.1，并将 --clip-norm 设为 0.1。

关于学习率计划，我们采用有效批量大小为 512，并进行线性预热 [42]，前 5,000 次迭代后，在接下来的 195,000 次迭代中进行学习率线性衰减，也就是说，

$$\eta_t = \begin{cases} \texttt{lr} \cdot \left(\frac{t}{5000}\right) & \text{for} \quad t \leq 5000\,, \\ \texttt{lr} \cdot \left\{\frac{200000-t}{195000}\right\} & \text{for} \quad t > 5000\,. \end{cases} \tag{2.5}$$

出于计算资源方面的考虑，我们对 BART-large 模型进行完整的 200,000 次迭代训练，而 R2C2 模型则在 15,000 次迭代时停止；我们将在 §II 2 E 中稍后讨论整体学习率 lr 和权重衰减 wd 的设置。

~关于分组学习率因子 λ_G，在标准运行中，我们采用标准的~均匀缩放 $\lambda_G = 1$；而在神经切线运行中，我们遵循理论建议 (1.108)–(1.116)，但同样忽略因子 $M = 4$：请关注 蓝色 （表 II 2 E 中以蓝色标示）。总体而言，我们的神经切线学习率缩放本质上是将词嵌入参数和位置嵌入参数的学习率都提高宽度倍—— $n = 1024$ 对于 BART-large，以及 $n = 2048$ 对于 R2C2——相对于其他模型参数而言。

E. 缩放策略比较

这里，我们比较采用标准均匀缩放的超参数训练的模型与采用神经切线缩放的超参数训练的模型的性能：简要总结见下文表 II 2 E。

AdamW–标准运行			
	初始标准差	重缩放	相对学习率因子
词嵌入 W_{ij}^{WE}	0.02		1
位置嵌入 $b_{t;i}^{\mathrm{PE}}$	0.02		1
Q–K–V–U–W–X 权重	0.02		1
输出 $\mathcal{N}$		1.0	

AdamW–神经切线运行			
	初始标准差	重缩放	相对学习率因子
词嵌入 W_{ij}^{WE}	1		$n^{-\frac{1}{2}}$
位置嵌入 $b_{t;i}^{\mathrm{PE}}$	0.02		$n^{-\frac{1}{2}}$
Q–K–V–U–W–X 权重	0.02$_{(\mathrm{yuck})}$		$n^{-\frac{3}{2}}$
输出 $\mathcal{N}$		$n^{-\frac{1}{2}}$	

为了对 BART-large 进行合理范围内的半严格比较，我们调整整体学习率 lr；但鉴于这些模型的规模现在是第 II1 节中所研究视觉 Transformer 的五倍，我们不再调整权重衰减 wd。具体而言，对于标准运行，我们将权重衰减固定为 wd $= 0.01$，并在对数$_2$ 网格中搜索最优的整体学习率 lr

空间；对于神经切线运行，根据第 II 1E 节末尾的说明，我们将搜索空间转换为 (lr，wd) $\rightarrow$ ($n^{3/2}$lr $n^{-3/2}$wd)，此时将 wd 固定为 $= 0.01 \cdot 1024^{-3/2} \approx$ 3e-7：见图3。事实上，在这一映射下，我们看到，标准统一缩放策略的最优 lr（我们得到 lr$^\star =$ 1e-3）与神经切线缩放策略的最优 lr（我们得到 lr$^\star = 1024^{3/2} \cdot$ 1e-3 $= 32.768$）相匹配。随后，我们比较这两种最优运行结果：见图4。

对于 R2C2，由于模型比 BART-large 大七倍、比视觉 Transformer 大三十倍，我们甚至不会尝试调整任何超参数。相反，在标准运行中，我们采用参考文献 [45] 中使用的 (lr，wd) $=$ (7e-4，0。01)——不过这里的预热时间和目标训练时长更短——而在神经切线运行中，我们将其调整为 ($n^{3/2}$lr $n^{-3/2}$wd) $=$ ($2048^{3/2} \cdot$ 7e-4，$2048^{-3/2} \cdot 0.01$) $\approx$ (64。88，1。08e-7)。这一选择基于我们对视觉 Transformer 和 BART-large 模型的调参结果：见图 5

对于 BART-large，标准统一缩放策略似乎能得到略好的模型，但神经切线缩放策略对超参数变化似乎更为稳健——至少在降低全局学习率 lr 时是如此——若能使用不同的随机种子开展更多实验就更好了。令人鼓舞的是，对于 R2C2，神经切线缩放策略似乎能够改善收敛，至少在训练早期如此。若能将实验进一步扩展到语言 Transformer 开始出现训练中期尖峰的规模，并观察神经切线缩放策略是否能缓解这些尖峰、是否能带来进一步的性能提升，将会很有意思。

