递归循环 Transformer

Recurrent Looped Transformer

Yifan Zhang[1] **Jichen Feng**[2] **Shihan Qin**[2]
Sun-Yuan Kung[1] **Mengdi Wang**[1]
[1] 普林斯顿大学 [2] 宾夕法尼亚大学

2026 年 9 月 12 日*

摘要

状态跟踪需要在每个输入上更新状态，而 Transformer 对每个 token 施加的深度固定，与序列长度无关。我们提出递归循环 Transformer（RLT），将模型的层分配给并行的因果编码器和递归解码器。在每个 token 上，解码器将编码器输出与上一个 token 的最终解码器状态融合，使计算路径随序列长度增长，而每个 token 的计算量不变。这一设计遵循三条原则：隐式推理（latent reasoning）、训练与推理零失配、硬件高效。我们在六个算法任务上，用三个种子比较八层模型的五种配比和八层 Transformer。在至多 40 位的数据上训练后，两种 RLT 配比在全部种子上都以 100% 的准确率泛化到 256 位奇偶校验，Transformer 则停留在随机水平。在训练长度八倍的交换 S_5 置换跟踪上，RLT 的最终状态准确率达到 97%，Transformer 低于 1%，且准确率随解码器层数增加而上升。在超出训练长度的模运算上，RLT 最高达到 93%，Transformer 为 33%。消融实验表明这些提升依赖反馈：移除反馈后，每种配比的奇偶校验和交换 S_5 都降至随机水平。每四个 token 的 chunk 更新一次反馈，可使 chunk 内的已知 token 并行计算，64 位奇偶校验仍保持在 99%；置换跟踪则依赖逐 token 反馈：分块后，长度 64 的交换 S_5 从 100% 降至 20%。

项目主页：https://github.com/yifanzhang-pro/recurrent-looped-tranformer

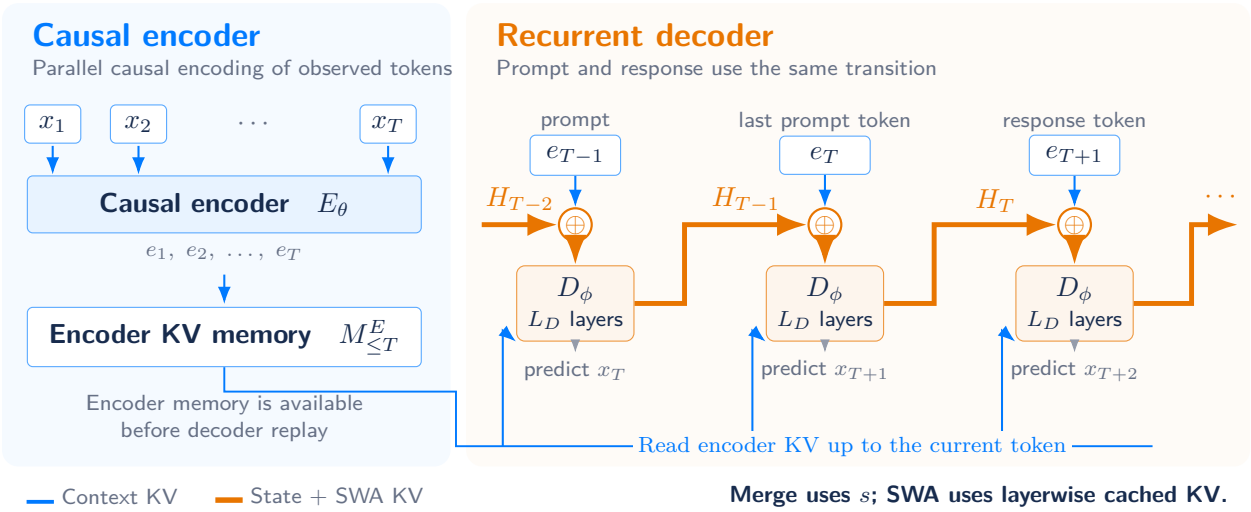


图 1 跨越提示与回复的递归计算。编码器构建因果键值（KV）记忆，解码器按 token 顺序处理。橙色箭头传递解码器状态 $H_t = (s_t, C_t^D)$：融合操作读取 s_t，每个 SWA 层读取各自的缓存。蓝色箭头提供截至当前位置的编码器记忆。图中展示提示的最后两次更新和回复的第一次更新。

*修订于 2026 年 10 月 5 日；中文译本对应同日英文公开版。

1 引言

算法状态跟踪要求模型随每个输入更新状态：奇偶校验需要逐位累积，置换跟踪需要依次复合群元素。Transformer 对每个位置只使用固定层数；除非 $\mathsf{TC}^0 = \mathsf{NC}^1$，固定深度的 Transformer 无法在任意长的序列上跟踪 S_5 置换的复合 (Merrill et al., 2024)。递归网络则把每次更新的结果传给下一次更新，计算路径随输入增长。在 Transformer 中加入这一连接，后续 token 就能沿随序列长度增长的路径继续利用先前的计算。

本文提出递归循环 Transformer（RLT），将模型的层分配给因果编码器和递归 Transformer 解码器（图 1）。编码器并行处理已知 token，构建 token 表示和全局键值记忆。在每个位置，门控融合将当前编码器表示与上一个 token 的最终解码器输出结合。随后，解码器通过注意力读取编码器前缀和自身近期激活，预测下一个 token。提示与回复使用同一状态转移。解码器深度为 L_D 时，处理 t 个 token 会形成穿过 tL_D 个解码器模块的递归路径，而每个 token 只经过 $L_E + L_D$ 层。

这一架构遵循三条设计原则。其一是隐式推理（latent reasoning）：反馈经由隐藏状态传递，后续 token 可以利用先前的计算，而这些计算不必写成 token；随着序列延长，递归路径没有固定的深度上限。其二是训练与推理零失配：预训练、监督微调、采样和策略重放使用同一状态转移。在强化学习中，用当前参数重放已采样的历史即可重建递归状态，因此训练器计算动作概率时使用的就是推理时的状态转移（附录 D）。其三是硬件效率：编码器并行处理已知 token，相互独立的序列可以把各自的解码器更新合并为一次批处理（图 10），反馈间隔大于一个 token 时，同一分块内的已知 token 可以一起计算。

这一设计带来两项仅含解码器的 Transformer 所没有的选择。第一是编码器与解码器的深度配比：在总层数不变时，将层移至解码器会延长递归路径，但也会增加训练与预填充时须按 token 顺序执行的解码器计算。第二是反馈间隔 B，即两次反馈更新之间的 token 数。RLT-1 每个 token 反馈一次（$B = 1$），RLT-2 在每个 B token 的分块（chunk）内共享一个反馈状态，RLT-0 移除反馈（$B = \infty$）。对 T 个已知 token，解码器在 RLT-1 中需要 TL_D 个串行模块阶段，在 RLT-2 中需要 $\lceil T/B \rceil L_D$ 个，在 RLT-0 中只需 L_D 个（表 1）；RLT-2 与 RLT-1 参数相同。

我们在六个算法任务上比较八层模型的五种配比和八层仅含解码器的 Transformer：加法、奇偶校验、含括号与不含括号的模五运算，以及标准与交换 S_5 状态跟踪。这些八层对照均使用三个初始化种子和共享的训练与测试数据，并在远超训练范围的长度上测试。我们另在奇偶校验与交换 S_5 上，将对称 RLT 配比从 $4+4$ 扩展至 $10+10$，各自与总深度相同的 Transformer 比较（第 3.4 节）。本文的贡献如下。

- **架构**。RLT 将编码器记忆复用 (Sun et al., 2024) 与贯穿整个解码器的逐 token 反馈结合。附录给出沿完整递归历史反向传播的训练方法、多轮缓存复用和精确的策略重放。
- **长度泛化**。在至多 40 位的数据上训练后，RLT 的 $5+3$ 与 $7+1$ 配比在每个种子上都达到 256 位奇偶校验 $100 \pm 0\%$ 的准确率，Transformer 为 $50.07 \pm 1.63\%$。在 256 次操作（训练长度的八倍）的交换 S_5 上，$4+4$ 的最终状态准确率为 $97.30 \pm 2.76\%$，Transformer 为 $0.85 \pm 0.30\%$。训练 5,000 步后，$6+2$ 在长度为 63 的无括号模五表达式上达到 $93.36 \pm 5.69\%$，Transformer 为 $33.20 \pm 2.33\%$（第 3.2 节）。
- **深度配比**。最佳配比取决于任务：奇偶校验在一层或三层解码器时泛化最好，而交换 S_5 在 256 次操作上的准确率随解码器层数增加而上升，从一层时的随机水平升至四层时的 97%。
- **反馈间隔**。移除反馈后，每种配比从长度 64 起在奇偶校验和交换 S_5 上都接近随机水平。使用四 token 的 chunk 时，RLT-2 的 64 位奇偶校验准确率保持在 $98.99 \pm 1.66\%$，但 64 次操作的交换 S_5 从 100% 降至 $19.60 \pm 6.54\%$；置换跟踪从逐 token 反馈中获益最多。

Feedback Transformer、Recurrent Transformer、Full-bandwidth Transformer、T^2MLR 和 Latent Recurrent Transformer 也通过反馈在 token 之间传递信息 (Fan et al., 2020; Oncescu et al., 2026;

Wang et al., 2026; Cai et al., 2026; Huang et al., 2026)。RLT 的反馈贯穿整个解码器，并与并行的编码器记忆分离；本文还把编码与递归更新之间的深度分配作为设计变量进行研究（第 4 节）。

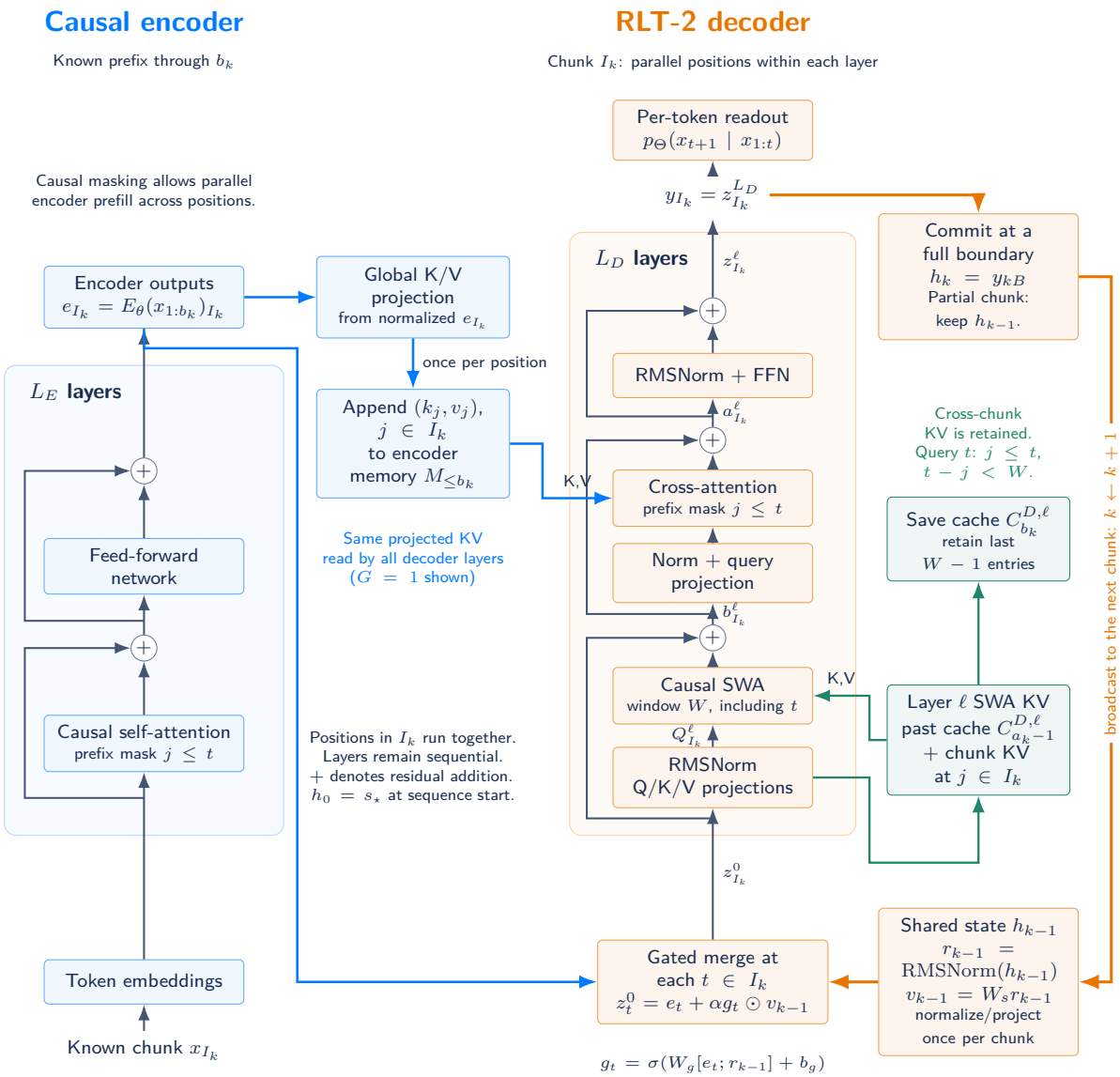


图 2 RLT-2 单个 chunk 的架构。计算自下向上进行，两个堆栈按标注深度重复。各位置 $t \in I_k$ 使用自己的门控，将编码器输出与前一边界的状态 h_{k-1} 融合。状态归一化和投影在 chunk 内共享。每个解码器层依次执行因果 SWA、仅访问前缀的编码器记忆注意力和 FFN，各子层均有残差连接。$G = 1$ 时所有解码器层读取同一组投影后的编码器 KV，各层分别跨 chunk 边界保留自己的 SWA KV。每个输出 y_t 都预测下一个 token，只有完整边界的 y_{kB} 成为下一个反馈状态。未完成的 chunk 在继续计算时保留 h_{k-1}。蓝色箭头表示编码器特征和记忆，绿色表示解码器 KV，橙色表示 chunk 级反馈。

2 方法

本节定义 RLT-1，以及两种改变其反馈间隔的对照。

2.1 序列与状态

设 $x_{1:S}$ 为以 $x_1 = \text{BOS}$ 开始的一条独立序列。推理时，$x_{1:T}$ 为提示，后续 token 构成回复。模型在 T 前后使用同一条件分布。令 L_E 和 L_D 分别为编码器与解码器深度，d 为残差宽度。编码器表示 $e_t \in \mathbb{R}^d$ 和递归输出 $s_t \in \mathbb{R}^d$ 均写作列向量。

完整的解码器状态为 $H_t = (s_t, C_t^D)$。缓存 C_t^D 保存各解码器 SWA 层保留的键值投影。窗口大小 $W \geq 1$ 包含当前 token，因此每层在一次更新后至多保留 $W-1$ 个历史位置。所有参数（包括可学习初始状态 $s_\star$）统记为 Θ。编码器和解码器分别记作 E_θ 和 D_ϕ。

2.2 因果编码器与记忆

对于已观察到的前缀，计算

$$e_{1:T} = E_\theta(x_{1:T}). \tag{2.1}$$

每个编码器层内的位置可使用因果掩码一起处理。对于解码器记忆组 $g \in \{1, \ldots, G\}$，构造

$$k_t^g = \mathcal{P}_K^g(e_t, t), \qquad v_t^g = W_V^g \,\text{RMSNorm}_E(e_t), \qquad M_{\leq t}^g = \{(k_j^g, v_j^g)\}_{j=1}^t. \tag{2.2}$$

键映射包含归一化、投影以及所采用的位置变换。解码器第 ℓ 层读取组 $g(\ell)$：$G = 1$ 表示各层共享记忆，$G = L_D$ 则允许各层使用独立投影。两类存储分别由不同的注意力子层读取：交叉注意力读取覆盖整个前缀的编码器记忆 $M_{\leq t}$，因果 SWA 读取有界窗口内的解码器 KV。

2.3 每个 token 使用同一状态转移

在 BOS 之前仅初始化一次：

$$H_0 = (s_\star, \varnothing). \tag{2.3}$$

对每个已观察到或已采样的 token，执行

$$u_t = \text{Merge}(e_t, s_{t-1}), \tag{2.4}$$

$$H_t = (s_t, C_t^D) = D_\phi(u_t; M_{\leq t}, C_{t-1}^D, t), \qquad t \geq 1, \tag{2.5}$$

$$p_\Theta(x_{t+1} \mid x_{1:t}) = \text{softmax}\big(W_o \,\text{RMSNorm}_o(s_t)\big)_{x_{t+1}}. \tag{2.6}$$

对 $H = (s, C^D)$，定义 $F_t(H) = D_\phi(\text{Merge}(e_t, s); M_{\leq t}, C^D, t)$。计算编码器特征后，解码器依次处理提示中的每个位置，以预测第一个回复 token：

$$H_T = F_T \circ F_{T-1} \circ \cdots \circ F_1(H_0). \tag{2.7}$$

生成从 H_T 的两个分量继续。根据 s_T 采样 x_{T+1} 后，增量编码该 token，追加由编码器生成的 KV，并计算 $H_{T+1} = F_{T+1}(H_T)$，其中包含各解码器 SWA 缓存的更新。

RNN 与状态空间视角。 RLT-1 可理解为一种循环神经网络（RNN），其递归单元由门控融合和整个 Transformer 解码器组成。该单元在不同 token 之间复用同一组参数，而解码器的各层可以使

用不同参数。对于解码器状态 $H = (s, C^D)$、编码器表示 e 和编码器记忆 M，定义状态转移函数与输出函数：

$$\mathcal{F}_\Theta(H; e, M, t) := D_\phi\big(\mathrm{Merge}(e, s); M, C^D, t\big), \tag{2.8}$$

$$\mathcal{G}_\Theta(H) := \mathrm{softmax}\big(W_o\,\mathrm{RMSNorm}_o(s)\big). \tag{2.9}$$

相应的非线性状态空间方程为

$$H_t = \mathcal{F}_\Theta(H_{t-1}; e_t, M_{\le t}, t), \tag{2.10}$$

$$p_\Theta(\cdot \mid x_{1:t}) = \mathcal{G}_\Theta(H_t). \tag{2.11}$$

状态转移同时更新反馈向量 s_t 和解码器缓存 C_t^D；输出函数将 s_t 映射为下一个 token 的概率分布。编码器提供条件输入 e_t 和 $M_{\le t}$，因此前文的记号对应于 $F_t(H) = \mathcal{F}_\Theta(H; e_t, M_{\le t}, t)$。在宽度、解码器深度和 SWA 窗口固定时，该解码器状态的大小有界，而编码器记忆 $M_{\le t}$ 随序列长度增长。

S4 和 Mamba 的状态空间模型（SSM）核心在各层输入固定后，对前一状态执行仿射更新；Mamba 的更新系数依赖输入 (Gu et al., 2021; Gu and Dao, 2023)。S4 利用时不变性进行卷积计算，Mamba 则对输入相关的仿射更新执行并行扫描。RLT-1 的门控融合与解码器则非线性地依赖前一解码器状态，因此这些仿射扫描算法不能直接用于其状态转移。

2.4 提示预填充与增量解码

预填充先计算提示的因果编码器表示与记忆，再依次求出 $H_1, \ldots, H_T$。即使整个提示的记忆已经可用，解码器在位置 t 的注意力仍限制为 $M_{\le t}$。生成阶段使用

$$(e_t, C_t^E) = E_\theta^{\mathrm{step}}(x_t, C_{t-1}^E) \tag{2.12}$$

更新编码器缓存 C^E，随后追加记忆并更新解码器。若保留编码器缓存，已观察到的 token 可以分块编码。解码器仍逐 token 顺序处理：每次更新都产生后续位置需要的隐藏状态与 SWA KV。

2.5 融合与解码器模块

采用以下门控融合：

$$r_{t-1} = \mathrm{RMSNorm}_s(s_{t-1}), \tag{2.13}$$

$$g_t = \sigma\big(W_g[e_t; r_{t-1}] + b_g\big), \tag{2.14}$$

$$u_t = e_t + \alpha\, g_t \odot W_s r_{t-1}. \tag{2.15}$$

其中 $W_g \in \mathbb{R}^{d\times 2d}$，$W_s \in \mathbb{R}^{d\times d}$，$\alpha$ 控制反馈幅度。主要对比实验使用 $\alpha = 0.1$。

从 $z_t^0 = u_t$ 开始，每个解码器模块依次应用因果 SWA、编码器记忆交叉注意力和前馈网络 (FFN)：

$$q_t^{D,\ell} = \mathcal{P}_Q^{D,\ell}(z_t^{\ell-1}, t), \tag{2.16}$$

$$k_t^{D,\ell} = \mathcal{P}_K^{D,\ell}(z_t^{\ell-1}, t), \qquad v_t^{D,\ell} = W_V^{D,\ell}\,\mathrm{RMSNorm}_{S,\ell}(z_t^{\ell-1}), \tag{2.17}$$

$$b_t^\ell = z_t^{\ell-1} + \mathrm{Attn}_\ell^D\Big(q_t^{D,\ell}, \{(k_j^{D,\ell}, v_j^{D,\ell})\}_{j=\max(1,t-W+1)}^{t}\Big), \tag{2.18}$$

$$a_t^\ell = b_t^\ell + \mathrm{Attn}_\ell^M\Big(\mathcal{P}_Q^{M,\ell}(b_t^\ell, t), M_{\le t}^{g(\ell)}\Big), \tag{2.19}$$

$$z_t^\ell = a_t^\ell + \mathrm{FFN}_\ell(\mathrm{RMSNorm}_{D,\ell}(a_t^\ell)), \qquad s_t = z_t^{L_D}. \tag{2.20}$$

注意力算子包含输出投影；查询与键映射包含各自的归一化和位置变换。各层在执行 SWA 前，先由该层输入形成当前位置的 KV，因此关注当前位置不会引入循环依赖。历史解码器 KV 来自 C^D_{t-1}。更新后，在 C^D_t 中保留位置 $\max(1, t-W+2), \ldots, t$；当 $W=1$ 时，该集合为空。图 3 展开了单次 token 更新中的编码器与解码器堆栈。

2.6 递归深度与执行成本

处理长度为 T 的提示和 n 个回复 token 后，反馈路径已穿过 $(T+n)L_D$ 个解码器模块，每个 token 则执行 $L_E + L_D$ 个模块（图 4）。早期状态的影响取决于门控、投影和状态转移雅可比矩阵的乘积。长度为 T 的提示需要 T 次顺序解码器更新；不同序列的更新可以组成批次，各自维护状态和缓存。附录 B 推导计算量与缓存存储量，并证明状态和下一 token 分布不随提示与回复的划分改变。

2.7 反馈间隔：RLT-1、RLT-2 与 RLT-0

RLT-1 与两种对照只在一个量上不同，即反馈间隔 B：进入解码器输入的状态每隔多少个 token 更新一次。RLT-1 每个 token 更新一次（$B=1$），RLT-2 每个 B token 的 chunk 更新一次，RLT-0 从不更新（$B=\infty$）。三者都保留 RLT-1 的因果编码器、编码器记忆、解码器 SWA、仅访问前缀的记忆注意力和输出头。

RLT-2 在 B 个 token 组成的 chunk 内保持反馈状态不变，并在 chunk 边界更新（图 2）。边界以 BOS 为基准，第 k 个 chunk 覆盖已知位置 $I_k = \{(k-1)B+1, \ldots, \min(kB, T)\}$。反馈状态从 $h_0 = s_\star$ 开始；第 k 个 chunk 的每个位置通过各自的门控，将编码器输出与同一个边界状态融合：

$$r_{k-1} = \mathrm{RMSNorm}_s(h_{k-1}), \qquad g_t = \sigma\big(W_g[e_t; r_{k-1}] + b_g\big), \tag{2.21}$$

$$z^0_t = e_t + \alpha\, g_t \odot W_s r_{k-1}, \qquad t \in I_k, \tag{2.22}$$

$$h_k = y_{kB} \quad \text{当} kB \le T. \tag{2.23}$$

其中 $y_t = z^{L_D}_t$ 是位置 t 的最终解码器输出；每个 y_t 都预测 x_{t+1}，但只有完整 chunk 的末位置输出成为下一个反馈状态。解码器对 chunk 内所有已知位置同时应用式 (2.20)，因果 SWA 也读取之前 chunk 保留的键。由于 h_{k-1} 只依赖该 chunk 之前的 token，因果掩码保证每个 y_t 仅依赖 $x_{1:t}$。末尾未完成的 chunk 使反馈状态停留在前一个完整边界，因此移动提示与回复的分界不会改变计算。当 $B=1$ 时，每个 token 都结束一个 chunk，RLT-2 与 RLT-1 完全相同。

RLT-0 对应 $B=\infty$ 的极限：永远不会到达 chunk 边界，因此没有解码器输出被反馈。$B=\infty$ 的 RLT-2 仍会让每个位置与常量初始状态 $s_\star$ 融合；RLT-0 去掉这一常量融合，连同 $s_\star$、状态归一化、门控和反馈投影一并移除，解码器直接接收 $z^0_t = e_t$（图 15）。

表 1 和图 5 在相同网络尺寸下比较三种反馈间隔。对于长度为 T 的已知前缀，解码器需要 $\lceil T/B \rceil L_D$ 个串行模块阶段，每次批量矩阵计算至多包含 $\min(B,T)$ 个位置：RLT-1 为 TL_D 个阶段，RLT-0 为 L_D 个。三者都执行 TL_D 次解码模块计算，生成时每步输出一个 token。因此，增大 B 是以反馈频率换取已知 token 的并行性。附录 I 给出分块训练、预填充和增量生成流程。

RLT-2 在任何 B 下参数形状都相同，但 chunk 大小会改变计算：它决定每个 token 读取哪一个反馈状态，因此同一组权重在不同的 B 下定义的是不同的函数。chunk 越大，解码器的串行路径越短；chunk 越小，奇偶校验与交换 S_5 的外推越好（第 3 节）。在种子 42、4 + 4 的 CPU 计时中，chunk4 和 RLT-0 的每步训练速度分别为 RLT-1 的 2.27 倍和 4.17–4.30 倍（附录 J.11）。由此可以设想一种分阶段调度：预训练初期使用较大的 B，以高并行度处理大部分 token；中期训练和后训练再逐步减小 B，直至 $B=1$，使模型适应高频反馈。推理与 RL 重放使用最后一个训练阶段的 chunk 大小。

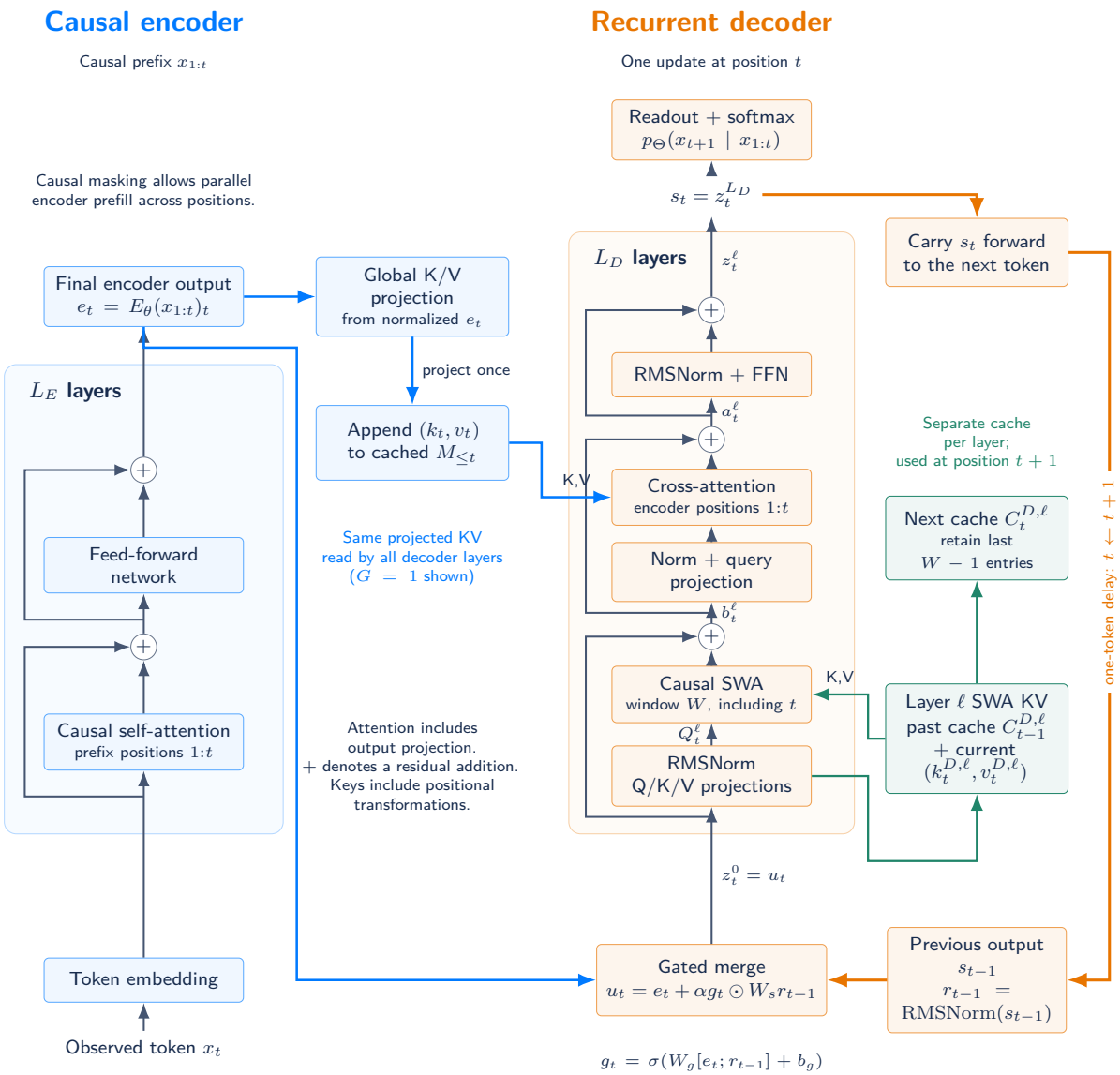


图 3 单次 token 更新的 RLT-1 架构。计算自下向上进行，各虚线框中的模块按标注深度重复。编码器输出同时用于门控融合和全局 KV 投影。当 $G = 1$ 时，所有解码器层使用各自的查询，读取同一份已投影并缓存的全局 KV。每个解码器层从自己的输入构建独立的 SWA KV，先关注局部窗口，再读取全局记忆，最后应用 FFN。残差路径绕过各注意力或 FFN 子层。解码器最终输出 s_t 进入下一个 token 的融合操作，各层同时保留自己的 SWA 缓存。蓝色箭头表示编码器特征和全局记忆，绿色表示局部 KV，橙色表示递归隐藏状态。

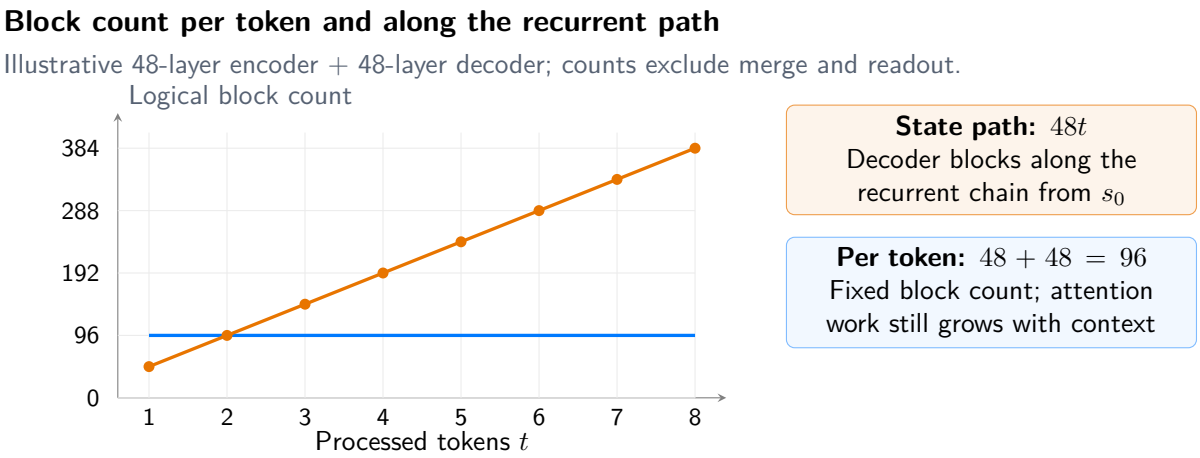


图 4　递归路径长度与每 token 的模块数。当 $L_E = L_D = 48$ 时，处理 t 个 token 后，路径穿过 $48t$ 个解码器模块。每个 token 执行 96 个编码器和解码器模块。橙色统计递归路径上的解码器模块，蓝色统计每个 token 在两个堆栈中执行的模块。

2.8　训练与状态重放

训练使用教师强制，并对所选的下一 token 目标计算交叉熵。完整的随时间反向传播（BPTT）经过递归输出、解码器 KV 和编码器记忆；掩去某个目标的损失不会跳过对应的状态更新。在强化学习（RL）中，若要在参数更新后评估已采样回复的概率，需用当前参数重放历史并重建状态。附录 D 给出预训练、监督微调和策略梯度的目标函数与重放条件。

3　算法任务实验

我们研究解码器深度与反馈频率如何影响长度泛化。实验覆盖加法、奇偶校验、带括号和无括号模运算，以及标准和交换两种 S_5 状态跟踪任务。先比较编码器与解码器的层数分配，再在固定配比下改变反馈间隔。另在奇偶校验与交换 S_5 上报告一组独立的深度扩展实验，将总层数从八层增至二十层（第 3.4 节）。附录 J.2 给出完整的六任务 2,000 步实验；附录 J.10 和 J.11 给出所有反馈变体及 5,000 步模运算对照。

3.1　模型与评估

层数分配与反馈频率实验使用八个逻辑层，宽度为 512，FFN 宽度为 1,365，注意力头数为四。我们将不共享权重的 RLT-1 Layer 4+4、5+3、6+2、7+1 和 8+0 与八层、仅含解码器的 Transformer（GPT Layer 8）比较。RLT-1 使用大小为八的 SWA 窗口、一个共享编码器记忆组，以及反馈系数 $\alpha = 0.1$。$8 + 0$ 虽然没有解码器模块，仍保留门控递归融合。RLT-1 为 26.10–28.73M 参数，Transformer 为 25.31M（表 5）。

八层模型的准确率比较均使用初始化种子 42、43 和 44，同一任务内共享训练数据流和留出样本。全局批量为 512，微批量为 32。AdamW 使用 $(\beta_1, \beta_2) = (0.9, 0.95)$，权重衰减为 0.1，梯度范数裁剪阈值为 1。基础学习率在前 200 步预热至 10^{-4}，随后在训练结束时降至 5×10^{-6}。奇偶校验、加法和 S_5 训练 2,000 步；本节的模五对照均训练 5,000 步，余弦调度覆盖相应训练预算。模

指标	RLT-1 ($B=1$)	RLT-2 (B)	RLT-0 ($B=\infty$)
已知 token 的每批解码位置数	1	至多 B	T
训练前向 / 预填充的解码模块阶段数	TL_D	KL_D	L_D
完整 BPTT 的反向依赖深度	$O(TL_D)$	$O(KL_D)$	$O(L_D)$
处理 T 个 token 的反馈状态更新数	T	$\lfloor T/B \rfloor$	0
处理 T 个 token 的解码模块计算数	TL_D	TL_D	TL_D
处理 T 个 token 的融合门控计算数	T	T	0
已知 token 的共享反馈投影数	T	K	0
生成 N 个 token 的自回归步骤数	N	N	N
每个新处理 token 的模块计算数	$L_E + L_D$	$L_E + L_D$	$L_E + L_D$
持久反馈状态	d 个值	d 个值	无

表 1 匹配网络尺寸时的执行成本。T 为已知输入 token 数，$K = \lceil T/B \rceil$，N 为生成的 token 数；三个变体共用 L_E 阶段的因果编码器预填充。省略公共的输出头及编码器记忆投影。RLT-2 复用归一化与投影后的边界状态，门控则依赖每个 token。反向深度统计解码器依赖图，编码器反向传播为公共部分。这些计数描述依赖关系和算术计算量。

型使用相同的宽度 μP 设置，以四个 CPU 线程执行 FP32 计算。RLT-1 和 RLT-2 的 TBPTT128 覆盖全部训练序列，因此没有截断梯度。

奇偶校验在 3–40 位二进制串上训练；S_5 在 32 次操作上训练，输入采用全部 120 个置换，或恒等置换与十个单次对换（交换变体），遵循 Grazzi et al. (2024)。无括号模五表达式使用 $+$、$-$ 和 $\times$，遵循乘法优先级，训练长度为 3–39 的奇数；带括号表达式使用长度为 3–40 的表达式树。每种测试长度使用共享的 1,024 个样本，评分指标为最终答案或状态准确率。加法训练随机采样 1–8 位操作数；每种测试位数使用共享的 256 对操作数，测量教师强制答案 token 准确率。各次运行按最低分布内（ID）验证样本损失选择泛化测试的检查点，完全同分时取最早一步。测试分数不参与检查点选择。均值与样本标准差（SD，$n = 3$）反映固定数据流上的初始化差异。附录 J 和 J.7 给出生成器、指标定义与检查点步数。

3.2 长度外推与解码器层数分配

RLT-1 在远超训练长度的奇偶校验与交换 S_5 序列上仍保持较高准确率（图 6）。在 256 位奇偶校验上，$5+3$ 和 $7+1$ 均保持 $100 \pm 0\%$，Transformer 为 $50.07 \pm 1.63\%$。优势也出现在学习过程中：第 500 步时，$6+2$ 的验证准确率达到 $99.44 \pm 0.98\%$，Transformer 为 $48.48 \pm 0.53\%$（图 7）。在另一组种子 42 的十六层实验中，RLT-1 Layer $8+8$、$9+7$、$11+5$ 和 $16+0$ 在 256 位上达到 100%，GPT Layer 16 为 49.41%（附录 J.9）。

交换 S_5 更适合分配较多解码器层。在 256 次操作，即训练长度的八倍处，$4+4$ 的最终状态准确率为 $97.30 \pm 2.76\%$，Transformer 为 $0.85 \pm 0.30\%$。在 512 次操作处，$4+4$ 仍有 $55.70 \pm 25.78\%$，Transformer 仍为 $0.85 \pm 0.30\%$。$7+1$ 和 $8+0$ 虽然在训练长度上准确率很高，在 256 次操作处已接近均匀预测参考值。图 14 另给出前缀 token 与全序列准确率。

训练 5,000 步后，RLT-1 在中等测试长度的模运算上领先。在长度为 63 的无括号表达式上，RLT-1 Layer $6+2$ 达到 $93.36 \pm 5.69\%$，Transformer 为 $33.20 \pm 2.33\%$。在长度为 64 的带括号表达式上，$5+3$ 达到 $67.97 \pm 2.91\%$，Transformer 为 $46.71 \pm 1.21\%$。表达式继续变长后准确率下降，且多个无括号模五配置在种子之间差异较大。

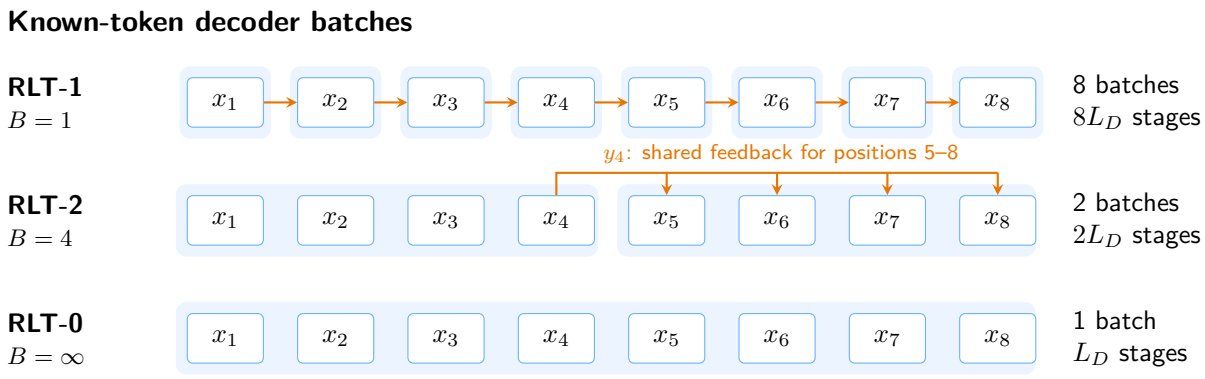


图 5　反馈间隔增大时，八个已知 token 的解码调度。各行依次为 RLT-1（$B = 1$）、RLT-2（$B = 4$）和 RLT-0（$B = \infty$）。每个阴影组为一个逐层解码批次，包含 L_D 个串行模块。橙色箭头表示反馈依赖，RLT-2 将 y_4 广播到第二个 chunk 的全部四个位置，RLT-0 则没有反馈。各行均使用因果 SWA，并在批次间保留缓存。自回归生成时，三个变体均逐 token 处理输入。

3.3　反馈频率与并行计算

我们在每种配比下比较反馈间隔 $B = 1$（RLT-1）、$B = 4$（RLT-2 chunk4）和 $B = \infty$（RLT-0），数据、优化器和种子与 RLT-1 相同（第 2.7 节）。每种配比下，RLT-0 比 RLT-1 少 787,968 个参数；RLT-2 与 RLT-1 参数相同。图 8 在 $4 + 4$ 下比较全部变体，附录 J.10 给出所有配比。

在 64 位奇偶校验上，RLT-1 达到 $100 \pm 0\%$，chunk4 达到 $98.99 \pm 1.66\%$，RLT-0 则接近随机水平，为 $50.23 \pm 3.80\%$。交换 S_5 对反馈间隔更敏感：在 64 次操作处，RLT-1 为 $100 \pm 0\%$，chunk4 为 $19.60 \pm 6.54\%$。RLT-0 和 Transformer 在该长度上接近 1/120 的均匀预测参考值。与 RLT-1 相比，四 token 的 chunk 在 64 位奇偶校验上下降约 1 个百分点，在 64 次操作的交换 S_5 上下降约 80 个百分点。

分块的影响还取决于配比（图 22 和 23）。chunk4 $4+4$ 在 256 位奇偶校验上仍有 $69.34 \pm 24.17\%$，而 chunk4 $7+1$ 和 $8+0$ 在 64 位时已接近随机水平。在交换 S_5 上，chunk4 $5+3$ 在 48 次和 64 次操作处分别为 $79.92 \pm 10.89\%$ 和 $50.16 \pm 18.77\%$，而 chunk4 $4+4$ 在 64 次操作处为 $19.60 \pm 6.54\%$。

3.4　奇偶校验与交换 S_5 的深度扩展

在独立的 48 次运行中，我们将不共享权重的 RLT-1 $4+4$、$6+6$、$8+8$ 和 $10+10$ 分别与 8、12、16 和 20 层 Transformer 比较。各模型使用初始化种子 42–44，训练 2,000 步，宽度为 512，FFN 宽度为 1,365，全局批量为 512，并在同一任务内共享训练数据流。本批实验使用 Apple M4 CPU FP32 执行环境和独立冻结的实现快照；其中的八层对照与前述层数分配实验分别报告。同一深度对照内匹配层数、训练样本数和优化器步数；RLT-1 的参数量多 10.40%–13.48%。附录 J.1 给出实验协议、参数量和来源信息。

在奇偶校验上，$6+6$ 和 $8+8$ 的三个种子在截至 256 的全部测试长度上均达到 100%（图 9）。在 256 位处，对应 Transformer 分别为 $49.93 \pm 1.86\%$ 和 $48.96 \pm 1.59\%$。$4+4$ 和 $10+10$ 各有一个种子在 256 位处接近随机水平，汇总结果分别为 $83.72 \pm 28.19\%$ 和 $84.11 \pm 27.51\%$。

在 500 次操作的交换 S_5 上，RLT-1 从 4+4 的 $37.08 \pm 31.11\%$，依次提高到 6+6 的 $76.46 \pm 38.09\%$、8+8 的 $92.29 \pm 10.08\%$ 和 10+10 的 $99.02 \pm 1.18\%$。二十层 Transformer 在该长度上为 $0.75 \pm 0.06\%$。$6+6$ 的三个种子分别为 99.90%、96.97% 和 32.52%。