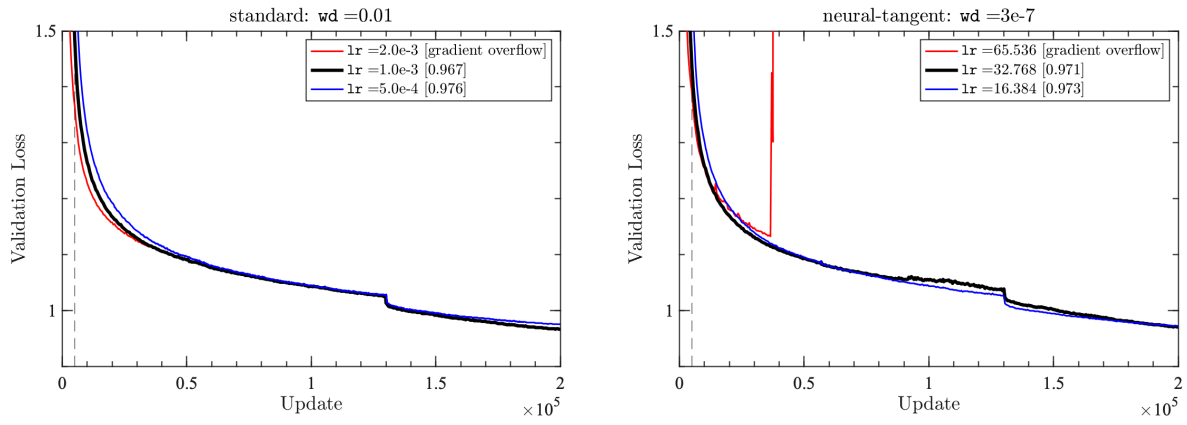


图3：使用 AdamW$_{(\beta_1,\beta_2,\epsilon)=(0.9,0.998,1e\text{-}6)}$训练的 BART-large 在标准均匀缩放（左）和神经切线缩放（右）策略下的超参数搜索。对于每种策略，图中绘出了验证损失随训练更新次数的变化。在图例中，我们记录了每条轨迹上验证损失最低时对应的全局学习率 lr [。请注意：*(i)* 两种缩放策略中，较高 lr 的运行（红色）均出现了梯度溢出；*(ii)* 降低 lr（从黑色变为蓝色）对标准均匀缩放策略的模型性能影响大于对神经切线缩放策略的影响；以及*(iii)* 对于神经切线缩放策略，$\text{lr} = 32.768$ 运行尽管在约 90,000 次迭代时出现了“波折”，最终仍追上了 $\text{lr} = 16.384$ 运行。]

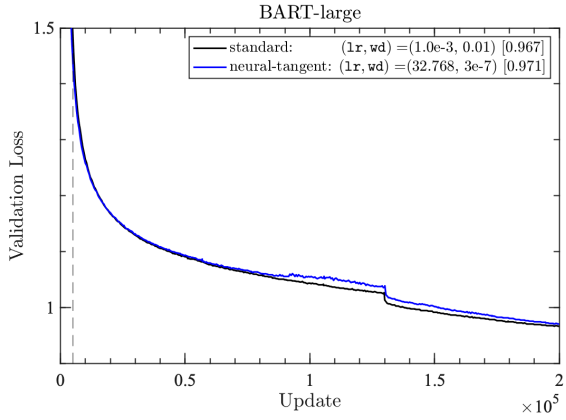


图 4：比较由 AdamW 训练的 BART-large 在标准均匀缩放策略（黑色）和神经切线缩放策略（蓝色）下的表现$(\beta_1,\beta_2,\epsilon)$=(0.9,0.998,1e-6)。对于每种策略，图中绘出了验证损失随训练更新次数的变化。图例中记录了每种缩放策略对应的训练超参数对（lr,wd） [，该超参数对在各自的训练轨迹上取得了最低验证损失]。

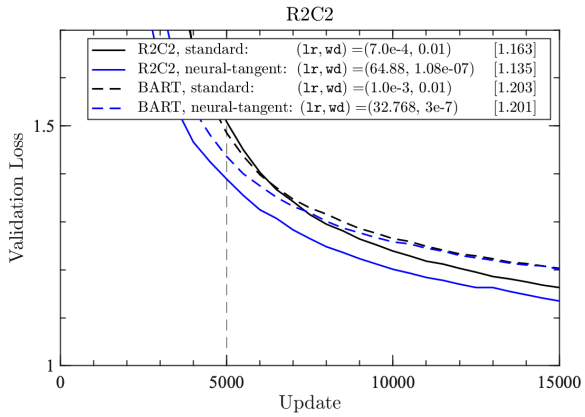


图 5：比较由 AdamW 训练的 R2C2 在标准均匀缩放策略（黑色）和神经切线缩放策略（蓝色）下的表现$(\beta_1,\beta_2,\epsilon)$=(0.9,0.998,1e-6)；为便于进一步比较，我们还纳入了 BART-large 的类似实验结果（虚线）。对于每种策略，图中绘出了验证损失随训练更新次数的变化。图例中记录了每种缩放策略对应的零样本训练超参数对（lr,wd） [，该超参数对在各自的训练轨迹上取得了最低验证损失]。