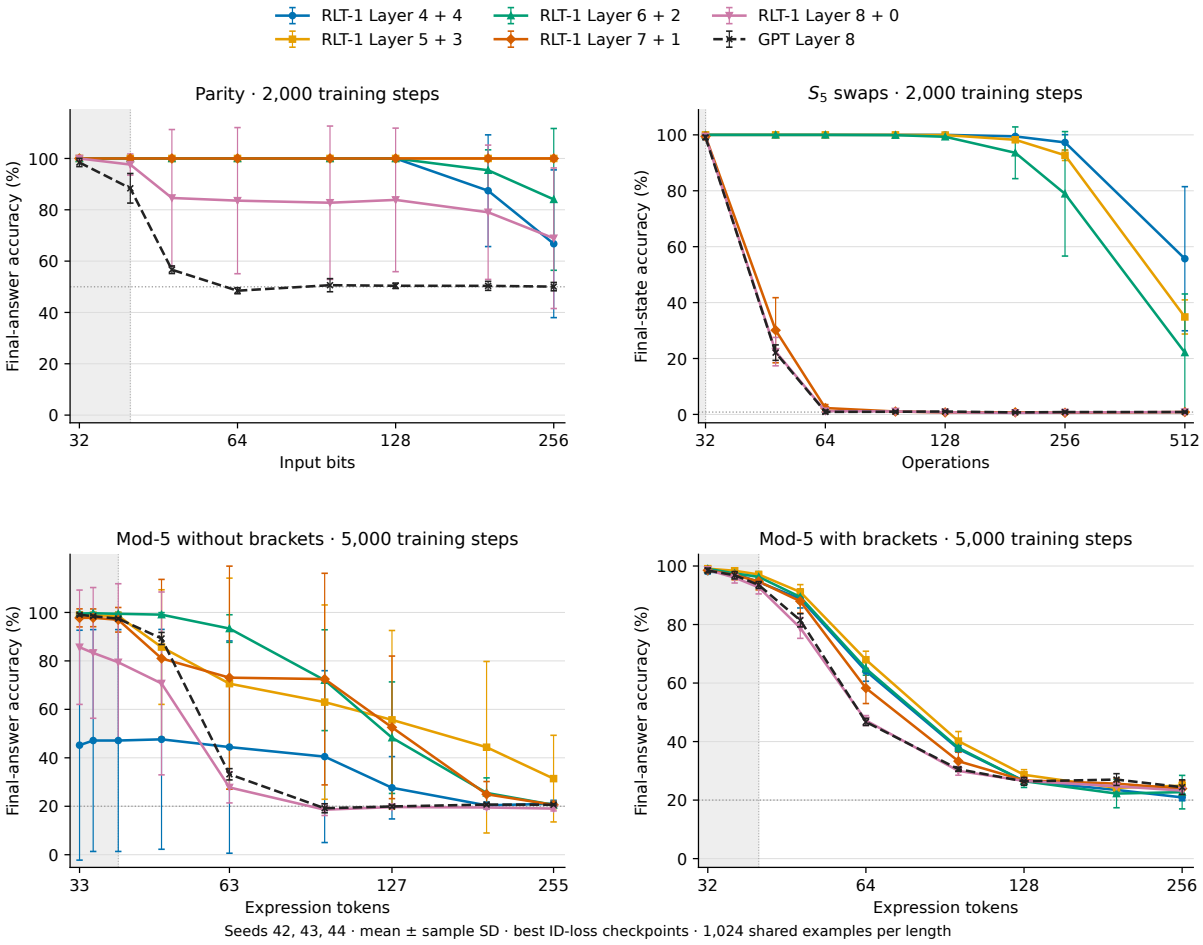


图 6 不同编码器与解码器配比的长度泛化。五种 RLT-1 配比与 GPT Layer 8 均展示完整测试长度网格。上排的奇偶校验与交换 S_5 使用 2,000 步训练结果；下排模五任务的所有模型均训练 5,000 步。各次运行选择最低 ID 损失检查点，同分时取最早一步。点与未裁剪的误差棒表示种子 42、43、44 的均值 ± 样本 SD，每种长度使用相同的 1,024 个样本。灰色区域标记训练长度，水平虚线标记均匀预测准确率。无括号模五标注实际奇数表达式长度。完整的六任务 2,000 步对照见图 13。

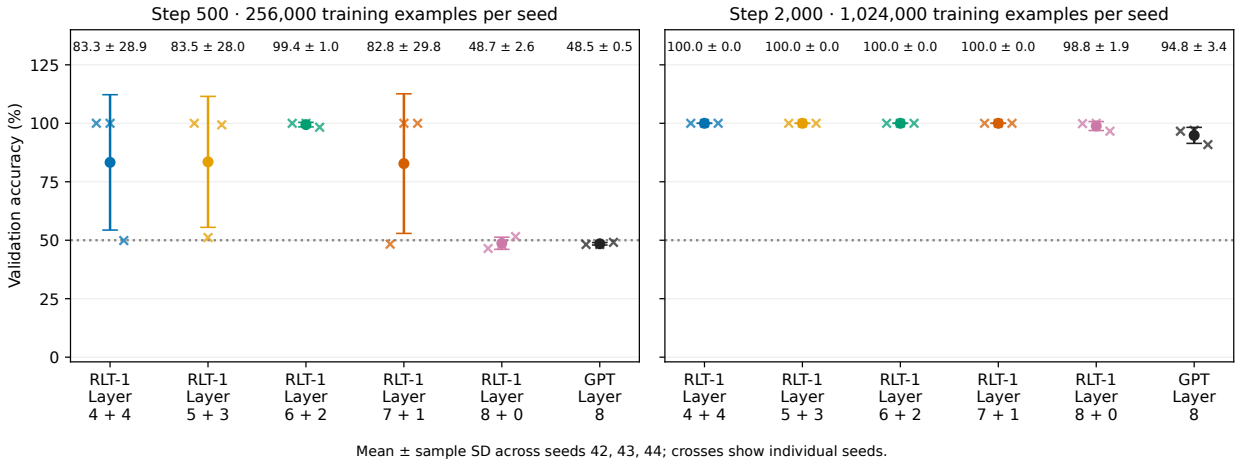


图 7　训练 500 步与 2,000 步后的奇偶校验结果。圆点和误差棒表示种子 42、43 和 44 的均值与样本 SD；叉号表示单个种子，并作少量水平偏移。SD 误差棒可能超过 100%。所有模型使用相同的 768 个验证样本；两个面板分别对应每个种子使用 256,000 和 1,024,000 个训练样本。

3.5　加法与标准 S_5

所有模型在训练位数范围内的加法验证集上均达到 100% 教师强制准确率，但超过八位后准确率下降；八层模型在 32 位时的均值为 14.89%–16.84%。标准 S_5 仍然困难，各反馈变体的最终状态准确率接近 1/120 参考值（图 24）。附录 J 给出完整的六任务曲线、终点结果表和奇偶校验各种子轨迹；附录 J.12 给出加法反馈系数消融。

4　相关工作

Transformer 与 RNN 混合模型。　Chen et al. (2018) 将 Transformer 编码器与基于 LSTM 的 RNMT+ 解码器结合，用于机器翻译。RLT-1 也将并行编码与递归解码结合。其解码器由带编码器记忆交叉注意力和 SWA 的 Transformer 模块组成，并在因果语言模型中处理提示与回复的全部 token。

编码器生成的记忆。　YOCO 为上层交叉解码器构建可复用的 KV 记忆，并允许预填充提前退出 (Sun et al., 2024)。DeepSeek-V4.1-Flash 从最终编码器状态投影得到解码器的全局 KV，并维护独立的解码器 SWA (DeepSeek-AI, 2026)。RLT-1 也从编码器状态构建全局记忆，但其递归解码器必须处理每个提示 token。

时间反馈。　Feedback Transformer 对每个 token 的各层表示进行可学习的加权求和，并让后续 token 在每一层通过注意力读取这份共享记忆 (Fan et al., 2020)。RLT-1 将前一步的最终解码器输出送入解码器输入处的门控融合，同时保留独立的编码器全局记忆和各层解码器 SWA 缓存。Recurrent Transformer 根据每层的输出构建该层的持久 KV，并给出精确的分块调度来改善内存数据搬运 (Oncescu et al., 2026)。RLT-1 的递归依赖贯穿整个解码器，而 Recurrent Transformer 的反馈发生在各层内部。

跨 token 的潜在状态反馈。　Full-bandwidth Transformer 通过门控线性单元，将上一个 token 的顶层隐藏状态与下一个 token 的嵌入结合，同时保留 Transformer 堆栈与 KV 缓存 (Wang et al., 2026)。它通过多轮计算，在轮次之间平移隐藏状态，实现 token 并行的教师强制训练，并混入前缀

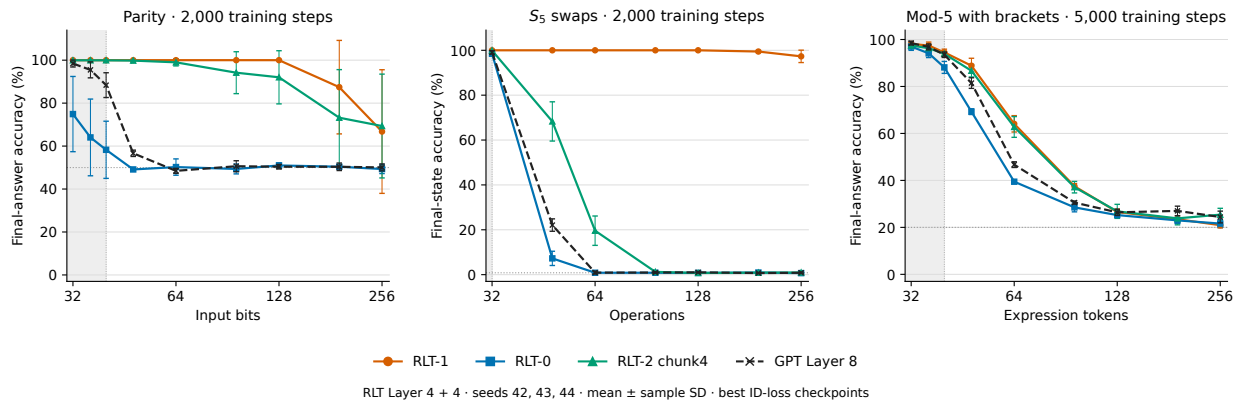


图 8 固定 $4+4$ 配比下的反馈频率比较。展示 RLT-1（$B=1$）、RLT-2 chunk4（$B=4$）、RLT-0（$B=\infty$）和 GPT Layer 8 在全部原生测试长度上的最终答案或状态准确率。奇偶校验与交换 S_5 训练 2,000 步，带括号模五训练 5,000 步。所有模型均使用种子 42–44 及其最低 ID 损失检查点。误差棒为未经裁剪的样本 SD，灰色区域标记训练长度，虚线标记均匀预测参考值。

计算以处理提示与生成之间的差异。RLT-1 将输出到输入的反馈放在编码器与解码器接口处，训练时在完整历史上顺序重放解码器。

T^2MLR 将上一个 token 缓存的中间层表示反馈到当前位置的较早层 (Cai et al., 2026)。其实验发现，中间层之间的递归可能优于贯穿整个网络的递归。训练通过固定次数的 Jacobi 迭代近似时间状态，并单独控制反向传播深度。RLT-1 的反馈贯穿整个解码器，训练时以 BPTT 重放完整历史，也可选择截断。

潜在递归语言模型。 Latent Recurrent Transformer（LRT）通过 KV 投影和残差注入复用上一个 token 的高层状态，保留仅含解码器的主干，并为每个生成 token 执行一次前向计算 (Huang et al., 2026)。其交错并行训练从共享状态缓冲区中读取状态，逐步细化部分位置。在 RL 中，LRT 使用已分离梯度的 rollout 状态初始化该缓冲区，以减小 rollout 与重算之间的差异。RLT-1 将编码器记忆与递归解码器分开，并在当前参数下重建完整历史，以执行精确的策略重放。

连续潜在计算。 Coconut 在潜在推理阶段将最后的隐藏状态作为下一个输入嵌入反馈，并通过课程学习逐渐以连续状态替代文本推理步骤 (Hao et al., 2024)。PonderLM-2 在预训练期间于普通 token 之间插入潜在步骤，并使用 Jacobi 迭代并行近似这些步骤的递归依赖 (Zeng et al., 2025)。RLT-1 对提示或回复中的每个普通 token 更新一次状态，不增加潜在位置。

块级与段级递归。 Block-Recurrent Transformers 通过注意力和门控更新持久状态向量，并在每个递归步骤中并行处理一块 token (Hutchins et al., 2022)。其 block-feedback 变体允许所有层通过交叉注意力读取前一块的递归状态。Recurrent Memory Transformer 在每段末尾添加写入记忆 token，将其最终表示作为下一段的记忆输入，并沿该连接执行 BPTT (Bulatov et al., 2022)。RLT-2 与之最接近：它在块边界处用最后一个 token 的最终解码器输出更新反馈状态，并通过门控送入下一块的每个解码器输入，无需专门的记忆 token 或独立的递归状态向量组。RLT-1 是其 $B=1$ 的情形，每个 token 都反馈，因此每个块内的解码器必须顺序执行。

跨深度的权重共享。 Universal Transformers 在深度方向共享权重，并可按位置自适应调整表示细化的步数 (Dehghani et al., 2018)。Saunshi et al. (2025) 研究重复应用共享 Transformer 堆栈如何增加推理的有效深度；递归深度语言模型则在推理时调整潜在计算量 (Geiping et al., 2025)。DeepLoop 分析权重共享的循环 Transformer 中重复访问参数对残差缩放的影响，并为 Post-LN 架

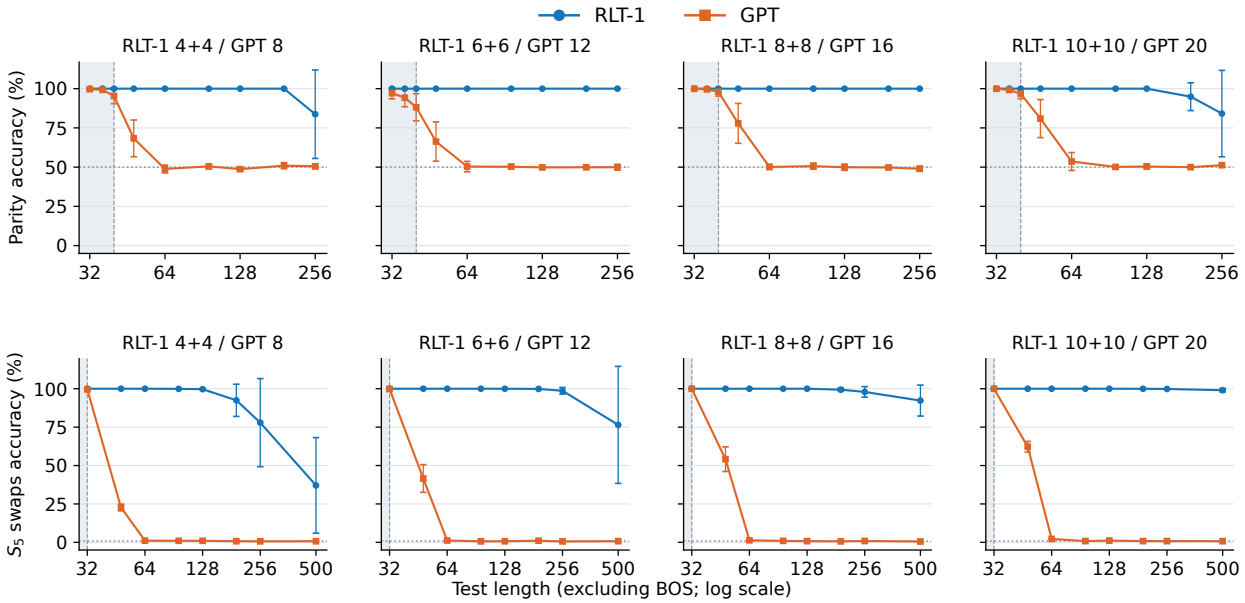


图 9　独立三种子实验的深度扩展。各列将 RLT-1 $4+4$、$6+6$、$8+8$ 和 $10+10$ 与 GPT Layer 8、12、16 和 20 配对。奇偶校验在 3–40 位上训练，交换 S_5 在 32 次操作上训练。全部运行训练 2,000 步，选择最低 ID 损失检查点，同分时取最早一步。每种长度使用相同的 1,024 个样本，点表示最终答案或状态准确率，未经裁剪的误差棒表示种子 42–44 的样本 SD。灰色区域标记训练长度，虚线标记均匀预测参考值，长度不含 BOS。奇偶校验与交换 S_5 最长分别测试到 256 和 500。参数量见表 2。

构推导考虑循环次数的缩放规则 (Li et al., 2026)。这些模型在每个位置上沿深度方向循环；RLT-1 的递归跨越 token，编码器与解码器之间的权重共享是可选的。

5　结论

我们提出了递归循环 Transformer（RLT）：它将并行的因果编码器与逐 token 反馈最终状态的解码器结合，使计算路径随序列长度增长，而每个 token 的计算量不变。在八层规模下，RLT 将奇偶校验和交换 S_5 跟踪泛化到远超训练长度的序列，而八层 Transformer 在这些长度上停留在随机水平；在超出训练长度的模运算上，RLT 的最佳配比也优于 Transformer。移除反馈后，奇偶校验和交换 S_5 上的提升随之消失。在总计八层时，编码器与解码器的配比是一个设计维度：奇偶校验在一层或三层解码器时泛化最好，交换 S_5 则至少需要两层解码器，并随解码器层数增加而提升，直到四层。在独立的深度扩展实验中，RLT $10+10$ 在 500 次操作的交换 S_5 上达到 $99.02 \pm 1.18\%$。反馈间隔 B 是第二个设计维度，在并行性与准确率之间权衡：$B = 4$ 可在每个 chunk 内并行处理已知 token，并使 64 位奇偶校验保持在 99%，而交换 S_5 从逐 token 反馈中获益最多。由于任何 B 下参数形状都相同，模型可以在预训练中使用大 chunk 以提高并行度，再在中期训练和后训练中逐步缩小到 $B = 1$。

致谢

我们使用大语言模型改进了本文的措辞。

参考文献

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Recurrent memory transformer. *arXiv preprint arXiv:2207.06881*, 2022. URL https://arxiv.org/abs/2207.06881.

Ziyang Cai, Xingyu Zhu, Yihe Dong, Yinghui He, and Sanjeev Arora. T^2MLR: Transformer with temporal middle-layer recurrence. *arXiv preprint arXiv:2607.15178*, 2026. URL https://arxiv.org/abs/2607.15178.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Niki Parmar, Mike Schuster, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. The best of both worlds: Combining recent advances in neural machine translation. *arXiv preprint arXiv:1804.09849*, 2018. URL https://arxiv.org/abs/1804.09849.

DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression. Technical report, DeepSeek-AI, 2026. URL https://huggingface.co/deepseek-ai/DeepSeek-V4.1-Flash/resolve/main/DeepSeek_V41_Tech_Report.pdf. Sections 2.2 and 3.2.2; publicly available from the official DeepSeek model repository.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018. URL https://arxiv.org/abs/1807.03819.

Angela Fan, Thibaut Lavril, Edouard Grave, Armand Joulin, and Sainbayar Sukhbaatar. Addressing some limitations of transformers with feedback memory. *arXiv preprint arXiv:2002.09402*, 2020. URL https://arxiv.org/abs/2002.09402.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *arXiv preprint arXiv:2502.05171*, 2025. URL https://arxiv.org/abs/2502.05171.

Riccardo Grazzi, Julien Siems, Arber Zela, Jörg K. H. Franke, Frank Hutter, and Massimiliano Pontil. Unlocking state-tracking in linear rnns through negative eigenvalues. *arXiv preprint arXiv:2411.12537*, 2024. URL https://arxiv.org/abs/2411.12537.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. URL https://arxiv.org/abs/2312.00752.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021. URL https://arxiv.org/abs/2111.00396.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024. URL https://arxiv.org/abs/2412.06769.

Zeyi Huang, Xuehai He, Liliang Ren, Yiping Wang, Baolin Peng, Hao Cheng, Shuohang Wang, Pengcheng He, Jianfeng Gao, Yong Jae Lee, and Yelong Shen. Latent recurrent transformer: Architecture exploration, training strategies, and scaling behavior. *arXiv preprint arXiv:2605.26797*, 2026. URL https://arxiv.org/abs/2605.26797.

DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. Block-recurrent transformers. *arXiv preprint arXiv:2203.07852*, 2022. URL https://arxiv.org/abs/2203.07852.

Shuzhen Li, Yifan Zhang, Jiacheng Guo, Quanquan Gu, and Mengdi Wang. DeepLoop: Depth scaling for looped transformers. *arXiv preprint arXiv:2607.13491*, 2026. URL https://arxiv.org/abs/2607.13491.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2404.08819.

Costin-Andrei Oncescu, Depen Morwani, Samy Jelassi, Alexandru Meterez, Mujin Kwun, and Sham Kakade. The recurrent transformer: Greater effective depth and efficient decoding. *arXiv preprint arXiv:2604.21215*, 2026. URL https://arxiv.org/abs/2604.21215.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2502.17416.

Yutao Sun, Li Dong, Yi Zhu, Shaohan Huang, Wenhui Wang, Shuming Ma, Quanlu Zhang, Jianyong Wang, and Furu Wei. You only cache once: Decoder-decoder architectures for language models. *arXiv preprint arXiv:2405.05254*, 2024. URL https://arxiv.org/abs/2405.05254.

Xi Wang, Ziyang Cai, Zheng Zhan, Harry Dong, Ying Fan, Gustavo de Rosa, Tim Pearce, and John Langford. Full-bandwidth transformer. *arXiv preprint arXiv:2608.08888*, 2026. URL https://arxiv.org/abs/2608.08888.

Boyi Zeng, He Li, Shixiang Song, Yixuan Wang, Ziwei He, Xinbing Wang, and Zhouhan Lin. PonderLM-2: Pretraining LLM with latent thoughts in continuous space. *arXiv preprint arXiv:2509.23184*, 2025. URL https://arxiv.org/abs/2509.23184.

Yifan Zhang et al. Reliable RL scaling requires accounting for Prefill–Decode kernel mismatch. Technical report, Pretraining-RL-Science project, August 2026. URL https://github.com/yifanzhang-pro/Pretraining-RL-Science/blob/master/Prefill_Decode_Kernel_Mismatch.pdf. Dated August 6, 2026; revised August 24, 2026.

附录

A 架构与执行细节 **19**
A.1 编码器与解码器的权重共享 19

B 计算性质 **19**
B.1 提示与回复的一致性 19
B.2 预填充计算量与顺序深度 19

C 硬件执行 **20**
C.1 独立序列的批处理 20
C.2 内存访问与参数复用 20
C.3 训练内存与数值一致性 20

D 训练目标 **21**
D.1 自回归预训练 21
D.2 监督微调 21
D.3 策略梯度与重放 21

E 多轮推理与缓存复用 **22**

F 执行流程 **23**
F.1 递归提示预填充 23
F.2 生成与新的外部输入 23
F.3 预训练与监督微调 23
F.4 当前策略的 RL 重放 23

G 因果性与梯度路径 **24**

H 缓存状态何时可以复用 **25**

I RLT-2：分块并行的隐藏状态反馈 **25**
I.1 分块算子与因果掩码 25
I.2 训练与预填充算法 26
I.3 增量生成与未完成的 chunk 26
I.4 训练与推理效率 27

J 实验细节 **28**
J.1 深度扩展实验协议与终点结果 28
J.2 完整的六任务 2,000 步实验 29
J.3 参数量 35
J.4 RLT-0 的实现 35
J.5 种子聚合与指标定义 35
J.6 完整训练快照的复现 35
J.7 长度泛化协议与补充指标 37
J.8 长度为 32 和 33 时的 token 准确率曲线 42
J.9 十六层奇偶校验：最佳与最终检查点 43
J.10 反馈变体的三种子比较 45
J.11 5,000 步模五反馈变体实验 50
J.12 固定学习率调度下的加法反馈系数 55

A 架构与执行细节

A.1 编码器与解码器的权重共享

共享权重的参考 RLT-1 令 $L_E = L_D = L$。编码器第 ℓ 层的自注意力与解码器第 ℓ 层的 SWA 共享兼容的查询、键、值和输出投影；对应的 FFN 也共享权重。编码器关注自身的因果上下文，解码器 SWA 关注窗口内的解码器激活。解码器交叉注意力使用独立的查询及输出投影，以及式 (2.2) 的记忆投影，在共享注意力和 FFN 之外增加计算。编码器与解码器保留各自的归一化模块；融合与读出也是独立模块。

两次计算共享权重，但产生独立的激活。不共享权重的 E_θ, D_ϕ 使用各自的参数，递归形式相同。两种版本都可以共享记忆组；各解码器层仍维护独立的 SWA 缓存。

B 计算性质

B.1 提示与回复的一致性

命题 B.1 (服务划分不变性)**.** 固定一条独立序列的参数、token 序列、位置约定和初始状态。假设采用精确算术，因果编码器的执行在数学上等价，SWA 窗口与缓存更新相同，解码器操作是确定性的。先批量执行编码器预填充、再递归更新解码器，与逐 token 增量处理该前缀，得到相同的状态和下一 token 分布。因此，对于固定的 token 历史，移动提示与回复的划分位置不会改变条件分布。

证明. 因果编码器的等价性保证两种调度具有相同的 e_t 和 $M_{\le t}$。二者均从 $H_0 = (s_\star, \varnothing)$ 开始。若在 $t-1$ 时状态一致，式 (2.5) 在 t 时会对相同输入执行相同操作，因此二者在 t 时的状态仍一致。由于状态转移不依赖服务划分，归纳可得结论。 □

B.2 预填充计算量与顺序深度

令 $C_E^{\mathrm{pf}}(T)$ 为编码器预填充计算量，$C_M(T)$ 为记忆投影计算量，$C_D^{\mathrm{step}}(t)$ 为一次解码器计算量：它读取 t 条编码器记忆、每层至多 W 个解码器位置，并包含融合开销。于是

$$C_{\mathrm{prefill}}(T) = C_E^{\mathrm{pf}}(T) + C_M(T) + \sum_{t=1}^{T} C_D^{\mathrm{step}}(t). \tag{B.1}$$

采用稠密注意力、且 KV 宽度与模型宽度成比例时，算术计算量大致为

$$O\big((L_E + L_D)(Td^2 + T^2 d) + GTd^2 + L_D T \min(W, T) d\big). \tag{B.2}$$

预填充需要 T 次顺序执行的解码器状态转移，每次包含 L_D 个模块。即使算术复杂度与稠密 Transformer 同阶，这些顺序依赖也可能降低硬件利用率。

每个新 token 执行 $L_E + L_D$ 个模块，另加融合与记忆投影。注意力计算量仍随上下文长度增长。设有效 KV 宽度为 d_{KV}，则推理缓存的存储量近似为

$$O\big((L_E + G)t\, d_{\mathrm{KV}} + L_D \min(t, W-1) d_{\mathrm{KV}}^D + d\big). \tag{B.3}$$

其中 d_{KV}^D 为解码器 SWA 的有效 KV 宽度，$O(d)$ 项用于保存当前递归输出。SWA 项只计入保留的历史；当前位置的 KV 和训练激活需要额外存储。权重共享减少参数存储，但仍需执行两次计算并保存各自的缓存。

C 硬件执行

C.1 独立序列的批处理

对于已知的训练序列或提示，编码器跨位置并行计算特征与记忆投影，解码器随后按位置顺序处理。独立序列的下一次解码器更新可以组成同一批次（图 10）。每条序列分别维护自己的编码器前缀、SWA 缓存、递归输出和位置。

推理时，每一步包含编码器更新、记忆追加、融合、解码器计算和读出。将请求组成批次可增大矩阵运算规模，并复用权重。小批量和不均匀的序列长度可能限制利用率。

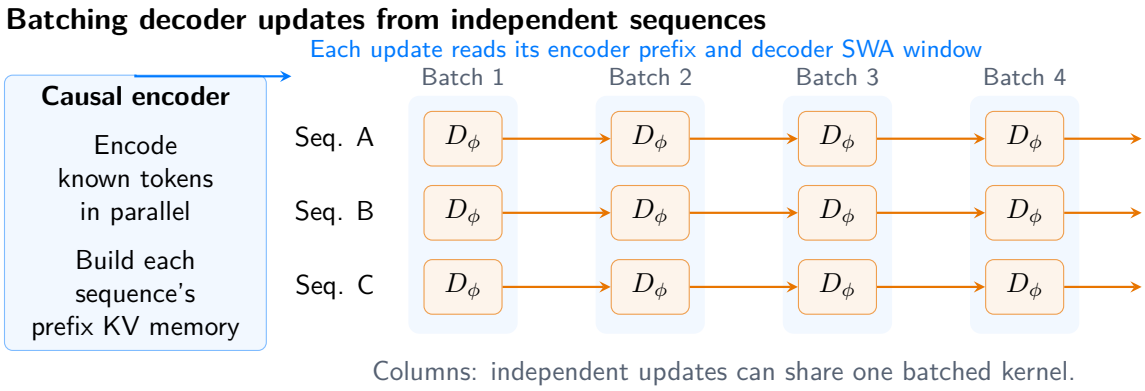


图 10 跨序列批量执行解码器更新。每列将独立更新组成一个批次，每行按 token 顺序展示一条序列。每次更新读取自己的编码器前缀和解码器 SWA 窗口。已知 token 可以在重放前并行编码，生成的 token 则在产生后编码。

C.2 内存访问与参数复用

当前缀和参数不变时，编码器 KV 保持不变。解码器各层复用该记忆，并将自己的 KV 追加到各自的 SWA 缓存中。较少的记忆组可减少 KV 存储和投影计算，较多的组则允许各层使用不同的变换。编码器与解码器共享权重也可能有助于让权重常驻加速器。

归一化、门控、状态投影和残差相加可以融合为一个内核。

C.3 训练内存与数值一致性

激活检查点通过在随时间反向传播（BPTT）时重算激活，减少需要保存的激活量。

Zhang et al. (2026) 指出，实际执行的策略不仅取决于权重，还取决于精度、缓存构造、归约运算和采样变换。采样器与训练器须对齐位置约定、随机行为和缓存内容，并检查不同执行模式下的数值是否一致。

D 训练目标

D.1 自回归预训练

基础目标是全序列的下一 token 预测：

$$\mathcal{L}_{\mathrm{PT}}(\Theta) = -\mathbb{E}_{x_{1:S}}\left[\frac{1}{S-1}\sum_{t=1}^{S-1}\log p_\Theta(x_{t+1} \mid x_{1:t})\right]. \tag{D.1}$$

初始化 $H_0 = (s_\star, \varnothing)$，计算因果编码器特征，并在位置 $1, \ldots, S-1$ 上更新解码器。每个非 BOS 目标都计入损失；编码器、记忆投影、融合和解码器联合训练。开始一篇独立文档时，重置解码器与编码器状态及位置，并禁止跨文档边界的注意力。分段训练时，应明确初始上下文与状态：丢弃较早的状态会改变似然所依赖的历史。

D.2 监督微调

对助手目标令 $m_{t+1} = 1$，对用户、系统、工具或填充目标令其为 0。对于至少包含一个选中目标的样本，优化

$$\mathcal{L}_{\mathrm{SFT}}(\Theta) = -\mathbb{E}_x\left[\frac{1}{\sum_{t=1}^{S-1} m_{t+1}}\sum_{\substack{1\le t<S\\ m_{t+1}=1}}\log p_\Theta(x_{t+1} \mid x_{1:t})\right]. \tag{D.2}$$

掩码只选择损失项，用户、系统和工具 token 仍更新解码器状态。助手损失的梯度会穿过这些更新，包括编码器记忆与解码器 KV。助手回合开始时，模型从已有状态继续。

D.3 策略梯度与重放

令 $c = x_{1:T}$ 为提示，$y = (y_1, \ldots, y_N)$ 为采样回复，其中 $y_i = x_{T+i}$。策略为

$$\pi_\Theta(y \mid c) = \prod_{i=1}^{N} p_\Theta(y_i \mid c, y_{<i}). \tag{D.3}$$

对于不依赖 Θ 的序列奖励 $R(c, y)$，定义 $J(\Theta) = \mathbb{E}_{c,\, y\sim\pi_\Theta}[R(c, y)]$。其同策略得分函数梯度为

$$\nabla_\Theta J = \mathbb{E}_{c,\, y\sim\pi_\Theta}\left[(R(c, y) - b(c))\sum_{i=1}^{N}\nabla_\Theta \log p_\Theta(y_i \mid c, y_{<i})\right], \tag{D.4}$$

其中 $b(c)$ 是不依赖回复的基线，在计算策略梯度时视为常数。重放时固定已采样的 token，梯度沿计算这些 token 对数概率的递归路径传播。

图 11 展示重放流程。采样器记录每个动作在实际采样分布 μ 下的对数概率。训练器从初始状态出发，在当前参数 Θ 下处理完整提示和已采样的回复前缀，重建编码器特征、递归输出及解码器 SWA 缓存。由所得动作概率构造比率

$$r_i(\Theta) = \exp\left(\log p_\Theta(y_i \mid c, y_{<i}) - \log \mu(y_i \mid c, y_{<i})\right). \tag{D.5}$$

此处目标策略为原始模型策略 p_Θ。行为概率必须包含采样器使用的温度缩放、截断和重新归一化。精确重要性采样要求 $p_\Theta(\cdot \mid h) \ll \mu(\cdot \mid h)$：在每个相关历史下，目标概率为正的动作，其行为概率

也必须为正。Top-k 或 top-p 采样通常会排除未经截断的 softmax 目标中概率为正的动作。如果目标策略本身也使用采样变换，应在目标函数与比率中一致应用该变换。

为保留前向概率及其对参数的导数，训练器需要对递归计算或其等价实现求导 (Zhang et al., 2026)。已记录的行为对数概率在参数更新后保持不变，因为它们描述的是采样这些动作时的策略。

对于固定提示和序列奖励，精确的轨迹重要性采样使用 $\prod_i r_i$，并需满足支持集与可积性条件。即使重放概率正确，逐 token 裁剪等替代损失仍可能引入偏差。

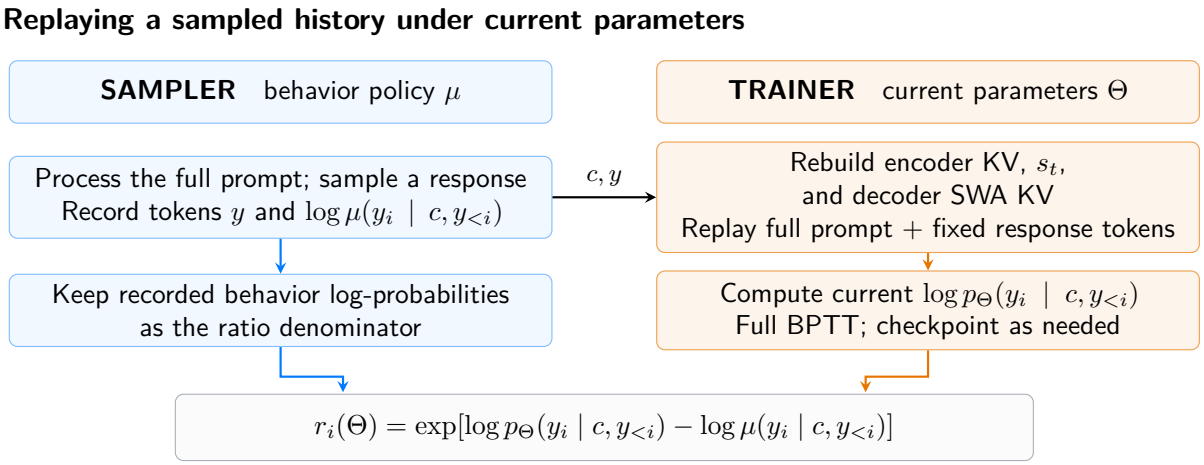


图 11　在当前参数下重放采样历史。采样器记录行为概率。训练器重建完整前缀，包括递归输出和解码器 SWA 缓存，以评估当前策略。比率比较当前动作概率与记录的行为概率。

E　多轮推理与缓存复用

对话构成一条从 BOS 开始的 token 历史。用户消息、工具结果、角色分隔符和助手 token 都执行编码器与解码器更新。例如，工具返回时，先结合缓存前缀编码其输出，再将这些 token 依次送入解码器，随后恢复生成。仅在开始新的独立序列或明确执行上下文重置时重置状态。

若需从保存的前缀继续，应保留编码器缓存 C_t^E、交叉注意力记忆 $M_{\le t}$、解码器状态 $H_t = (s_t, C_t^D)$、token 与位置、SWA 窗口约定以及模型版本。权重固定时，无论提示在哪个位置结束，这些值都可复用。若要复现相同的采样输出，还需要保存采样器的随机状态。如果只保存了文本，应重放文本来重建状态。

编辑或截断较早的 token 会使后续缓存状态失效。应从编辑位置之前的有效检查点重新计算；改写早期 token 的聊天模板也遵循同一规则。在多轮 RL 中，用户和工具 token 更新状态并传递梯度，但不乘动作重要性比率，因为这些 token 并非由策略采样。

F 执行流程

完整的解码器状态为 $H_t = (s_t, C_t^D)$，初始值为 $H_0 = (s_\star, \varnothing)$。继续编码所需的状态与编码器生成的交叉注意力记忆分别维护。所有调度均使用式 (2.20) 中的模块顺序和窗口约定。

F.1 递归提示预填充

1. 以 $x_1 = \text{BOS}$ 和 $H_0 = (s_\star, \varnothing)$ 开始一条独立序列，初始化编码器缓存与位置。
2. 使用因果掩码并行编码 $x_{1:T}$，构建编码器 KV 记忆。
3. 对 $t = 1, \ldots, T$，计算 $H_t = D_\phi(\text{Merge}(e_t, s_{t-1}); M_{\le t}, C_{t-1}^D, t)$。各解码器层读取当前位置的 KV 及保留的 SWA 历史。两种注意力掩码均排除 t 之后的位置。
4. 从 s_T 预测第一个回复 token。为下一次更新保留 H_T、编码器缓存、记忆和位置元数据。

F.2 生成与新的外部输入

从 s_t 预测的分布中采样 x_{t+1}，随后更新编码器缓存、记忆和完整解码器状态 H_{t+1}。所得状态用于预测 x_{t+2}。每个被处理的 token 恰好执行一次递归更新，并在每个解码器 SWA 层插入一次 KV。

用户或工具提供的多个 token 可在已有前缀条件下，使用因果掩码批量编码。解码器从保存的完整状态出发，依次处理这些 token，同时更新 s_t 和 C_t^D。

若生成达到长度上限时，最后一个输出 token 尚未被模型处理，应单独记录它。缓存只表示已处理的前缀；恢复时，先处理一次待处理 token，再继续处理后续 token。

F.3 预训练与监督微调

1. 用因果编码器计算 $e_{1:S-1}$ 并构建记忆。
2. 初始化 $H_0 = (s_\star, \varnothing)$，展开位置 $1, \ldots, S-1$ 的全部计算，包括各层 SWA 缓存更新。
3. 预训练累积所有有效的下一 token 损失，SFT 仅累积助手目标的损失。所有上下文 token 都执行状态更新。
4. 按每个样本选中的目标数量归一化，再按式 (D.1) 和式 (D.2) 对样本求平均。对完整计算反向传播，并按需使用检查点。

没有选中目标的样本不计入损失均值。若改为按 token 求平均，选中目标较多的样本会得到更高权重。

打包在一起的独立序列需要各自的递归状态、SWA 缓存、编码器缓存、位置和注意力掩码。即使损失被掩码屏蔽，两种注意力机制也必须限制在各自文档边界内。

F.4 当前策略的 RL 重放

1. 读取 rollout 的 token 历史、动作掩码、实际行为对数概率与采样元数据。行为概率包含全部采样变换和重新归一化。
2. 在前向重放和反向传播期间固定当前参数值。采用目标策略的位置、SWA 和执行约定，并启用参数梯度。
3. 重算已知历史的编码器表示。从 $H_0 = (s_\star, \varnothing)$ 开始，经过全部提示 token，重建递归输出与每个解码器 SWA 缓存。

4. 按顺序重放后续 token。在处理每个已采样动作之前，从前一状态读取该动作在当前策略下的对数概率。若 EOS 被采样，也应计入。处理夹在动作之间、后续预测所需的全部外部 token。
5. 根据动作对数概率、奖励或优势，以及所需的行为比率，构造选定的 RL 损失。动作因子仅作用于策略采样的 token。
6. 反向传播并更新参数；在下一次精确的当前策略重放前，使依赖参数的编码器和解码器缓存失效。

动作掩码选择损失项，同时保留经过用户与工具 token 的梯度。在 `no_grad` 下重放提示，或分离提示 KV，可保持前向值不变，但会丢失完整 BPTT 所需的提示状态梯度。

确定性的前向计算（例如关闭 dropout）可为每个动作给出可复现的概率。对于随机策略，应说明动作概率是以内部随机性为条件，还是对该随机性取平均。每次重放只对应一次随机实现；边缘概率需要对该随机性求平均。

G 因果性与梯度路径

命题 G.1 (因果性)**.** 在确定性执行下，若编码器是因果的，编码器记忆限制为当前前缀，且解码器 SWA 是因果的，则 $H_t = (s_t, C_t^D)$ 仅依赖 $x_{1:t}$ 和参数（包括 $s_\star$）。

证明. 编码器的因果性意味着各 e_j 仅依赖 $x_{1:j}$，因此 $M_{\le t}$ 仅依赖 $x_{1:t}$。初始状态不包含未来 token 的信息。假设 H_{t-1} 仅依赖 $x_{1:t-1}$。下一次更新使用该状态、e_t 和 $M_{\le t}$。在每个解码器层，当前位置的 KV 由因果可用的层输入形成，SWA 不读取大于 t 的位置。追加当前 KV 和淘汰旧条目不会引入未来信息。因此，s_t 和 C_t^D 均仅依赖 $x_{1:t}$，归纳完成。 □

固定教师强制得到的编码器特征，将解码器缓存表示为固定槽位，并在尚未填满时使用有效性掩码，定义

$$\mathcal{J}_t = \frac{\partial H_t}{\partial H_{t-1}}. \tag{G.1}$$

则

$$\frac{\partial H_t}{\partial H_j} = \mathcal{J}_t \mathcal{J}_{t-1} \cdots \mathcal{J}_{j+1}, \qquad j < t, \tag{G.2}$$

其中

$$\mathcal{J}_t = \begin{pmatrix} \dfrac{\partial s_t}{\partial s_{t-1}} & \dfrac{\partial s_t}{\partial C_{t-1}^D} \\ \dfrac{\partial C_t^D}{\partial s_{t-1}} & \dfrac{\partial C_t^D}{\partial C_{t-1}^D} \end{pmatrix}. \tag{G.3}$$

这个分块雅可比矩阵同时包含经过递归输出与解码器 KV 的路径。

编码器记忆保留过去编码器输出的信息；解码器 SWA 则可直接访问近期的解码器投影。已淘汰的解码器条目仍可能通过此前读取过它们的状态或激活影响后续计算。

完整的参数导数还包含编码器特征、记忆、各次状态转移和解码器 KV 投影对参数的依赖。令 B_T^E 表示继续生成回复所需的全部编码器侧边界张量，包括继续编码所需的 KV 和编码器生成的交叉注意力记忆。对于回复损失 $\ell(\Theta, H_T(\Theta), B_T^E(\Theta))$，链式法则给出

$$\frac{d\ell}{d\Theta} = \frac{\partial \ell}{\partial \Theta} + \frac{\partial \ell}{\partial H_T}\frac{\partial H_T}{\partial \Theta} + \frac{\partial \ell}{\partial B_T^E}\frac{\partial B_T^E}{\partial \Theta}. \tag{G.4}$$

第一项将边界张量视为固定值，其余项计入这些张量对前缀计算的依赖。特别地，

$$\frac{\partial \ell}{\partial H_T}\frac{\partial H_T}{\partial \Theta} = \frac{\partial \ell}{\partial s_T}\frac{\partial s_T}{\partial \Theta} + \frac{\partial \ell}{\partial C_T^D}\frac{\partial C_T^D}{\partial \Theta}. \tag{G.5}$$

即使前向概率不变，分离 s_T、解码器 KV 或编码器边界张量，也会从完整梯度中移除相应项。

H 缓存状态何时可以复用

当保存的前缀与当前参数、已处理 token、位置、初始化、SWA 窗口及策略执行设置一致时，可以用它替代前向重放。需要保存递归输出、每个解码器 SWA 缓存、继续编码所需的状态、编码器记忆及相关元数据。保存的前缀可以从序列起点开始，也可以来自当前参数和执行设置下的精确检查点。

已分离梯度的缓存可以保留正确数值，却缺少训练所需的计算图。完整 BPTT 需要保留或重算前缀计算图，使梯度能够到达每个依赖参数的边界张量。

一次更新内的激活检查点以相同参数和随机状态重算激活。可变缓存的实现必须在重算时恢复保存的内容，并保留反向传播所需的激活。

截断 BPTT 则保留数值，同时切断选定的梯度依赖。完整解码器状态的分离为

$$\widetilde{H}_t = (\mathrm{stopgrad}(s_t), \mathrm{stopgrad}(C_t^D)). \tag{H.1}$$