致谢

在个人方面，E.D. 感谢 Stephen Roller 和 Kurt Shuster 提供 R2C2 方面的建议；S.Y. 感谢 Jaehoon Lee 在缩放策略的理论和实践方面提供各种建议，感谢 Dan Roberts 提供写作建议，并感谢 Ross Girshick、Mike Lewis、Eric Mintun 和 Dan Roberts 与其合作开展关于 $C_Q = C_K = C_V = C_W = 1$ 和 $C_U = C_X = 0$ 影响的未发表研究；S.Z. 感谢 FAIR 提供充足的计算资源，并给予跨实验室合作的自由。

从技术层面来说，为了凸显埋藏在脚注29中的未来研究方向，我们一致承认，受限于时间和精力，我们未能妥善考虑批量大小 $|\mathcal{A}_t|$、序列长度 T、词汇表大小 n_{vocab} 与词元分布之间的相互作用，但仍希望我们的工作能为未来研究提供一个有益的轶事性数据点。

A. 自注意力统计量的注意力

欲知他人，必先知己。

Bumblebee

在本 附录中，我们将确定查询–键点积 (1.16) 的统计量，，

$$\widetilde{\Omega}^{h}_{\alpha;tt'} \equiv \frac{1}{\sqrt{C}} \sum_{c=1}^{C} q^{h}_{\alpha;t;c} k^{h}_{\alpha;t';c} = \frac{1}{\sqrt{C}} \sum_{c=1}^{C} \sum_{i_1,i_2=1}^{n} Q^{h}_{ci_1} K^{h}_{ci_2} s_{\alpha;t;i_1} s_{\alpha;t';i_2} \,, \tag{A1}$$

而这又决定了自注意力矩阵的统计量。在正文中，我们已经看到，该乘积 (1.60) 的均值会趋于零，

$$\begin{aligned} \mathbb{E}\left[\widetilde{\Omega}^{h}_{\alpha;tt'}\right] =& \mathbb{E}\left[\frac{1}{\sqrt{C}} \sum_{c=1}^{C} \sum_{i,j=1}^{n} Q^{h}_{ci} K^{h}_{cj} s_{\alpha;t;i} s_{\alpha;t';j}\right] \\ =& \frac{1}{\sqrt{C}} \sum_{c=1}^{C} \sum_{i,j=1}^{n} \mathbb{E}\left[Q^{h}_{ci}\right] \mathbb{E}\left[K^{h}_{cj}\right] \mathbb{E}\left[s_{\alpha;t;i} s_{\alpha;t';j}\right] = 0 \,, \end{aligned} \tag{A2}$$

而其协方差 (1.61) 则为

$$\begin{aligned} & \mathbb{E}\left[\widetilde{\Omega}^{h_1}_{\alpha_1;t_1t_1'} \widetilde{\Omega}^{h_2}_{\alpha_2;t_2t_2'}\right] \\ =& \frac{1}{C} \sum_{c_1,c_2=1}^{C} \sum_{i_1,i_2,j_1,j_2=1}^{n} \mathbb{E}\left[Q^{h_1}_{c_1i_1} K^{h_1}_{c_1j_1} s_{\alpha_1;t_1;i_1} s_{\alpha_1;t_1';j_1} Q^{h_2}_{c_2i_2} K^{h_2}_{c_2j_2} s_{\alpha_2;t_2;i_2} s_{\alpha_2;t_2';j_2}\right] \\ =& \frac{1}{C} \sum_{c_1,c_2=1}^{C} \sum_{i_1,i_2,j_1,j_2=1}^{n} \frac{C_Q}{n} \frac{C_K}{n} \delta_{c_1c_2} \delta_{i_1i_2} \delta_{j_1j_2} \delta^{h_1h_2} \mathbb{E}\left[s_{\alpha_1;t_1;i_1} s_{\alpha_1;t_1';j_1} s_{\alpha_2;t_2;i_2} s_{\alpha_2;t_2';j_2}\right] \\ =& \delta^{h_1h_2} C_Q C_K \mathbb{E}\left[\left(\frac{1}{n} \sum_{i=1}^{n} s_{\alpha_1;t_1;i} s_{\alpha_2;t_2;i}\right) \left(\frac{1}{n} \sum_{j=1}^{n} s_{\alpha_1;t_1';j} s_{\alpha_2;t_2';j}\right)\right] \\ =& \delta^{h_1h_2} C_Q C_K F_{(\alpha_1;t_1)(\alpha_2;t_2)} F_{(\alpha_1;t_1')(\alpha_2;t_2')} + O\left(\frac{1}{n}\right) \\ \equiv& \delta^{h_1h_2} A_{(\alpha_1;t_1t_1')(\alpha_2;t_2t_2')} \,, \end{aligned} \tag{A3}$$

$$\tag{A4}$$

其中在最后一行，我们引入了用于查询与键点积的核 $A_{(\alpha_1;t_1t_1')(\alpha_2;t_2t_2')}$ 。假设查询和键权重的分布关于零对称——通常情况确实如此——我们可以清楚地看到，所有奇数点关联函数都为零，因为

$$\mathbb{E}\left[\widetilde{\Omega}^{h_1}_{\alpha_1;t_1t_1'} \cdots \widetilde{\Omega}^{h_{2m-1}}_{\alpha_{2m-1};t_{2m-1}t_{2m-1}'}\right] = 0 \,. \tag{A5}$$

因此我们将重点关注 对偶数点关联函数。接下来，像之前那样将它们展开

对于协方差（后续将提供逐步说明），我们得到

$$
\begin{aligned}
&\mathbb{E}\left[\widetilde{\Omega}^{h_1}_{\alpha_1;t_1t_1'}\cdots\widetilde{\Omega}^{h_{2m}}_{\alpha_{2m};t_{2m}t_{2m}'}\right] \qquad \text{(A6)}\\
=&\frac{1}{C^m}\sum_{c_1,\ldots,c_{2m}=1}^{C}\sum_{i_1,\ldots,i_{2m}=1}^{n}\sum_{j_1,\ldots,j_{2m}=1}^{n}\mathbb{E}\left[Q^{h_1}_{c_1i_1}\cdots Q^{h_{2m}}_{c_{2m}i_{2m}}\right]\mathbb{E}\left[K^{h_1}_{c_1j_1}\cdots K^{h_{2m}}_{c_{2m}j_{2m}}\right]\\
&\quad\times\mathbb{E}\left[\left(s_{\alpha_1;t_1;i_1}\cdots s_{\alpha_{2m};t_{2m};i_{2m}}\right)\left(s_{\alpha_1;t_1';j_1}\cdots s_{\alpha_{2m};t_{2m}';j_{2m}}\right)\right]\\
=&\frac{1}{C^m}\sum_{c_1,\ldots,c_{2m}=1}^{C}\sum_{i_1,\ldots,i_{2m}=1}^{n}\sum_{j_1,\ldots,j_{2m}=1}^{n}\left(\frac{C_Q}{n}\frac{C_K}{n}\right)^m\\
&\quad\times\left[\sum_{\text{all pairing}}\left(\delta_{c_{p_1}c_{p_2}}\cdots\delta_{c_{p_{2m-1}}c_{p_{2m}}}\right)\left(\delta_{i_{p_1}i_{p_2}}\cdots\delta_{i_{p_{2m-1}}i_{p_{2m}}}\right)\left(\delta^{h_{p_1}h_{p_2}}\ldots\delta^{h_{p_{2m-1}}h_{p_{2m}}}\right)\right]\\
&\quad\times\left[\sum_{\text{all pairing}}\left(\delta_{c_{q_1}c_{q_2}}\cdots\delta_{c_{q_{2m-1}}c_{q_{2m}}}\right)\left(\delta_{j_{q_1}j_{q_2}}\cdots\delta_{j_{q_{2m-1}}j_{q_{2m}}}\right)\left(\delta^{h_{q_1}h_{q_2}}\ldots\delta^{h_{q_{2m-1}}h_{q_{2m}}}\right)\right]\\
&\quad\times\mathbb{E}\left[\left(s_{\alpha_1;t_1;i_1}\cdots s_{\alpha_{2m};t_{2m};i_{2m}}\right)\left(s_{\alpha_1;t_1';j_1}\cdots s_{\alpha_{2m};t_{2m}';j_{2m}}\right)\right]+O\left(\frac{1}{n}\right)\\
=&\frac{1}{C^m}C^m\sum_{i_1,\ldots,i_{2m}=1}^{n}\sum_{j_1,\ldots,j_{2m}=1}^{n}\left(\frac{C_Q}{n}\frac{C_K}{n}\right)^m\\
&\quad\times\left[\sum_{\text{all pairing}}\left(\delta_{i_{p_1}i_{p_2}}\cdots\delta_{i_{p_{2m-1}}i_{p_{2m}}}\right)\left(\delta^{h_{p_1}h_{p_2}}\ldots\delta^{h_{p_{2m-1}}h_{p_{2m}}}\right)\left(\delta_{j_{p_1}j_{p_2}}\cdots\delta_{j_{p_{2m-1}}j_{p_{2m}}}\right)\right]\\
&\quad\times\mathbb{E}\left[\left(s_{\alpha_1;t_1;i_1}\cdots s_{\alpha_{2m};t_{2m};i_{2m}}\right)\left(s_{\alpha_1;t_1';j_1}\cdots s_{\alpha_{2m};t_{2m}';j_{2m}}\right)\right]+O\left(\frac{1}{C}\right)\\
=&\left(C_QC_K\right)^m\sum_{\text{all pairing}}\left(\delta^{h_{p_1}h_{p_2}}\ldots\delta^{h_{p_{2m-1}}h_{p_{2m}}}\right)\\
&\quad\times\mathbb{E}\Bigg[\left(\frac{1}{n}\sum_{i_1=1}^{n}s_{\alpha_{p_1};t_{p_1};i_1}s_{\alpha_{p_2};t_{p_2};i_1}\right)\cdots\left(\frac{1}{n}\sum_{i_m=1}^{n}s_{\alpha_{p_{2m-1}};t_{p_{2m-1}};i_m}s_{\alpha_{p_{2m}};t_{p_{2m}};i_m}\right)\\
&\qquad\times\left(\frac{1}{n}\sum_{j_1=1}^{n}s_{\alpha_{p_1};t_{p_1}';j_1}s_{\alpha_{p_2};t_{p_2}';j_1}\right)\cdots\left(\frac{1}{n}\sum_{j_m=1}^{n}s_{\alpha_{p_{2m-1}};t_{p_{2m-1}}';j_m}s_{\alpha_{p_{2m}};t_{p_{2m}}';j_m}\right)\Bigg]+O\left(\frac{1}{C}\right)\\
=&\sum_{\text{all pairing}}\left(\delta^{h_{p_1}h_{p_2}}\ldots\delta^{h_{p_{2m-1}}h_{p_{2m}}}\right)\\
&\quad\times A_{(\alpha_{p_1};t_{p_1}t_{p_1}')(\alpha_{p_2};t_{p_2}t_{p_2}')}\cdots A_{(\alpha_{p_{2m-1}};t_{p_{2m-1}}t_{p_{2m-1}}')(\alpha_{p_{2m}};t_{p_{2m}}t_{p_{2m}}')}+O\left(\frac{1}{C}\right).
\end{aligned}
$$