仅分离 s_t 会留下可能经过解码器 KV 的路径；仅分离解码器 KV 则会留下经过递归输出的路径。编码器侧边界张量还可能提供跨越同一边界的其他路径。

SWA 淘汰使旧 KV 不再进入未来的注意力窗口。完整 BPTT 仍会对之前使用过这些条目的计算求导，因此训练所需的激活存储可能超过推理缓存本身。

权重更新后，应在下一次精确训练计算之前，重算前缀数值及其梯度图。

I RLT-2：分块并行的隐藏状态反馈

第 2.7 节定义了 RLT-2 的融合与边界更新，图 2 展示单个 chunk；本附录给出其掩码、训练与预填充流程、增量生成和成本。第 3 节与附录 J.10 报告相关实验。

I.1 分块算子与因果掩码

固定 chunk 大小 $B \geq 1$，将 BOS 计为位置一，并以每条独立序列的起点为边界基准。对于长度为 T 的已知前缀，定义

$$K = \lceil T/B \rceil, \qquad a_k = (k-1)B + 1, \qquad b_k = \min(kB, T), \qquad I_k = \{a_k, \ldots, b_k\}. \tag{I.1}$$

解码器缓存初始为空；式 (2.22) 得到的融合输入 $z^0_{I_k}$ 依次经过分块算子和输出头：

$$y_{I_k}, C^D_{b_k} = D^{\mathrm{chunk}}_\phi(z^0_{I_k}; M, C^D_{a_k-1}, I_k), \tag{I.2}$$

$$p_\Theta(x_{t+1} \mid x_{1:t}) = \mathrm{softmax}\big(W_o\,\mathrm{RMSNorm}_o(y_t)\big)_{x_{t+1}}, \qquad t \in I_k. \tag{I.3}$$

分块算子逐层应用式 (2.20)，每层同时处理 I_k 内所有已知位置。查询 t 仅访问 $\max(1, t-W+1) \leq j \leq t$ 的解码器键，包括之前 chunk 保留的键，并且只访问 $j \leq t$ 的编码器记忆。SWA 和编码器位置均不在 chunk 边界重置，chunk 大小也不必等于 SWA 窗口。

每层先由该层输入投影所有位置的 KV，再执行带掩码的注意力，因此因果 SWA 可并行处理这些位置。最后一个位置的输出将当前 chunk 的因果计算传给下一 chunk，无需池化或额外的摘要 token。即使整条序列只占一个 chunk，仍存在可学习初始状态的融合；RLT-0 则移除了该融合及其参数。

I.2 训练与预填充算法

教师强制训练中，$x_{1:T}$ 表示已处理的输入 token，在有后续目标的位置计算下一 token 损失。提示预填充和已知历史的策略重放使用同一前向过程：

1. 对已知 token 做因果编码并构建编码器 KV 记忆。初始化 $h = s_\star$、空解码器 SWA 缓存和全局位置一。
2. 按顺序处理 chunk。每个 chunk 只计算一次 $\mathrm{RMSNorm}_s(h)$ 和 $W_s\,\mathrm{RMSNorm}_s(h)$，然后广播到各位置的门控融合。
3. 在每个解码器层中，并行投影 chunk 的 Q/K/V，结合历史和当前 chunk 的 KV 执行因果 SWA，然后执行前缀记忆注意力与 FFN。每层保留最后 $W-1$ 个解码器 KV 条目用于继续计算。
4. 在每个位置计算 logits 和所选的下一 token 损失。完整 chunk 结束后，将 h 更新为其最后一个位置的输出；末尾未完成的 chunk 不更新 h。
5. 训练时，累积序列损失并通过全部 chunk 反向传播，再更新参数。预填充时，保存缓存、h、已处理 token 数，以及最后一个位置的输出或下一 token 的 logits。

完整 BPTT 通过边界隐藏状态、跨 chunk 的解码器 KV 和编码器记忆传播梯度。分块改变前向依赖；是否截断梯度、何时更新优化器参数，需要分别设定。RL 重放使用与采样相同的 B、边界基准、掩码和位置，并在当前参数下重建状态后计算动作概率。

I.3 增量生成与未完成的 chunk

设已处理 token 数为 t，当前 chunk 偏移为 $r = t \bmod B$。保存的反馈状态为 $h_{\lfloor t/B \rfloor}$，即使提示结束在 chunk 中间也如此。处理下一个已知或采样 token（位置 $j = t+1$）时：

1. 更新因果编码器并追加其记忆 KV。将 e_j 与保存的边界状态融合；该状态在当前 chunk 内保持不变。
2. 使用因果 SWA 缓存和前缀记忆执行一次解码器更新，得到 y_j、更新后的 KV 缓存和下一 token 的 logits。
3. 若 $j \bmod B = 0$，则以 y_j 更新反馈状态；否则保留原值。将已处理 token 数更新为 j。

最新的 y_j 始终用于预测下一个 token，无论它是否更新反馈状态。例如 $B = 4$、提示长六个 token 时，位置 5–8 都与 $h_1 = y_4$ 融合；预填充处理位置 5、6 不会用 y_6 替换该状态。y_6 预测 token 7；只有处理 token 8 后，y_8 才成为 h_2。已知的后续 token 可并行处理到下一个固定边界，再使用新状态继续。填充、消息边界及提示与回复的分界，都不会触发未完成 chunk 的反馈更新。打包的独立序列分别维护边界基准和状态。

固定权重并采用精确算术时，批量和增量执行在数学算子和随机行为一致的条件下等价。随机操作须关闭，或为对应操作分配相同的随机数。此时，每个位置读取相同的编码器前缀、前序边界状态及各层因果 KV。先对 chunk 内各层、再对 chunk 归纳，可得两种执行方式的输出和边界状态相同。改变 B 或边界基准会改变模型计算，使保存的状态无法直接用于后续计算。

分块预填充与逐 token 解码可能使用不同的矩阵和注意力内核及浮点归约顺序，即使执行是确定性的，也可能产生不同的 logits 和边界状态。对于 RL，应将分块重放与逐 token 参考实现比较；后者复现 rollout 的编码器、解码器和输出头计算，包括精度、批次布局、内核选择及随机设置。

I.4 训练与推理效率

表 1 假设编码器与解码器深度、宽度、窗口和记忆组数相同，且 $L_D \geq 1$。表中的解码器模块阶段数表示逐层调度中的串行依赖，不包含注意力核内部归约、通信和硬件相关延迟。

固定网络尺寸并采用稠密编码器与记忆注意力时，公共前向算术复杂度为

$$O\big((L_E + L_D)Td^2 + (L_E + L_D)T^2d + GTd^2 + L_DT\min(W, T)d\big). \tag{I.4}$$

RLT-1 额外需要 $O(Td^2)$ 的融合计算。RLT-2 的 T 次位置相关门控也需要 $O(Td^2)$ 融合计算，但每个 chunk 内可复用状态归一化和投影。它使用 K 个串行解码批次，每批至多 B 个位置，可增大矩阵计算规模并复用权重。总解码算术量仍覆盖全部 T 个 token，完整 BPTT 仍需保存或重算所有 chunk 的激活。

在上下文长度为 t 的生成阶段，各变体每个 token 都需运行编码器与解码器并读取可用编码器记忆。RLT-2 在下一边界前可以复用反馈投影；位置相关门控、解码器模块和 KV 写入仍逐 token 执行。公共推理缓存规模为

$$O\big((L_E + G)t\, d_{\mathrm{KV}} + L_D \min(t, W - 1)d^D_{\mathrm{KV}}\big). \tag{I.5}$$

RLT-1 和 RLT-2 额外保存 $O(d)$ 反馈状态；RLT-2 还保存 chunk 偏移，并可缓存 $O(d)$ 的归一化和投影状态。分块预填充需要临时激活与当前 chunk 的 KV。

受控基准可扫描 $B \in \{1, 2, 4, 8, 16, 32\}$ 并加入 RLT-0，固定数据、网络尺寸、精度、优化器、批量大小和注意力内核。测量内容包括任务准确率、训练吞吐量、不同提示长度的预填充延迟、不同缓存上下文长度的每 token 生成时间，以及设备峰值内存。

$B = 1$ 检查与 RLT-1 的等价性，扰动未来 token 检查因果性。在参数不变时，完整 chunk、未完成 chunk 和逐 token 执行之间 logits、所选动作对数概率及边界状态的最大差异和均方根差异，可将执行差异与策略更新的影响区分开。可微参考实现还应比较完整 BPTT 的参数梯度，包括经过边界状态、解码器 KV 和编码器记忆的路径。

J 实验细节

J.1 深度扩展实验协议与终点结果

本批共有 48 次完整运行：两个任务、四种深度配对、两个模型家族，以及三个初始化种子（42、43、44）。冻结代码基点为 `d1a4516b`；完整实现摘要、配置、检查点哈希及各种子分数保存在 `references/depth-sweep-seeds42-44/`。运行环境来源和复现说明位于 `data/depth-extension/`。全部运行使用 Apple M4（macOS arm64）、Python 3.14.7、PyTorch 2.12.0、CPU FP32 和四个训练线程。这与前述层数分配实验的 Linux/Intel Xeon CPU 环境和冻结实现不同。

各 RLT-1 的编码器与解码器权重不共享，使用一个共享编码器记忆 KV 组、大小为八的 SWA 窗口，以及 0.1 的反馈系数。两类模型均使用宽度 512、四个注意力头、FFN 宽度 1,365 和宽度 μP 参数化。全局批量为 512，微批量为 32，累积 16 次，共训练 2,000 个优化器步，即每次运行呈现 1,024,000 个训练样本。AdamW 动量系数为 $(0.9, 0.95)$，权重衰减为 0.1，梯度范数裁剪阈值为 1；前 200 步预热至基础学习率 10^{-4}，随后余弦衰减至 5×10^{-6}。训练、验证和测试数据种子分别为 20260914、20260915 和 20260916，只有模型初始化发生变化。

奇偶校验在长度 3–40 上使用最终标签监督；交换 S_5 在由恒等置换和十个对换组成的 32 次操作序列上训练，监督每个中间置换状态。每 100 步验证一次，每种 ID 长度使用 256 个样本（奇偶校验为 32、36、40，交换 S_5 为 32）。各次运行选择加权 ID 验证样本损失最低的检查点，完全同分时取最早一步。奇偶校验测试长度为 32、36、40、48、64、96、128、192 和 256；交换 S_5 为 32、48、64、96、128、192、256 和 500。同一任务与长度的全部深度、模型家族和种子共享 1,024 个留出样本。408 个各种子测试结果汇总为 136 个均值与样本 SD，分母为 $n - 1 = 2$；SD 表示固定数据上的初始化波动。终点表和正文图报告最终答案或状态准确率；数据包另保留前缀 token 和全序列准确率。

虽然配置 TBPTT128，但包含 BOS 的训练输入最长只有 41 或 33 个 token，因此训练梯度未被截断。评估设置 `tbptt_chunk_size=0`，递归状态连续传递。交换 S_5 的终点为 500 次操作，与层数分配实验的 512 次操作终点不同。早期参考数据包中两个种子 42 的 512 位奇偶校验补充测试缺少对应的种子 43/44 结果，因此未纳入本组汇总。

表 2 深度扩展模型参数量与最长测试长度准确率。数值为三个种子的均值 ± 样本 SD，单位为百分比。参数列为不重复计数的可训练参数量。

Model	Parameters	Parity (256)	S_5 swaps (500)
RLT-1 $4+4$	28,726,912	83.72 ± 28.19	37.08 ± 31.11
GPT 8	25,314,304	50.55 ± 1.58	0.72 ± 0.39
RLT-1 $6+6$	42,362,752	100.00 ± 0.00	76.46 ± 38.09
GPT 12	37,900,288	49.93 ± 1.86	0.75 ± 0.11
RLT-1 $8+8$	55,998,592	100.00 ± 0.00	92.29 ± 10.08
GPT 16	50,486,272	48.96 ± 1.59	0.55 ± 0.15
RLT-1 $10+10$	69,634,432	84.11 ± 27.51	99.02 ± 1.18
GPT 20	63,072,256	51.24 ± 0.69	0.75 ± 0.06

表 3 训练 2,000 步后的验证准确率（%）。所有条目均为种子 42、43 和 44 的均值 ± 样本 SD。加法使用教师强制答案 token，形式任务使用最终标签或状态。每次运行已使用 1,024,000 个训练样本。

Task	RLT-1 Layer 4 + 4	RLT-1 Layer 5 + 3	RLT-1 Layer 6 + 2	RLT-1 Layer 7 + 1	RLT-1 Layer 8 + 0	GPT Layer 8
Addition	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
Parity	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	98.83 ± 1.92	94.84 ± 3.43
Mod 5, no brackets	45.36 ± 46.43	94.18 ± 7.29	69.62 ± 33.01	70.01 ± 43.15	60.33 ± 35.70	64.02 ± 37.64
Mod 5, brackets	70.53 ± 10.75	74.35 ± 6.71	74.87 ± 3.72	74.78 ± 3.01	75.17 ± 6.78	73.87 ± 9.07
S_5, swaps	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	99.35 ± 0.81	99.61 ± 0.68	99.09 ± 0.23
S_5, standard	0.78 ± 0.68	2.47 ± 1.26	2.21 ± 1.48	2.08 ± 0.98	1.82 ± 1.19	0.52 ± 0.23

J.2 完整的六任务 2,000 步实验

本附录报告八层实验的全部 108 次运行：六种架构在六个任务上使用初始化种子 42、43 和 44，每次训练 2,000 步。内容包括训练长度范围内的验证曲线，以及各次运行分布内最佳检查点的长度泛化。

J.2.1 模型、任务与评估协议

模型、优化器设置、批量大小和执行方式与第 3 节相同；余弦调度在第 2,000 步降至 5×10^{-6}。每 100 步执行一次验证，每次运行使用 1,024,000 个训练样本。

- **加法（Addition）：**随机采样 1–8 位操作数，将数字按逆序序列化。256 个验证样本使用相同的位数范围。我们测量教师强制答案 token 准确率：预测时提供正确的前序 token，计分包含答案格式符号与 EOS，排除提示和填充位置。
- **奇偶校验（Parity）：**使用长度为 3–40 的二进制串训练，只监督最终的奇偶位。验证集汇总长度为 32、36 和 40 的样本，每种长度 256 个。
- **模运算（Mod-5）：**计算含 +、− 和 × 的表达式，并对结果取模五。无括号变体遵循乘法优先级，训练表达式长度为 3–39 的奇数，验证长度为 33、35 和 39。带括号变体使用表达式树，训练长度为 3–40，验证长度为 32、36 和 40。每种验证长度包含 256 个样本，仅监督最终值。
- S_5 **状态跟踪：**复合 32 个置换，在每次输入后监督累计乘积，遵循 Grazzi et al. (2024)。标准变体的输入覆盖全部 120 个置换；交换变体使用恒等置换和十个单次对换。每个变体有 256 条长度为 32 的验证序列。主要指标为最终状态准确率，前缀 token 准确率列于表 6。

对每次运行，先合并所设验证长度的计数，再跨初始化求平均。图 12 的每一步均包含全部三个种子，表 3 统一比较第 2,000 步。

J.2.2 奇偶校验：学习速度与初始化差异

第 500 步时，RLT-1 Layer $6+2$ 的验证准确率为 $99.44 \pm 0.98\%$，Transformer 为 $48.48 \pm 0.53\%$。RLT-1 Layer $4+4$、$5+3$ 和 $7+1$ 的均值约为 83%，SD 为 28–30 个百分点；$8+0$ 仍接近随机水平。到第 600 步，$6+2$ 的三个种子均已达到 100%。第 2,000 步时，$4+4$ 至 $7+1$ 均达到 $100 \pm 0\%$，$8+0$ 为 $98.83 \pm 1.92\%$，Transformer 为 $94.84 \pm 3.43\%$。图 7 比较早期与最终检查点。

J.2.3 带括号与无括号模运算

无括号模运算在第 2,000 步呈现较大的初始化差异。RLT-1 Layer 5+3 的均值最高，为 94.18±7.29%，Transformer 为 $64.02 \pm 37.64\%$。$4+4$ 的种子 42、43 和 44 分别得到 17.58%、19.53% 和 98.96%，

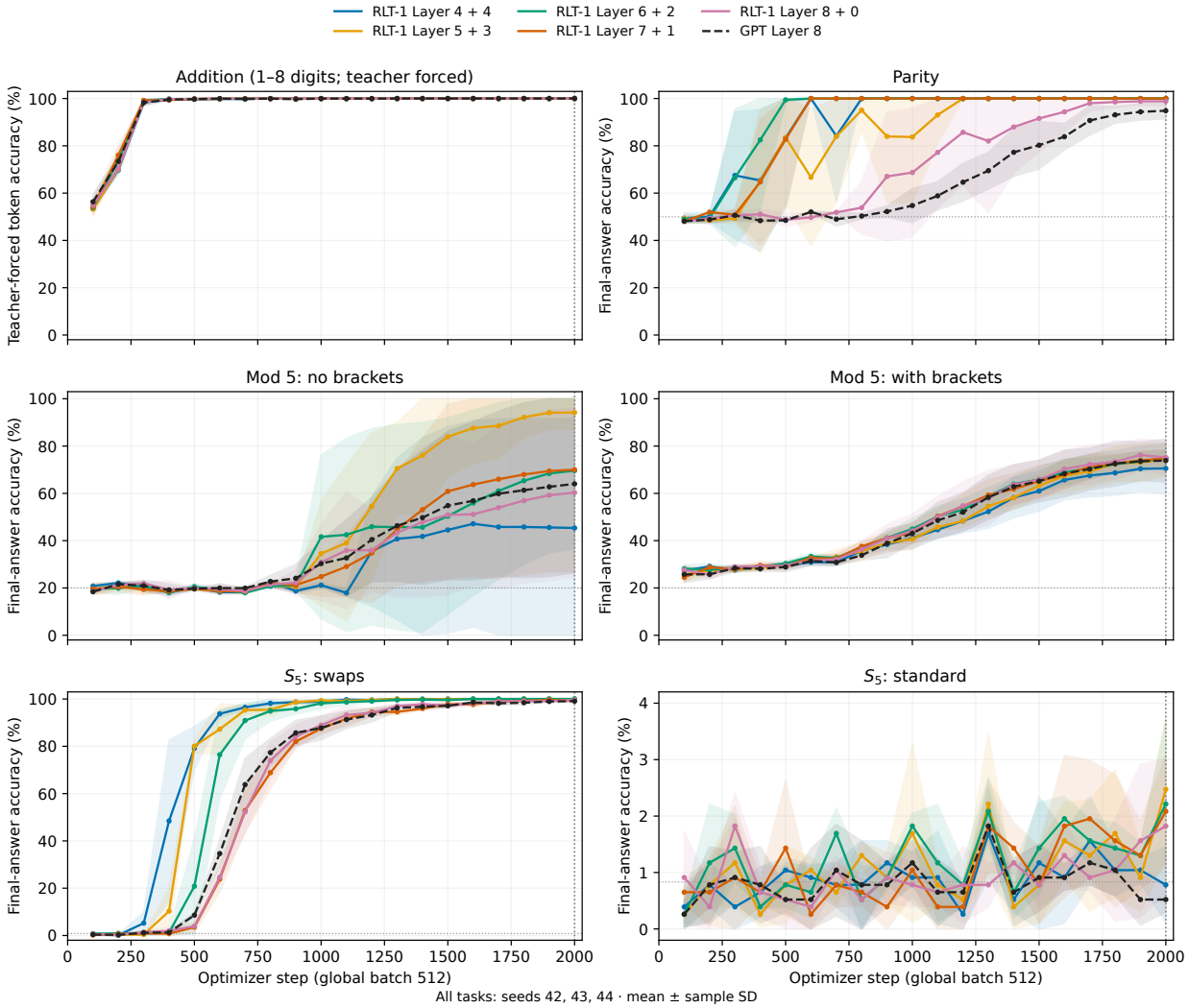


图 12　训练过程中的验证准确率，每个任务均使用三个种子。加法测量教师强制答案 token 准确率，其余任务测量最终标签或最终状态准确率。曲线与阴影表示种子 42、43 和 44 的均值 ± 样本 SD，每一步均为 $n = 3$；阴影带裁剪到准确率范围内。曲线未经平滑，均延伸至第 2,000 步。横向点线表示均匀预测的准确率，标准 S_5 使用较窄的纵轴范围。

均值为 $45.36 \pm 46.43\%$。在这一训练预算下，同一种架构可能学会任务，也可能停留在随机水平附近。

带括号表达式上的模型均值更接近：RLT-1 各配置为 70.53–75.17%，Transformer 为 $73.87 \pm 9.07\%$。两种生成器的运算符结构和标签分布不同。括号也占据位置，因此相同的 token 长度可能包含不同数量的算术运算。

J.2.4 加法与置换状态跟踪

第 2,000 步时，六种架构在混合 1–8 位加法验证集上均达到 $100 \pm 0\%$ 的教师强制 token 准确率。在交换 S_5 上，RLT-1 Layer $4+4$、$5+3$ 和 $6+2$ 的最终状态准确率为 $100 \pm 0\%$。$7+1$ 与 $8+0$ 分别为 $99.35 \pm 0.81\%$ 和 $99.61 \pm 0.68\%$，Transformer 为 $99.09 \pm 0.23\%$。

标准 S_5 的平均最终状态准确率为 0.52–2.47%，120 类均匀预测的参考值为 0.83%。平均前缀 token 准确率为 5.40–7.99%（表 6），这一指标包含较容易的早期状态。

J.2.5 长度泛化

检查点选择方式与第 3 节相同。同一任务的全部 18 个模型与种子组合在各长度上使用相同样本：加法为 256 对操作数，形式任务为 1,024 条序列。图 13 展示全部测试长度，表 4 汇总各任务最长测试长度的结果。附录 J.7 给出检查点步骤和补充指标。

加法与奇偶校验。 操作数超出 1–8 位训练范围后，教师强制加法准确率下降。每个操作数为九位时，RLT-1 Layer $7+1$ 为 $68.05 \pm 3.64\%$，Transformer 为 $58.61 \pm 2.44\%$。到 32 位时，各模型均值处于 14.89–16.84%。

即使训练长度上的预测全部正确，奇偶校验的长度泛化仍随层数分配而变化。在 256 位时，$5+3$ 和 $7+1$ 保持 $100 \pm 0\%$，$6+2$ 为 $84.05 \pm 27.63\%$，$4+4$ 为 $66.76 \pm 28.78\%$。$8+0$ 为 $68.91 \pm 27.39\%$，Transformer 为接近随机水平的 $50.07 \pm 1.63\%$。