这里，在第一个等号中，我们明确写出了查询–键点积，并利用查询权重和键权重的统计独立性，将两者的期望分开；在第二个等号中，我们使用了维克定理，将查询权重的期望表示为辅助指标 1、.....、 $2m$ 的所有 $(2m-1)!!$ 种配对之和，并对键权重做了同样的处理（如果查询和键权重服从正态分布，这一步是精确的，而

它涉及非正态分布的 $1/n$ 修正）；在第三个等式中，我们对每头通道索引求和，并注意到，当查询权重与键权重对应的索引配对不同时，其贡献会受到 $(1/C = H/n)$ 抑制（如果这段解释不够清楚，我们建议针对 $2m = 4$ 的情形明确执行这一步）；在第四步中，我们只需对嵌入索引求和；在第五步中，我们利用八倍结果 (1.52) 截去 $1/n$ 修正，并使用新引入的核的定义 $A_{(\alpha_1;\ t_1t_1')(\alpha_2;\ t_2t_2')}$。

总而言之，在 $1/C = H/n$ 的主阶下，查询–键点积 $\widetilde{\Omega}^h_{\alpha;tt'}$ 服从零均值、高斯统计，其协方差为一阶，且为 $\delta^{h_1h_2}A_{(\alpha_1;\ t_1t_1')(\alpha_2;\ t_2t_2')}$。因此，查询–键点积 $\widetilde{\Omega}^h_{\alpha;tt'}$ 的任何一阶函数，尤其是自注意力矩阵 $\Omega^h_{\alpha;tt'}$，其期望值均为一阶（除非该函数只涉及查询–键点积的奇次幂）。我们还强调，所有不同的注意力头在统计上相互独立——这由克罗内克δ $\delta^{h_1h_2}$ 体现；例如，脚注 14 中出现的期望值 $\mathbb{E}\left[\Omega^h_{\alpha_1;\ t_1t_1'}\Omega^h_{\alpha_2;\ t_2t_2'}\right]$ 对所有注意力头 h 都取相同的一阶值，并可表示为一个 $\left(|\mathcal{D}|T^2\right)$ 维高斯积分，其核为 $A_{(\alpha_1;\ t_1t_1')(\alpha_2;\ t_2t_2')}$。

关于脚注 14，为了展示其中的因式分解，只需注意，上述高斯统计的五重推导 (A6) 原样成立，从而得到

$$
\begin{aligned}
&\mathbb{E}\left[\widetilde{\Omega}^{h_1}_{\alpha_1;t_1t_1'}\cdots\widetilde{\Omega}^{h_{2m}}_{\alpha_{2m};t_{2m}t_{2m}'}\mathcal{F}(s)\right] \qquad \text{(A7)}\\
&=(C_QC_K)^m\sum_{\text{all pairing}}\left(\delta^{h_{p_1}h_{p_2}}\cdots\delta^{h_{p_{2m-1}}h_{p_{2m}}}\right)\\
&\quad\times\mathbb{E}\Bigg[\left(\frac{1}{n}\sum_{i_1=1}^{n}s_{\alpha_{p_1};t_{p_1};i_1}s_{\alpha_{p_2};t_{p_2};i_1}\right)\cdots\left(\frac{1}{n}\sum_{i_m=1}^{n}s_{\alpha_{p_{2m-1}};t_{p_{2m-1}};i_m}s_{\alpha_{p_{2m}};t_{p_{2m}};i_m}\right)\\
&\qquad\times\left(\frac{1}{n}\sum_{j_1=1}^{n}s_{\alpha_{p_1};t'_{p_1};j_1}s_{\alpha_{p_2};t'_{p_2};j_1}\right)\cdots\left(\frac{1}{n}\sum_{j_m=1}^{n}s_{\alpha_{p_{2m-1}};t'_{p_{2m-1}};j_m}s_{\alpha_{p_{2m}};t'_{p_{2m}};j_m}\right)\mathcal{F}(s)\Bigg]+O\left(\frac{1}{C}\right)
\end{aligned}
$$