模运算。 无括号表达式长度为 63 时，RLT-1 Layer $5+3$ 为 $66.96 \pm 40.04\%$，Transformer 为 $24.48 \pm 3.89\%$。长度为 127 时，$5+3$ 仍有 $39.65 \pm 19.81\%$，Transformer 为 $19.30 \pm 1.04\%$。到长度 255 时，各模型均值为 18.00–21.42%，接近 20% 的均匀预测参考值。中间长度上较大的 SD 与无括号模五验证曲线中的训练结果差异一致。带括号模五运算也随表达式变长而降低准确率。

置换状态跟踪。 交换 S_5 在 256 次操作时，RLT-1 Layer $4+4$ 的最终状态准确率为 $97.30 \pm 2.76\%$，Transformer 为 $0.85 \pm 0.30\%$。$5+3$ 和 $6+2$ 分别为 $92.71 \pm 1.87\%$ 和 $78.91 \pm 22.26\%$，$7+1$ 与 $8+0$ 接近均匀预测参考值。在 512 次操作时，$4+4$ 仍有 $55.70 \pm 25.78\%$ 的最终状态准确率和 $91.16 \pm 6.09\%$ 的前缀 token 准确率；Transformer 分别为 $0.85 \pm 0.30\%$ 和 $9.33 \pm 0.11\%$。标准 S_5 在该长度下仍然较低，最终状态准确率均值为 0.72–1.43%。

图 14 比较不同测试长度下的三种状态跟踪准确率。前缀 token 准确率对所有中间状态的预测正确性求平均；最终状态准确率只评价最后一个状态。全序列准确率要求每个前缀预测都正确。最终状态正确并不排除中间预测出错，而当错误出现在序列后段时，前缀 token 准确率仍可能较高。

J.2.6 训练稳定性与反馈对照

奇偶校验中，种子 42 的 RLT-1 Layer $5+3$ 从第 500 步的 100% 降至第 600 步的 48.05%，随后在第 700 步恢复至 100%。图 18 展示未经平滑的损失，附录 J 给出奇偶校验的各种子曲线。

附录 J.10 和 J.11 在每种配比下用 RLT-0 与 RLT-2 隔离反馈的作用。

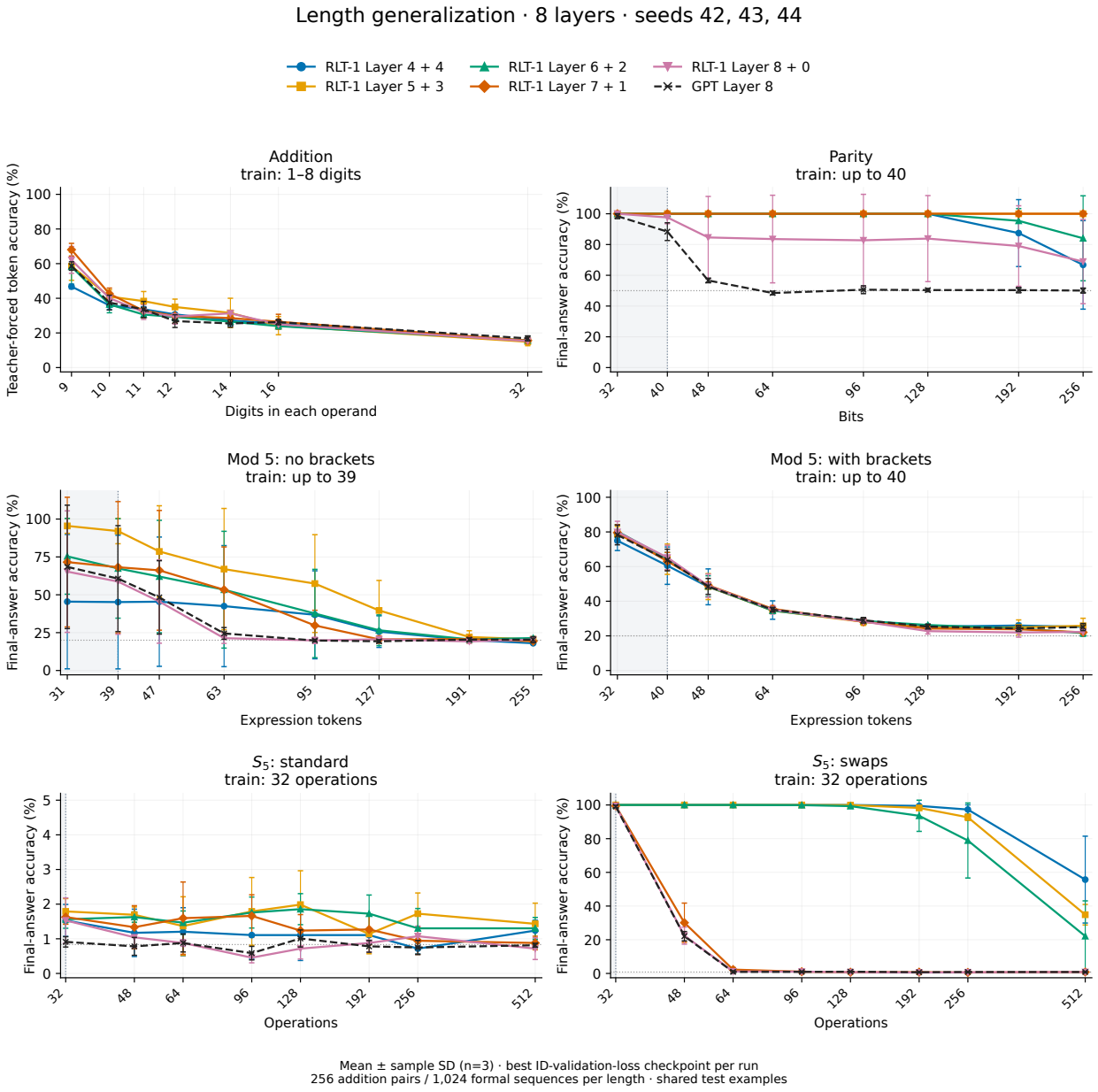


图 13　分布内最佳检查点的长度泛化，三个初始化种子。加法在每种操作数位数的 256 对样本上测量教师强制答案 token 准确率；形式任务在每个长度的 1,024 条序列上测量最终标签或最终状态准确率。点与误差棒表示种子 42、43 和 44 的均值 ± 样本 SD，所有模型使用相同测试样本。SD 误差棒可能超出准确率范围。灰色区域表示训练长度，横向点线表示均匀预测准确率。无括号模五运算使用实际奇数表达式长度，不包含 BOS 和末尾等号。

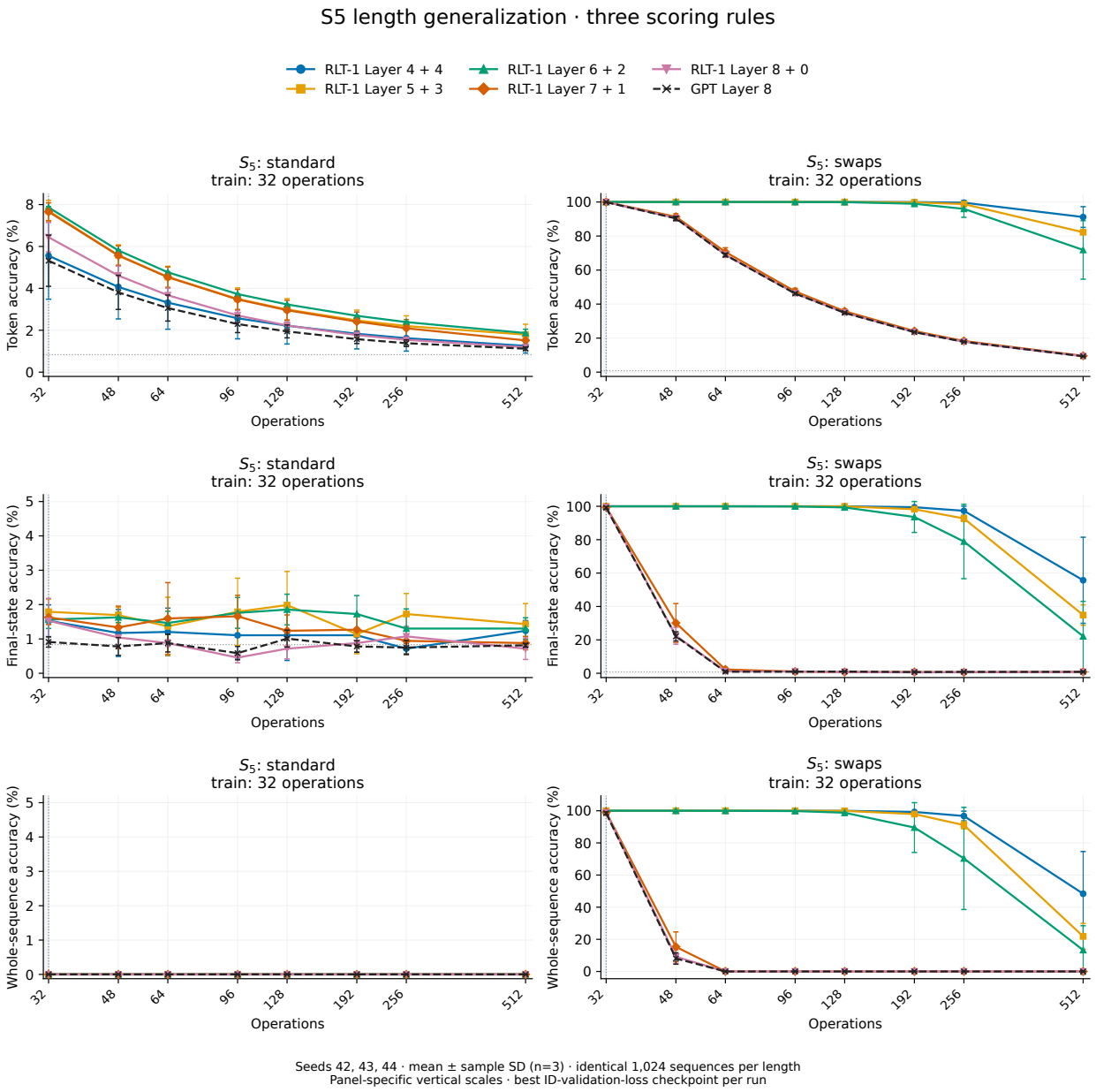


图 14　三种评分规则下的 S_5 长度泛化。各行依次为前缀 token、最终状态和全序列准确率，各列为标准与交换输入。每个模型与种子在每种长度下接收相同的 1,024 条序列。点和误差棒表示三个初始化种子的均值 ± 样本 SD。全序列成功要求所有前缀预测正确，各面板独立设置纵轴范围以展示标准 S_5 上较低的准确率。

表 4 最长测试长度上的准确率（%）。条目为三个初始化种子的均值 ± 样本 SD，采用图 13 的检查点与评分规则。加法报告教师强制 token 准确率，形式任务报告最终标签或最终状态准确率。

Task (test length)	RLT-1 Layer 4 + 4	RLT-1 Layer 5 + 3	RLT-1 Layer 6 + 2	RLT-1 Layer 7 + 1	RLT-1 Layer 8 + 0	GPT Layer 8
Addition (32)	15.30 ± 0.44	14.89 ± 2.27	15.74 ± 1.58	15.59 ± 1.57	15.31 ± 0.93	16.84 ± 1.45
Parity (256)	66.76 ± 28.78	100.00 ± 0.00	84.05 ± 27.63	100.00 ± 0.00	68.91 ± 27.39	50.07 ± 1.63
Mod 5, no brackets (255)	18.00 ± 0.39	20.57 ± 0.62	21.42 ± 0.49	19.34 ± 0.54	20.44 ± 0.91	20.35 ± 2.01
Mod 5, brackets (256)	25.20 ± 1.71	25.81 ± 4.34	21.58 ± 1.86	22.04 ± 1.72	22.30 ± 1.13	25.07 ± 2.05
S_5, standard (512)	1.24 ± 0.30	1.43 ± 0.60	1.30 ± 0.31	0.88 ± 0.20	0.72 ± 0.31	0.81 ± 0.06
S_5, swaps (512)	55.70 ± 25.78	34.86 ± 6.10	22.14 ± 20.95	0.85 ± 0.06	0.85 ± 0.20	0.85 ± 0.30

J.3 参数量

表 5 列出不重复计数的可训练参数量。每个 RLT-1 解码器模块同时具有 SWA 和编码器记忆交叉注意力；每个 Transformer 模块只有一个注意力子层。RLT-1 Layer $8+0$ 在每个 token 处应用可学习的状态投影和门控融合。

表 5 八层模型的不重复可训练参数量。同一模型在六个任务中的规模相同。

Model	Parameters (M)	Relative to GPT Layer 8
RLT-1 Layer $4+4$	28.73	1.135×
RLT-1 Layer $5+3$	28.20	1.114×
RLT-1 Layer $6+2$	27.68	1.093×
RLT-1 Layer $7+1$	27.15	1.073×
RLT-1 Layer $8+0$	26.10	1.031×
GPT Layer 8	25.31	1.000×

J.4 RLT-0 的实现

`rlt_parallel_tiny` 对照由 nanogptpro-dev PR #164 合并。它采用不共享权重的 $4+4$ 编码器与解码器分配、宽度 512、四个注意力头、FFN 宽度 1,365、词表容量 277、大小为八的 SWA 窗口、一个编码器记忆组，以及宽度 μP。它共有 27,938,944 个不重复参数，架构见第 2.7 节。训练采用完整反向传播，RLT-0 与 RLT-1 的检查点由各自的架构标识区分。

J.5 种子聚合与指标定义

令 $a_{m,s,t}$ 为模型 m、初始化种子 s 在训练第 t 步的验证准确率。对每个任务均报告

$$\bar{a}_{m,t} = \frac{1}{3} \sum_{s \in \{42,43,44\}} a_{m,s,t}, \qquad \mathrm{SD}_{m,t} = \sqrt{\frac{1}{2} \sum_{s \in \{42,43,44\}} (a_{m,s,t} - \bar{a}_{m,t})^2}. \tag{J.1}$$

我们在同一步聚合全部三个种子，不进行插值或缺失值填补。所有曲线延伸至第 2,000 步，表 3 统一使用该步。

奇偶校验和模五运算每个样本监督一个最终标签。先合并各次运行中三个大小相同的验证组，再跨种子求平均。S_5 监督每个前缀；最终状态准确率只统计每条序列的最后一个状态，前缀 token 准确率合并全部受监督位置。加法使用完整提示和正确的先前答案 token 进行教师强制。其分母包含数字、空格、括号与 EOS 等全部答案 token，排除提示与填充位置。加法损失是按样本归一化的教师强制交叉熵。

J.6 完整训练快照的复现

形式任务的训练、验证与测试数据种子分别为 20260914、20260915 和 20260916。加法使用数据种子 42。全部 108 次运行使用基于上游提交 `d1a4516` 的实现，均完成 2,000 步。训练日志于 2026 年 9 月 17 日冻结，清单记录精确采集时间和来源哈希。

目录 `data/experiments-depth8/` 中的 `manifest.json` 和 `configs.json` 记录运行标识、设置以及来源与语料指纹。目录 `scripts/` 中的 `plot_depth8.py` 根据 2,160 条验证记录和 216,000 条

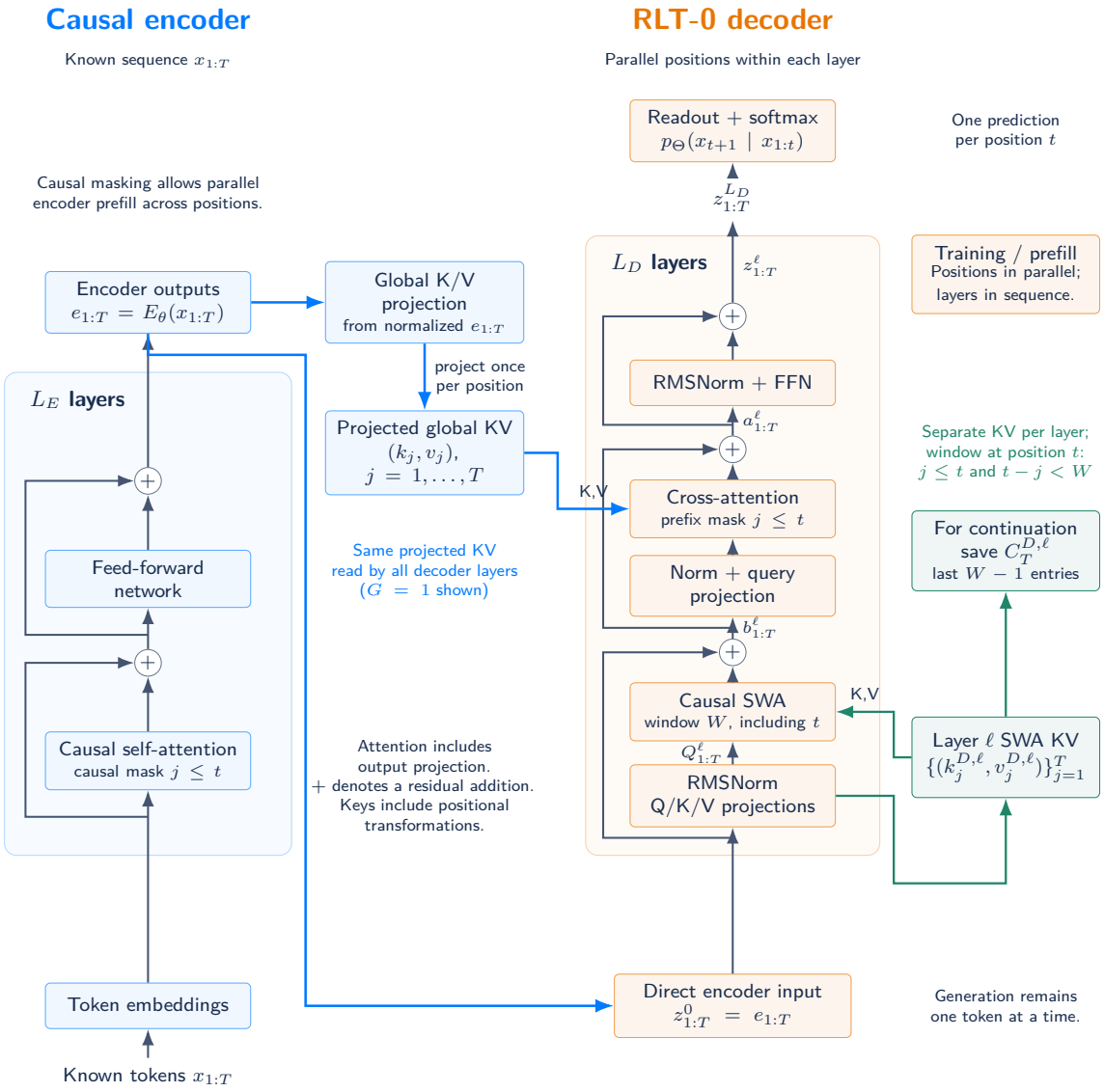


图 15　训练或提示预填充阶段的 RLT-0。解码器直接接收编码器输出，$z^0_{1:T} = e_{1:T}$，门控融合与最终隐藏状态反馈路径均被移除。各虚线框中的模块按标注深度重复。当 $G = 1$ 时，所有解码器层读取同一份已投影的全局 KV；查询位置 t 可读取编码器位置 $j \le t$。各解码器层构造自己的 SWA KV，并使用包含当前位置、大小为 W 的因果窗口。已知位置在每层内一起处理，各层则按深度顺序执行。增量生成保留编码器缓存、全局记忆和各层最近的 $W - 1$ 条 SWA 缓存，逐 token 进行。

表 6 固定 S_5 验证集在第 2,000 步的最终状态与前缀 token 准确率（%），报告种子 42、43 和 44 的均值 ± 样本 SD。

Task	Metric	RLT-1 Layer 4 + 4	RLT-1 Layer 5 + 3	RLT-1 Layer 6 + 2	RLT-1 Layer 7 + 1	RLT-1 Layer 8 + 0	GPT Layer 8
S_5, swaps	Final state	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	99.35 ± 0.81	99.61 ± 0.68	99.09 ± 0.23
S_5, swaps	Prefix token	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	99.96 ± 0.05	99.95 ± 0.06	99.91 ± 0.06
S_5, standard	Final state	0.78 ± 0.68	2.47 ± 1.26	2.21 ± 1.48	2.08 ± 0.98	1.82 ± 1.19	0.52 ± 0.23
S_5, standard	Prefix token	5.57 ± 2.11	7.82 ± 0.55	7.99 ± 0.29	7.67 ± 0.54	6.47 ± 0.65	5.40 ± 1.21

未经平滑的逐步训练记录重新生成训练图表。固定长度 token 准确率数据包含 90 次已完成的形式任务运行，共 1,800 条验证记录；同一目录中的 `plot_token_accuracy.py` 重新生成对应图形。

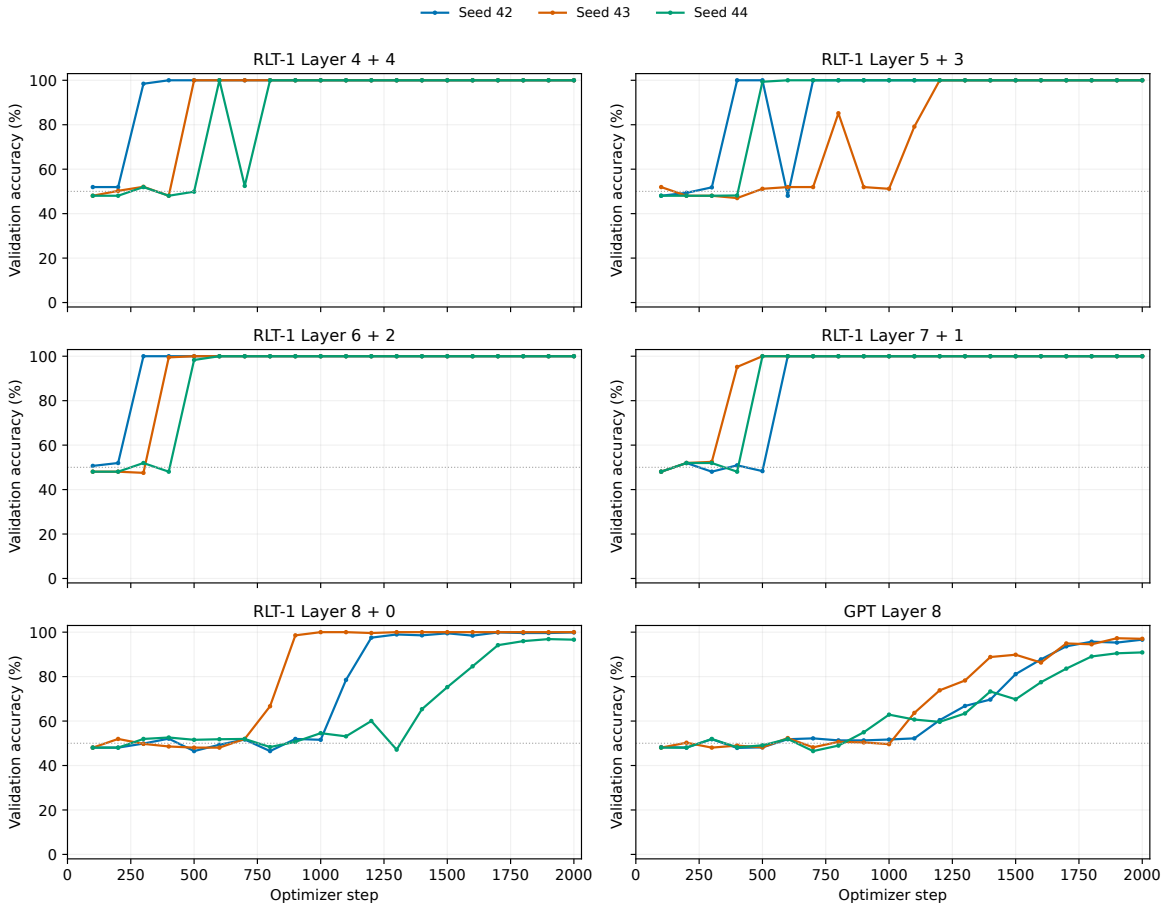


图 16 各初始化种子的奇偶校验准确率，均延伸至第 2,000 步。所有运行共享索引训练数据流与验证集；曲线显示暂时回退和学习速度差异。

J.7 长度泛化协议与补充指标

该实验使用全部 108 次已完成的运行，每种架构与任务均包含种子 42、43 和 44。每次运行按训练日志中最低的分布内验证样本损失选择检查点；形式任务先合并所设验证长度，损失相同时选择最早步骤。表 7 列出所选步骤，检查点与训练结果的哈希随数据保存。测试结果不参与检查点选择。

加法每个操作数的测试位数为 9、10、11、12、14、16 和 32。奇偶校验与带括号模五运算测试长度为 32、40、48、64、96、128、192 和 256；两种 S_5 任务使用 32、48、64、96、128、192、256 和 512。无括号表达式由操作数与二元运算符交替构成，因此每个请求的偶数长度 L 映射为实际表达式长度 $L-1$，即 31、39、47、63、95、127、191 和 255。形式任务长度不包含 BOS 和末尾等号；无括号模五运算加入这两个标记后有 $L+1$ 个输入 token。共有 47 个共享测试集和 846 个模型、种子与长度组合的评估。

加法在每个位数下使用 256 对互不重复的无序操作数，按种子 20260916 从原始测试划分采样。两个操作数均恰好具有指定的位数且无前导零，逆序数字序列化与训练一致。评估计算教师强制答案 token 准确率，不执行自回归生成。每个形式任务测试集由原任务生成器和测试种子 20260916 产生，包含 1,024 条序列。所有模型与初始化在同一任务及长度下使用相同样本。共享留出集沿用上一轮基准；新增的形式任务测试集排除重复样本和已有验证、测试样本。

检查所选检查点与留出集指纹后，我们复用 697 个模型、种子与长度组合的结果，并评估剩余 149 个组合。所有指标均从保存的逐样本预测或教师强制 token 计数重新计算。评估使用 CPU FP32、四个算子内线程和训练时记录的数值运行环境，检查点权重保持固定。新评估的批量为 32，复用的原生形式任务评估批量为四。批量检查中，RLT-1 Layer $4+4$ 对每种 S_5 任务的 32 条长度 512 序列，在两种批量下给出完全相同的 32,768 个状态预测。教师强制加法评估器也在 32 对共享的 32 位操作数上，复现了 RLT-1 Layer $4+4$ 与 GPT Layer 8 保存的逐样本 token 计数。

对每个测试长度，先计算各初始化的分数，再将上述均值与 SD 公式中的训练步骤替换为长度。误差棒表示共享测试样本上三个初始化之间的样本 SD。图 17 补充报告六任务 token 准确率；三种 S_5 评分规则的比较见附录 J.2.5（图 14）。

目录 `data/experiments-depth8-generalization/` 包含 846 条逐种子记录、282 个聚合点、检查点与数据集指纹，以及验证报告。脚本 `scripts/plot_generalization.py` 在生成图表前，检查种子和长度是否齐全、由计数计算的分数、样本 SD，以及检查点与训练快照的对应关系。

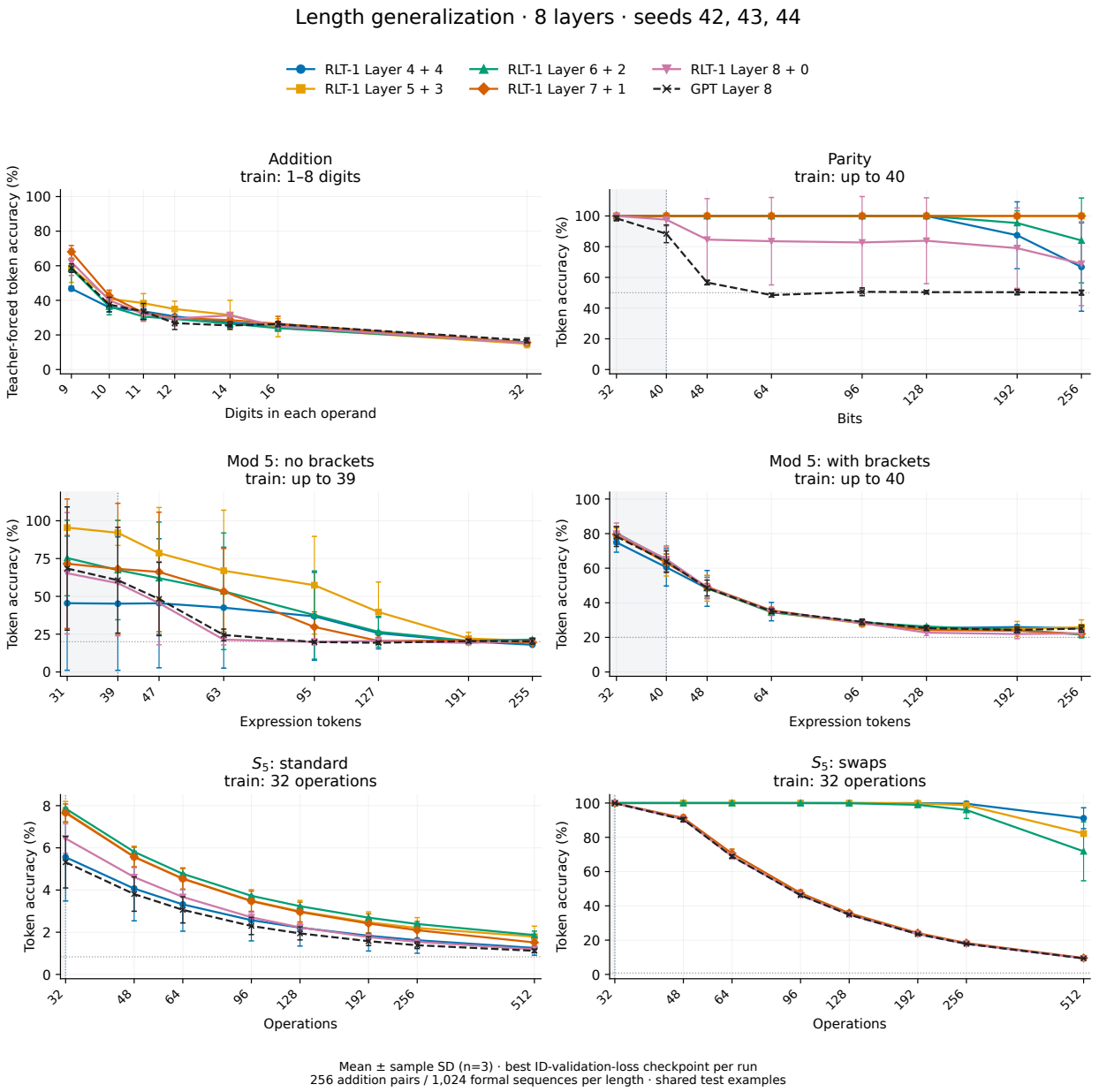


图 17　不同测试长度上的 token 准确率，三个初始化种子。检查点与共享样本同图 13。加法对答案 token 使用教师强制，S_5 对每个前缀状态评分，奇偶校验与模五运算各对一个最终标签评分。点和误差棒表示种子 42、43 和 44 的均值 ± 样本 SD，灰色区域表示训练长度。

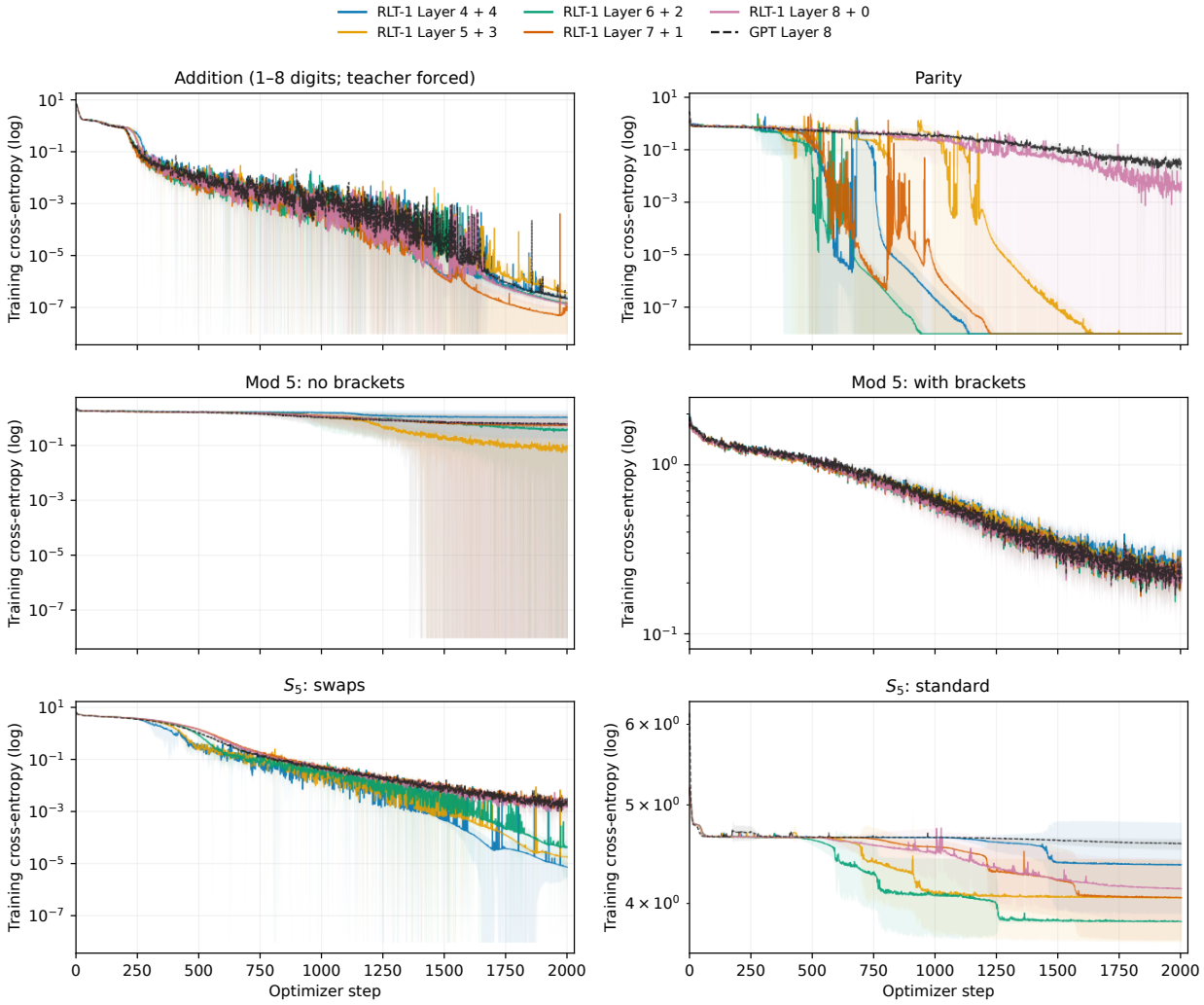


图 18 未经平滑的全局批量训练交叉熵，每个任务均展示三个种子的均值 ± 样本 SD。在对数轴上，低于 10^{-8} 的均值或阴影边界按 10^{-8} 显示。损失在优化器更新前计算，记录的梯度范数为裁剪前的值。各任务的监督方式不同，应在同一任务内比较损失。

表 7 按最低分布内验证损失选取的长度泛化检查点步数。每次运行均完成 2,000 步训练，分别列出种子 42、43 和 44。

Task	Seed	RLT-1 Layer 4 + 4	RLT-1 Layer 5 + 3	RLT-1 Layer 6 + 2	RLT-1 Layer 7 + 1	RLT-1 Layer 8 + 0	GPT Layer 8
Addition	42	2000	2000	2000	2000	2000	2000
	43	2000	2000	2000	2000	2000	2000
	44	2000	2000	2000	1900	2000	2000
Parity	42	900	1600	800	1500	2000	2000
	43	1100	2000	900	1500	1900	1900
	44	1400	1000	1200	1100	2000	2000
Mod 5, no brackets	42	800	2000	2000	2000	2000	400
	43	800	2000	2000	800	400	2000
	44	2000	2000	2000	2000	2000	1900
Mod 5, brackets	42	2000	2000	2000	2000	2000	1900
	43	2000	2000	2000	2000	2000	2000
	44	2000	2000	2000	2000	2000	2000
S_5, standard	42	1800	2000	2000	2000	2000	2000
	43	2000	2000	2000	2000	2000	2000
	44	2000	2000	1800	2000	2000	2000
S_5, swaps	42	2000	2000	2000	1900	2000	2000
	43	2000	2000	2000	2000	2000	1900
	44	2000	2000	2000	1900	2000	2000

J.8 长度为 32 和 33 时的 token 准确率曲线

图 19 比较奇偶校验、带括号模五运算和两种 S_5 任务在长度 32 时的 token 准确率，以及无括号模五运算在长度 33 时的 token 准确率。无括号表达式的数字与二元运算符交替出现，因此有效内容长度为奇数；33 是已配置验证长度中最接近 32 的值。快照包含 90 次运行的 1,800 个验证检查点：六种架构、五个形式任务、三个种子，每次均完成 2,000 步训练。长度为 32 时，S_5 统计 256 条序列中的 8,192 个前缀状态；奇偶校验与模五运算每条序列评分一个最终标签。

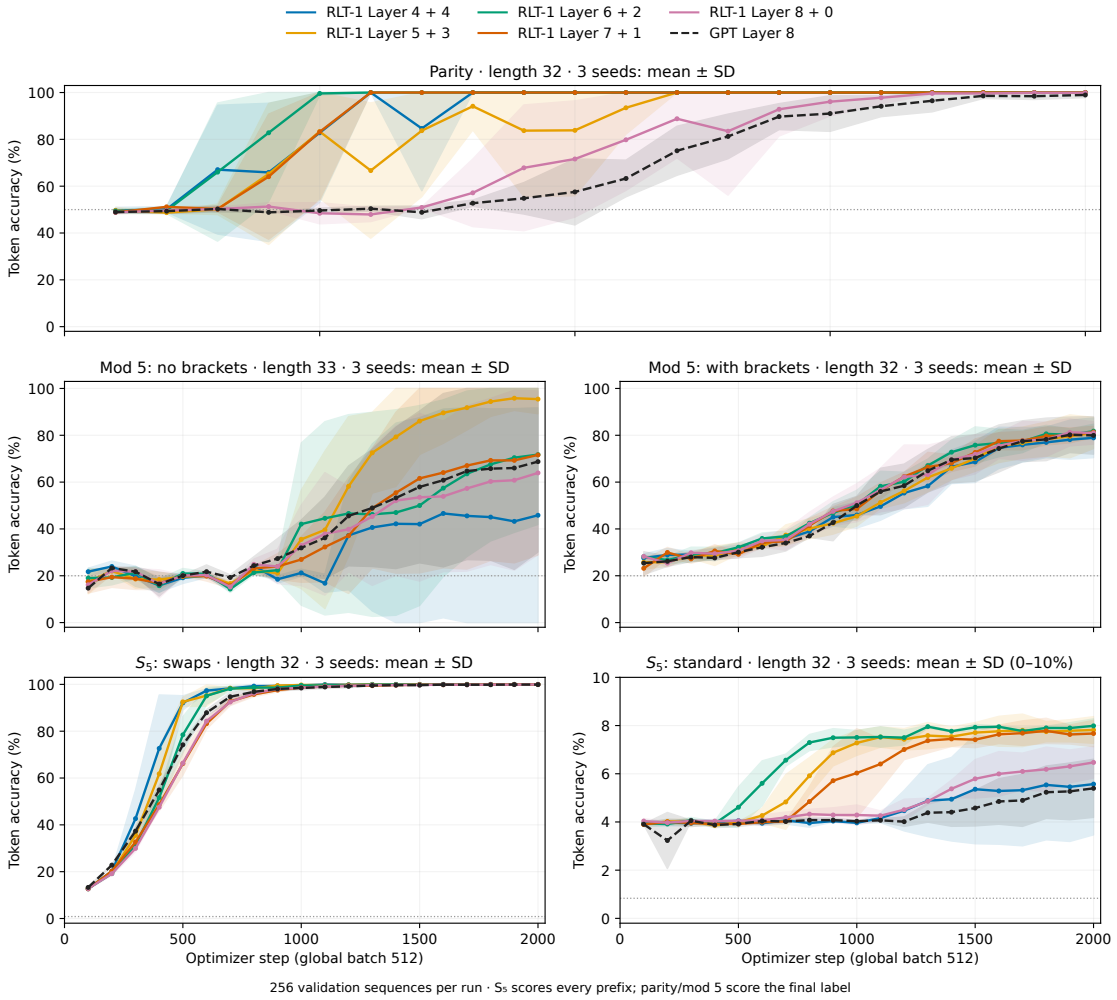


图 19 五个形式任务的验证 token 准确率，每个任务三个种子。无括号模五运算的有效内容长度为 33，其余面板为 32。每个点使用同一训练步骤的全部三个初始化种子；阴影为均值 ± 样本 SD，并裁剪到准确率范围内。S_5 对所有前缀状态评分，奇偶校验与模五运算对最终标签评分。所有曲线均延伸至第 2,000 步。标准 S_5 使用较窄的纵轴范围，横向点线表示均匀预测准确率。

表 8 十六层模型的 256 位奇偶校验，种子 42。最佳分布内损失检查点与第 2,000 步的最终答案准确率（%）。

Model	Best step	Best checkpoint	Step 2,000
RLT-1 Layer 8 + 8	1400	100.00	100.00
RLT-1 Layer 9 + 7	1300	100.00	100.00
RLT-1 Layer 10 + 6	1800	62.70	66.11
RLT-1 Layer 11 + 5	1300	100.00	100.00
RLT-1 Layer 12 + 4	1000	98.83	98.83
RLT-1 Layer 13 + 3	2000	99.41	99.41
RLT-1 Layer 14 + 2	1500	63.57	62.11
RLT-1 Layer 15 + 1	1100	94.14	94.82
RLT-1 Layer 16 + 0	1900	100.00	100.00
GPT Layer 16	2000	49.41	49.41

J.9 十六层奇偶校验：最佳与最终检查点

种子 42 的十个奇偶校验模型均已完成 2,000 次更新，训练长度为 3–40，全局批量为 1,024，微批量为 32。图 20 在每个长度的同一组 1,024 个样本上，比较分布内验证损失最低的检查点与第 2,000 步检查点。分布内损失完全相同时取最早步骤，因此 $8+8$ 和 $9+7$ 分别选择第 1,400 步和第 1,300 步。每对检查点采用相同的评估批量；最佳检查点的预测与此前报告一致。

80 个模型与长度组合中，74 个的准确率不变。在 256 位时，$10+6$ 从 62.70% 变为 66.11%，$14+2$ 从 63.57% 变为 62.11%，$15+1$ 从 94.14% 变为 94.82%。RLT-1 的 $8+8$、$9+7$、$11+5$ 和 $16+0$ 在两种检查点下均保持 100%，GPT Layer 16 为 49.41%。

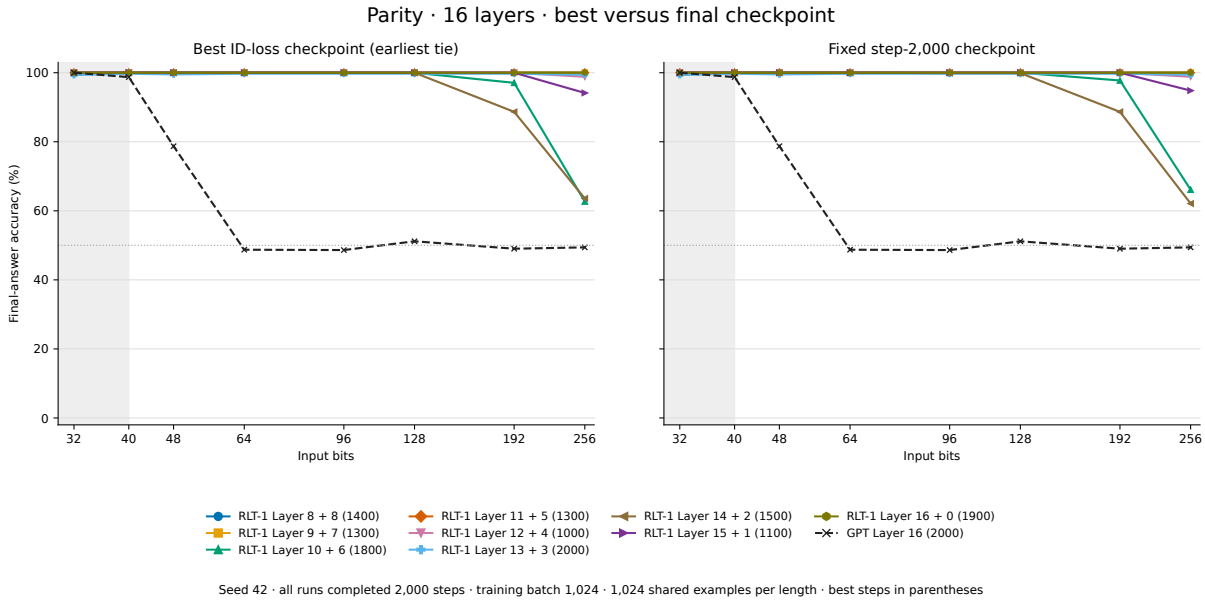


图 20　十六层奇偶校验：最佳与最终检查点。种子 42，每个长度共用 1,024 个样本；括号为最佳分布内损失检查点的步数。灰色区域为训练长度，虚线为 50% 随机参考线。