其中插入任意关于信号 $s_{\alpha;t;i}$ 的函数 $\mathcal{F}(s)$，然后我们便可再次利用八倍结果 (1.52)，完成证明 (1.64)。

如果你一直在 保持注意力，然后还有一项交易（1.155）需要封闭，也就是脚注 30 .

这种封闭本质上归结为认识到

$$\frac{1}{C^m}\sum_{c_1,\ldots,c_{2m}=1}^{C}\mathbb{E}\left[Q_{c_1i_1}^{h_1}\cdots Q_{c_{2m}i_{2m}}^{h_{2m}}\right]\mathbb{E}\left[K_{c_1j_1}^{h_1}\cdots K_{c_{2m}j_{2m}}^{h_{2m}}\left(K_{c_{2m+1}j_{2m+1}}^{h_{2m+1}}K_{c_{2m+2}j_{2m+2}}^{h_{2m+2}}\right)\right] \tag{A8}$$

$$=\frac{1}{C^m}\sum_{c_1,\ldots,c_{2m}=1}^{C}\left(\frac{C_Q}{n}\frac{C_K}{n}\right)^m$$

$$\times\left[\sum_{\text{all pairing}}\left(\delta_{c_{p_1}c_{p_2}}\cdots\delta_{c_{p_{2m-1}}c_{p_{2m}}}\right)\left(\delta_{i_{p_1}i_{p_2}}\cdots\delta_{i_{p_{2m-1}}i_{p_{2m}}}\right)\left(\delta^{h_{p_1}h_{p_2}}\ldots\delta^{h_{p_{2m-1}}h_{p_{2m}}}\right)\right]$$

$$\times\left[\sum_{\text{all pairing}}\left(\delta_{c_{q_1}c_{q_2}}\cdots\delta_{c_{q_{2m+1}}c_{q_{2m+2}}}\right)\left(\delta_{j_{q_1}j_{q_2}}\cdots\delta_{j_{q_{2m+1}}j_{q_{2m+2}}}\right)\left(\delta^{h_{q_1}h_{q_2}}\ldots\delta^{h_{q_{2m+1}}h_{q_{2m+2}}}\right)\right]$$

$$=\left(\frac{C_Q}{n}\frac{C_K}{n}\right)^m\delta_{c_{2m+1}c_{2m+2}}\delta_{j_{2m+1}j_{2m+2}}\delta^{h_{2m+1}h_{2m+2}}$$

$$\times\left[\sum_{\text{all pairing}}\left(\delta_{i_{p_1}i_{p_2}}\cdots\delta_{i_{p_{2m-1}}i_{p_{2m}}}\right)\left(\delta^{h_{p_1}h_{p_2}}\ldots\delta^{h_{p_{2m-1}}h_{p_{2m}}}\right)\left(\delta_{j_{p_1}j_{p_2}}\cdots\delta_{j_{p_{2m-1}}j_{p_{2m}}}\right)+O\left(\frac{1}{C}\right)\right].$$

为了使推导更简洁 并具有更好的泛化能力，请参阅文献 [15]，寻找层间相关性。

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, 等人，《语言模型是少样本学习者》，载于神经信息处理系统进展，第 33 卷，第 1877–1901 页。2020年。[2] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young, 等人，《扩展语言模型：训练 Gopher 的方法、分析与见解》，arXiv:2112.11446 [计算与语言].[3] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti, 等人，《使用 DeepSpeed 和 Megatron 训练 Megatron-Turing NLG 530B：大规模生成式语言模型》，arXiv:2201.11990 [计算与语言].[4] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, 等人，《PaLM：基于 Pathways 扩展语言建模》，arXiv:2204.02311 [计算与语言].[5] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin,等人，《OPT：开放式预训练 Transformer 语言模型》，arXiv:2205.01068 [计算与语言].[6] R. M. Neal，《无限网络的先验》，载于神经网络的贝叶斯学习，第 29–53 页。Springer，1996年。[7] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein，《深度神经网络作为高斯过程》，载于国际学习表征会议。2018年。

[8] A. G. d. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani, 《宽深度神经网络中的高斯过程行为》，载于国际学习表征会议。2018年。[9] A. Jacot, F. Gabriel, and C. Hongler, 《神经切线核：神经网络中的收敛与泛化》，载于神经信息处理系统进展，第 31 卷，第 8571–8580 页。2018年。[10] B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli, 《通过瞬态混沌提升深度神经网络的指数表达能力》，载于神经信息处理系统进展，第 29 卷，第 3360–3368 页。2016年。[11] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, 《论深度神经网络的表达能力》，载于国际机器学习会议，第 2847–2854 页。2017年。[12] S. S. Schoenholz, J. Gilmer, S. Ganguli, and J. Sohl-Dickstein, 《深度信息传播》，载于国际学习表征会议。2017年。[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, 《注意力就是一切所需》，载于神经信息处理系统进展，第 30 卷，第 5998–6008 页。2017年。[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, 等人, 《一张图像胜过 16×16 个词：用于大规模图像识别的 Transformer》，载于国际学习表征会议。2021年。[15] D. A. Roberts, S. Yaida, and B. Hanin, 深度学习理论原理。剑桥大学出版社，2022年。arXiv:2106.10165 [cs.LG]。https://deeplearningtheory.com。[16] J. L. Ba, J. R. Kiros, and G. E. Hinton, 《层归一化》，arXiv:1607.06450 [stat.ML]。[17] D. Doshi, T. He, and A. Gromov, 《通过部分雅可比矩阵对宽而深的神经网络进行临界初始化：一般理论与应用》，arXiv:2111.12143 [cs.LG]。[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, 《ImageNet：大规模分层图像数据库》，载于 *2009*年 *IEEE*计算机视觉与模式识别会议，第 248–255 页。2009年。[19] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, 《视觉Transformer扩展》，载于*IEEE/CVF*计算机视觉与模式识别会议，第 12104–12113 页。2022年。[20] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Doll´ar, and R. Girshick, 《早期卷积有助于 Transformer 更好地进行视觉感知》，载于神经信息处理系统进展，第 34 卷，第 30392–30400 页。2021年。[21] N. Shazeer, 《GLU 变体改进 Transformer》，arXiv:2002.05202 [计算与语言]。[22] O. Press and L. Wolf, 《利用输出嵌入改进语言模型》，载于计算语言学协会欧洲分会第 *15* 届会议论文集：第 *2* 卷，短篇论文，第 157–163 页。2017年。[23] S. Yaida, 《元原则族的超参数缩放策略》，arXiv:2210.04909

[cs.LG].[24] G. Yang 和 E. J. Hu，《张量程序 IV：无限宽神经网络中的特征学习》，载于《国际机器学习会议》，第 11727–11737 页。2021年。[25] G. Yang、E. J. Hu、I. Babuschkin、S. Sidor、X. Liu、D. Farhi、N. Ryder、J. Pachocki、W. Chen 和 J. Gao，《张量程序 V：通过零样本超参数迁移调优大型神经网络》，arXiv:2203.03466 [cs.LG].[26] E. Dyer 和 G. Gur-Ari，《费曼图视角下宽网络的渐近分析》，载于《国际学习表征会议》。2020年。[27] B. Hanin 和 M. Nica，《神经切线核的有限深度和宽度修正》，载于《国际学习表征会议》。2019年。[28] D. P. Kingma 和 J. Ba，《Adam：一种随机优化方法》，arXiv:1412.6980[cs.LG].[29] I. Loshchilov 和 F. Hutter，《解耦权重衰减正则化》，arXiv:1711.05101 [cs.LG].[30] G. K. Zipf，《人类行为与最省力原则：人类生态学导论》。Ravenio Books，2016年。[31] P. Micikevicius、S. Narang、J. Alben、G. Diamos、E. Elsen、D. Garcia、B. Ginsburg、M. Houston、O. Kuchaiev、G. Venkatesh 等，《混合精度训练》，arXiv:1710.03740 [cs.AI].[32] Z. Liu、H. Mao、C.-Y. Wu、C. Feichtenhofer、T. Darrell 和 S. Xie，《面向2020年代的卷积网络》，载于《IEEE/CVF计算机视觉与模式识别会议》，第 11976–11986 页。2022年。[33] O. Russakovsky、J. Deng、H. Su、J. Krause、S. Satheesh、S. Ma、Z. Huang、A. Karpathy、A. Khosla、M. Bernstein 等，《ImageNet大规模视觉识别挑战赛》，《国际计算机视觉杂志》115卷第3期，（2015年）211–252。[34] E. D. Cubuk、B. Zoph、J. Shlens 和 Q. V. Le，《Randaugment：以缩减搜索空间实现实用的自动数据增强》，载于《IEEE/CVF计算机视觉与模式识别会议》，第 702–703 页。2020年。[35] H. Zhang、M. Cisse、Y. N. Dauphin 和 D. Lopez-Paz，《Mixup：超越经验风险最小化》，载于《国际学习表征会议》。2018年。[36] S. Yun、D. Han、S. J. Oh、S. Chun、J. Choe 和 Y. Yoo，《CutMix：用于训练强分类器的可定位特征正则化策略》，载于《IEEE/CVF国际计算机视觉会议》，第 6023–6032 页。2019年。[37] Z. Zhong、L. Zheng、G. Kang、S. Li 和 Y. Yang，《随机擦除数据增强》，载于《AAAI人工智能会议论文集》，第 34卷，第 13001–13008 页。2020年。[38] R. Wightman 等，《PyTorch图像模型》，2019年。[39] C. Szegedy、V. Vanhoucke、S. Ioffe、J. Shlens 和 Z. Wojna，《重新思考用于计算机视觉的Inception架构》，载于《IEEE计算机视觉与模式识别会议》，第 2818–2826 页。2016年。[40] G. Huang、Y. Sun、Z. Liu、D. Sedra 和 K. Q. Weinberger，《具有随机深度的深度网络》，

载于计算机视觉—*ECCV 2016*：第*14*届欧洲会议，阿姆斯特丹，荷兰，*2016*年*10*月*11*日至*14*日，会议论文集，第四部分 *14*，第 646–661 页。2016年。[41] B. T. Polyak 和 A. B. Juditsky，《通过平均加速随机逼近》，*SIAM* 控制与优化学报**30** 卷第4期，（1992年）838–855。[42] P. Goyal、P. Doll´ar、R. Girshick、P. Noordhuis、L. Wesolowski、A. Kyrola、A. Tulloch、Y. Jia 和 K. He，《精确的大批量随机梯度下降：1小时内训练ImageNet》，arXiv:1706.02677[cs.CV].[43] I. Loshchilov 和 F. Hutter，《SGDR：带热重启的随机梯度下降》，载于国际学习表征会议。2017年。[44] M. Lewis、Y. Liu、N. Goyal、M. Ghazvininejad、A. Mohamed、O. Levy、V. Stoyanov 和 L. Zettlemoyer，《BART：面向自然语言生成、翻译和理解的去噪序列到序列预训练》，载于第*58*届计算语言学协会年会论文集，第 7871–7880 页。2020年。[45] K. Shuster、M. Komeili、L. Adolphs、S. Roller、A. Szlam 和 J. Weston，《寻求知识的语言模型：用于对话和提示补全的模块化搜索与生成》，载于计算语言学协会研究成果：*EMNLP 2022*，阿布扎比，阿拉伯联合酋长国，*2022*年*12*月*7*日至*11*日，第 373–393 页。2022年。[46] M. Ott、S. Edunov、A. Baevski、A. Fan、S. Gross、N. Ng、D. Grangier 和 M. Auli，《fairseq：快速、可扩展的序列建模工具包》，载于计算语言学协会北美分会*2019*年会议（演示）论文集，第 48–53 页。2019年。[47] Y. Liu、M. Ott、N. Goyal、J. Du、M. Joshi、D. Chen、O. Levy、M. Lewis、L. Zettlemoyer 和 V. Stoyanov，《RoBERTa：一种经过稳健优化的BERT预训练方法》，arXiv:1907.11692[cs.CL].[48] Y. Zhu、R. Kiros、R. Zemel、R. Salakhutdinov、R. Urtasun、A. Torralba 和 S. Fidler，《对齐书籍与电影：通过观看电影和阅读书籍实现故事化视觉解释》，载于*IEEE* 国际计算机视觉会议，第 19–27 页。2015年。[49] J. Devlin、M.-W. Chang、K. Lee 和 K. Toutanova，《BERT：用于语言理解的深度双向Transformer预训练》，arXiv:1810.04805 [cs.CL].[50] S. Nagel，《CC-News》，2016年。http://commoncrawl.org/2016/10/news-dataset-available.[51] A. Gokaslan 和 V. Cohen，《Openwebtext语料库》，2019年。http://Skylion007.github.io/OpenWebTextCorpus.[52] A. Radford、J. Wu、R. Child、D. Luan、D. Amodei、I. Sutskever、等，《语言模型是无监督多任务学习器》，*OpenAI*博客**1** 卷第8期，（2019年）9。[53] T. H. Trinh 和 Q. V. Le，《常识推理的一种简单方法》，arXiv:1806.02847[cs.AI].[54] A. Conneau、K. Khandelwal、N. Goyal、V. Chaudhary、G. Wenzek、F. Guzm´an、E. Grave、M. Ott,

L. Zettlemoyer 和 V. Stoyanov，“大规模无监督跨语言表征学习”，载于第*58*届计算语言学协会年会论文集，第8440–8451页。2020年。[55] J. Baumgartner、S. Zannettou、B. Keegan、M. Squire 和 J. Blackburn，“Pushshift Reddit 数据集”，载于 *AAAI*网络与社交媒体国际会议论文集，第14卷，第830–839页。2020年。[56] S. Roller、E. Dinan、N. Goyal、D. Ju、M. Williamson、Y. Liu、J. Xu、M. Ott、K. Shuster、E. M. Smith等，“构建开放域聊天机器人的方法”，载于计算语言学协会欧洲分会第*16*届会议论文集：主会场，第300–325页。2021年。