表 9 所有八层变体训练结束时的验证结果。单位为百分比，报告种子 42–44 的均值 ± 样本标准差；加法使用教师强制 token，其余任务使用最终答案或状态。加法、奇偶校验及 S_5 为 2,000 步，模五为 5,000 步。

Family	Layers	Addition	Parity	S_5 swaps	S_5 standard	Mod-5 flat	Mod-5 brackets
RLT-1	4 + 4	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	0.78 ± 0.68	46.48 ± 46.35	97.31 ± 1.14
RLT-1	5 + 3	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	2.47 ± 1.26	99.13 ± 1.28	98.26 ± 0.49
RLT-1	6 + 2	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	2.21 ± 1.48	99.65 ± 0.33	98.00 ± 1.48
RLT-1	7 + 1	100.00 ± 0.00	100.00 ± 0.00	99.35 ± 0.81	2.08 ± 0.98	98.39 ± 2.78	97.14 ± 1.16
RLT-1	8 + 0	100.00 ± 0.00	98.83 ± 1.92	99.61 ± 0.68	1.82 ± 1.19	81.12 ± 30.90	96.57 ± 0.85
RLT-0	4 + 4	100.00 ± 0.00	66.45 ± 18.12	98.96 ± 0.23	0.65 ± 0.81	93.49 ± 1.92	94.27 ± 2.00
RLT-0	5 + 3	100.00 ± 0.00	69.66 ± 10.15	98.05 ± 0.68	0.39 ± 0.39	73.52 ± 24.88	92.53 ± 2.76
RLT-0	6 + 2	100.00 ± 0.00	58.51 ± 1.50	98.57 ± 0.60	1.56 ± 1.17	63.72 ± 38.67	95.88 ± 1.64
RLT-0	7 + 1	100.00 ± 0.00	83.12 ± 25.82	99.09 ± 0.81	1.43 ± 0.60	97.48 ± 1.01	97.74 ± 0.53
RLT-0	8 + 0	100.00 ± 0.00	85.68 ± 18.25	98.96 ± 0.81	0.91 ± 0.60	97.44 ± 1.73	97.35 ± 0.67
RLT-2 chunk4	4 + 4	100.00 ± 0.00	99.87 ± 0.23	100.00 ± 0.00	2.47 ± 1.19	73.87 ± 44.47	96.40 ± 1.09
RLT-2 chunk4	5 + 3	100.00 ± 0.00	83.12 ± 29.13	99.48 ± 0.23	1.04 ± 0.60	71.74 ± 45.58	98.00 ± 0.60
RLT-2 chunk4	6 + 2	100.00 ± 0.00	95.49 ± 7.82	99.48 ± 0.23	1.95 ± 1.70	98.91 ± 1.33	97.09 ± 1.66
RLT-2 chunk4	7 + 1	100.00 ± 0.00	78.43 ± 16.65	98.83 ± 1.70	0.65 ± 0.23	97.70 ± 2.56	97.01 ± 1.38
RLT-2 chunk4	8 + 0	100.00 ± 0.00	76.52 ± 23.05	99.48 ± 0.90	0.78 ± 0.78	98.78 ± 0.08	96.57 ± 0.42
GPT	8	100.00 ± 0.00	94.84 ± 3.43	99.09 ± 0.23	0.52 ± 0.23	99.13 ± 0.78	97.01 ± 1.04

表 10 4 + 4 最佳检查点的测试准确率。最终答案或状态准确率（%），三个种子的均值 ± 样本标准差，每个长度共用 1,024 个样本。标准 S_5 的长度 32 属于分布内，其余行均为分布外。

Task / length	RLT-1	RLT-0	RLT-2 chunk4	GPT Layer 8
Parity / 64	100.00 ± 0.00	50.23 ± 3.80	98.99 ± 1.66	48.47 ± 1.18
S_5 swaps / 48	100.00 ± 0.00	7.26 ± 3.17	68.29 ± 8.75	22.10 ± 2.76
S_5 standard / 32	1.53 ± 0.46	1.20 ± 0.72	1.37 ± 0.87	0.91 ± 0.15
Mod-5 flat / 63	44.43 ± 43.81	21.84 ± 2.17	50.78 ± 38.02	33.20 ± 2.33
Mod-5 brackets / 64	64.10 ± 3.49	39.45 ± 0.61	62.76 ± 4.42	46.71 ± 1.21

J.10 反馈变体的三种子比较

我们将八层实验扩展至 RLT-0 和 RLT-2 chunk4，均覆盖 4 + 4 至 8 + 0 和种子 42、43、44。这两个系列共 180 次运行，模五训练 5,000 步，其余四项任务训练 2,000 步。宽度、数据生成、优化器、批量和 CPU 执行配置与第 3 节一致；余弦学习率调度在各自训练结束时完成。RLT-1 与 RLT-2 的反馈系数为 0.1，使用 TBPTT128；RLT-0 移除反馈融合，每种配比减少 787,968 个参数，使用完整 BPTT。加法、奇偶校验和两种 S_5 的 RLT-1 与 Transformer 对照沿用原三种子实验。

表 9 报告全部 288 次运行训练结束时的验证结果。图 21 展示 4 + 4 训练曲线，数据仓库提供全部配比的曲线。每个验证均值都包含同一步的种子 42、43、44，包括模五 Transformer 对照。

形式任务按照最低分布内验证样本损失选择检查点，同分取最早步骤。我们从保存的预测重新统计最终状态、前缀 token 和全序列准确率，并核验不同变体与种子使用相同的测试集。每个长度有 1,024 个样本，沿用原评估程序的测试长度，最长为 256，无括号模五为 255。第 3.2 节的 512 次操作 S_5 结果仅涉及 RLT-1 与 Transformer。

第 3 节讨论奇偶校验与交换 S_5 的结果；表 10 给出 4 + 4 在选定长度上的数值，包括两项模五任务。RLT-0 与 RLT-2 的 30 次加法训练，在第 2,000 步的混合 1–8 位验证集上均达到 100% 教师强制 token 准确率。9–32 位图比较的是 RLT-1 与 Transformer。

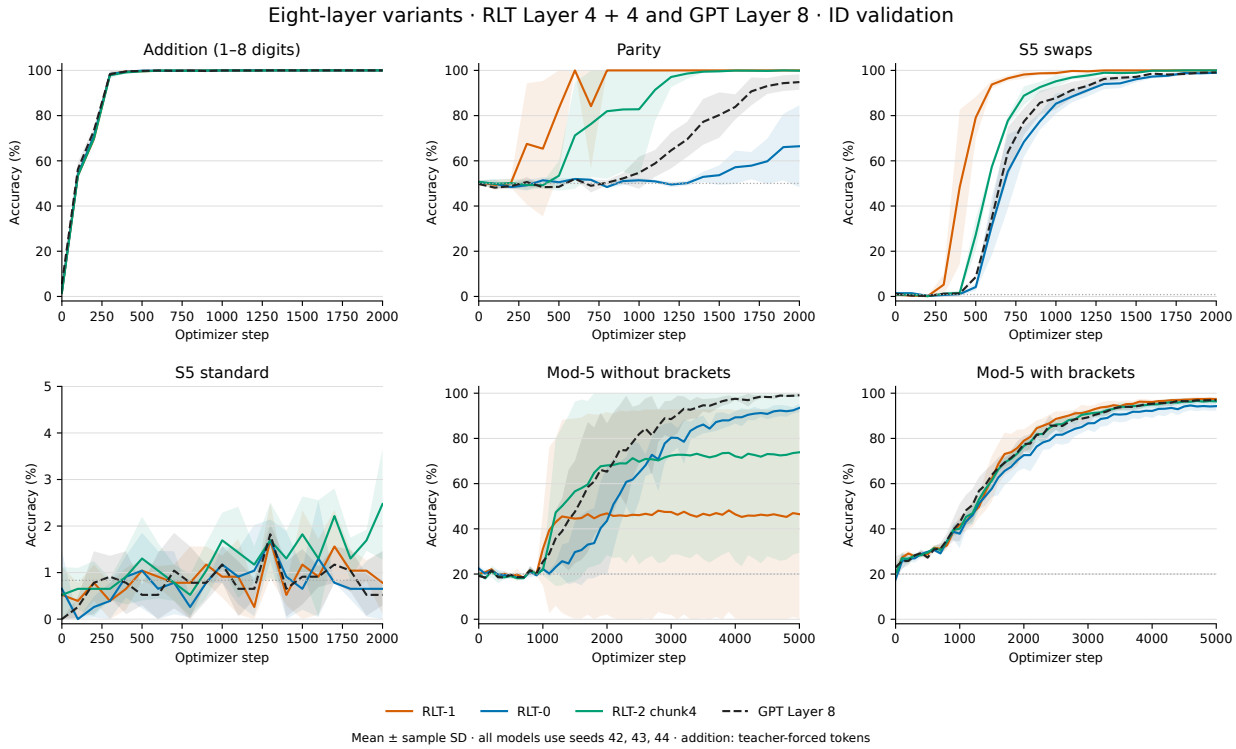


图 21 $4+4$ **反馈变体的训练验证曲线**。所有模型的曲线和阴影均为种子 42–44 的均值 ± 样本标准差。加法使用教师强制答案 token，其余任务使用最终答案或状态。阴影裁剪至 0–100%，标准 S_5 使用更窄的纵轴。

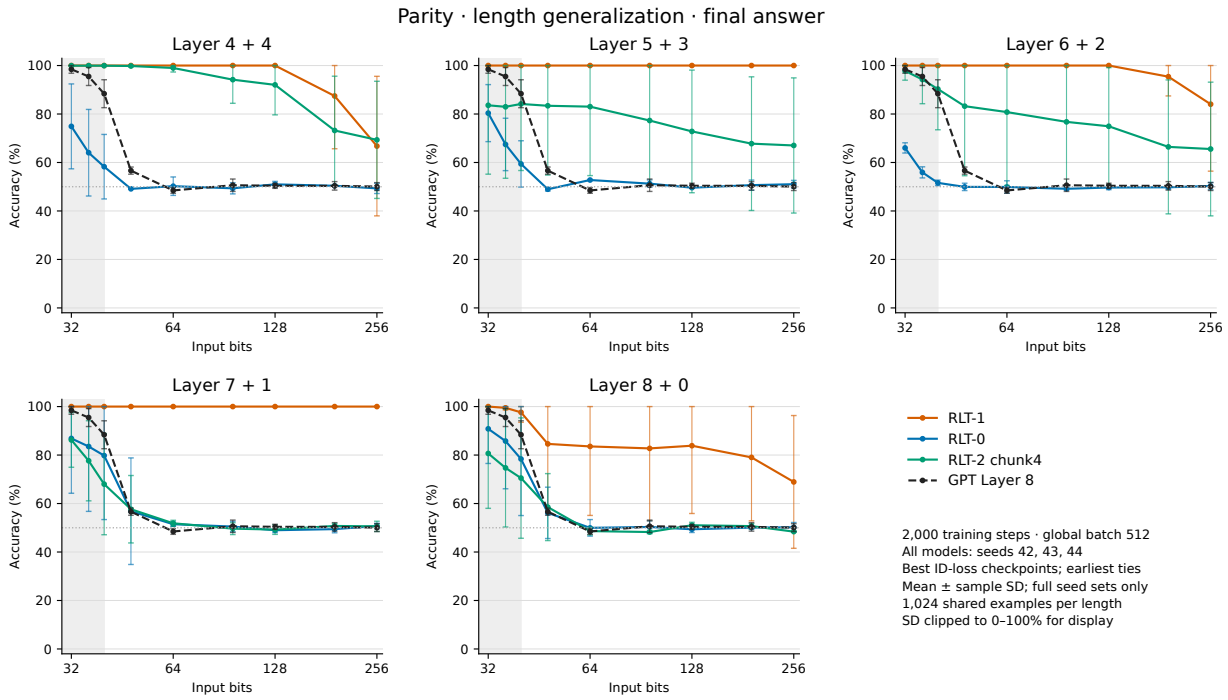


图 22　各反馈变体和层数配比的奇偶校验长度泛化。所有模型使用三个种子，训练 2,000 步，并选择最佳分布内损失检查点。误差棒为样本标准差，显示范围裁剪至 0–100%；灰色区域为训练长度，虚线为随机参考线。

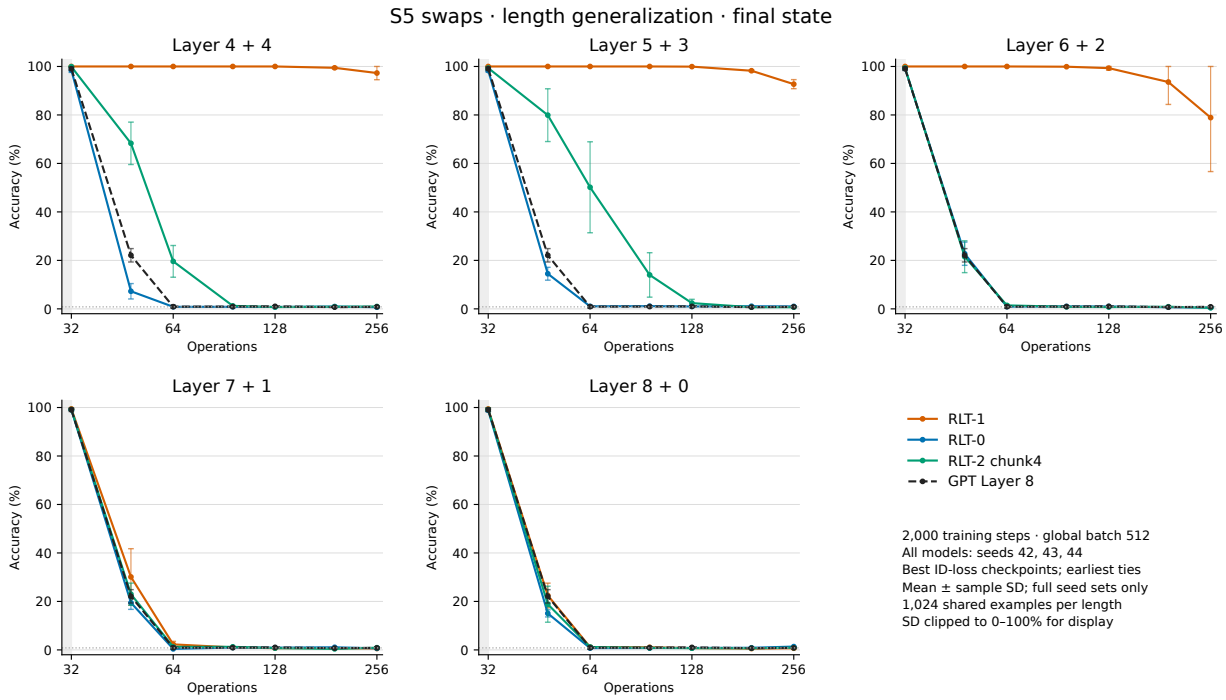


图 23 各反馈变体的交换版 S_5 长度泛化。最终状态准确率为三个种子的均值 ± 样本标准差，各自从 2,000 步训练中选择分布内验证损失最低的检查点。误差棒显示范围裁剪至 0–100%。

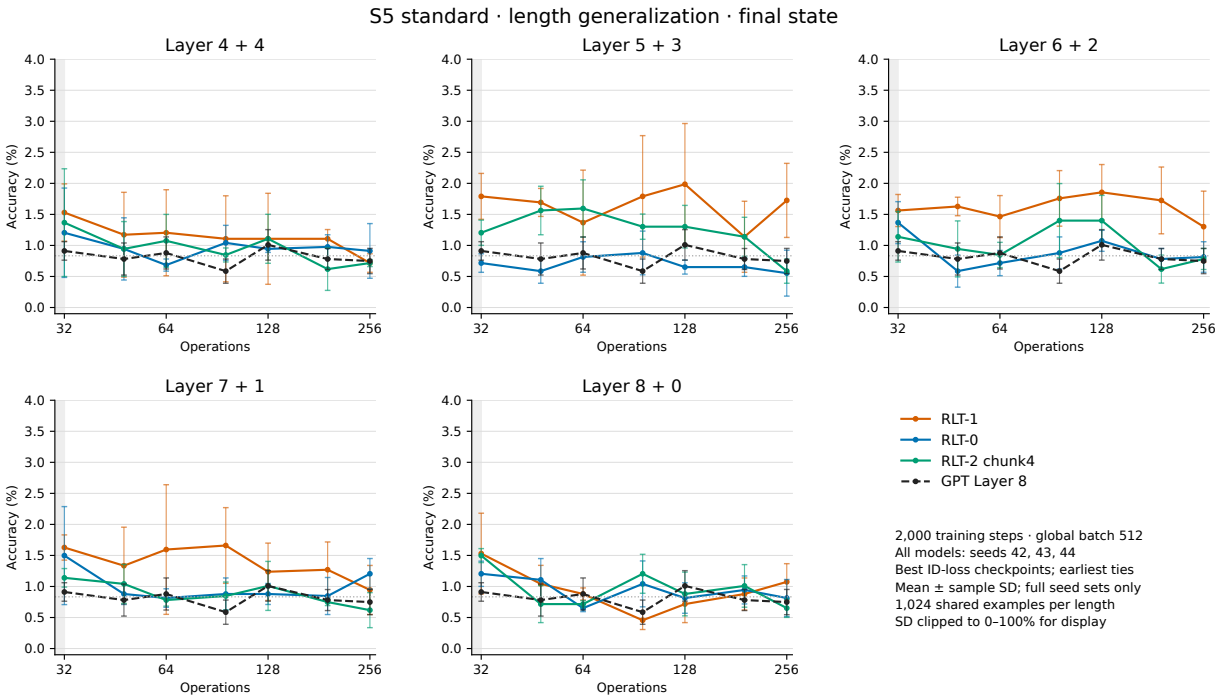


图 24　各反馈变体的标准 S_5 长度泛化。最终状态准确率，三个种子的均值 ± 样本标准差；所有模型训练 2,000 步。纵轴放大随机参考线（1/120）附近的范围。

J.11 5,000 步模五反馈变体实验

全部 96 次模五运行均训练 5,000 次更新。RLT-0、RLT-1 和 RLT-2 chunk4 各有 30 次运行，覆盖两项任务、五种配比和种子 42–44。Transformer 共六次运行，两项任务均使用这三个种子。

无括号模五在训练 5,000 步后仍对初始化敏感。chunk4 的 $4+4$ 在长度 33 上，最佳检查点的分布内准确率按种子 42、43、44 排列为 100.00%、20.21%、98.73%。长度 63 时，其三种子均值为 $50.78 \pm 38.02\%$，同配比 RLT-1 为 $44.43 \pm 43.81\%$，RLT-0 为 $21.84 \pm 2.17\%$，GPT Layer 8 为 $33.20 \pm 2.33\%$。

带括号模五在长度 64、$4+4$ 时，RLT-1 为 $64.10 \pm 3.49\%$，chunk4 为 $62.76 \pm 4.42\%$，RLT-0 为 $39.45 \pm 0.61\%$，GPT Layer 8 为 $46.71 \pm 1.21\%$。

种子 42 的 CPU 计时中，两项模五任务的 $4+4$ 配比下，chunk4 每步训练速度为 RLT-1 的 2.27 倍，RLT-0 为 4.17–4.30 倍。这些单种子均值来自共享集群的第 1,001–2,000 步，不含验证、检查点写入和日志时间。冻结结果、逐种子记录、数据来源和图表生成命令位于 `data/variants-20260925/`。

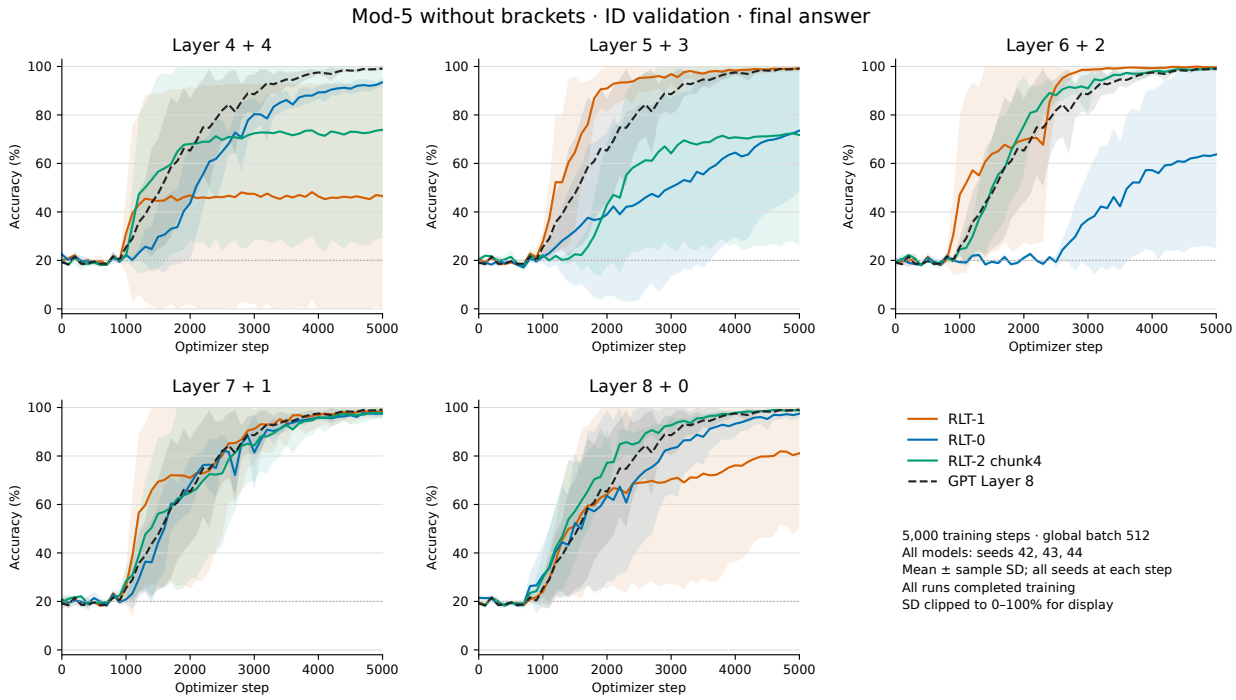


图 25　无括号模五：分布内验证。所有模型均报告种子 42–44 的均值 ± 样本标准差，均训练 5,000 步。阴影裁剪至 0–100%。

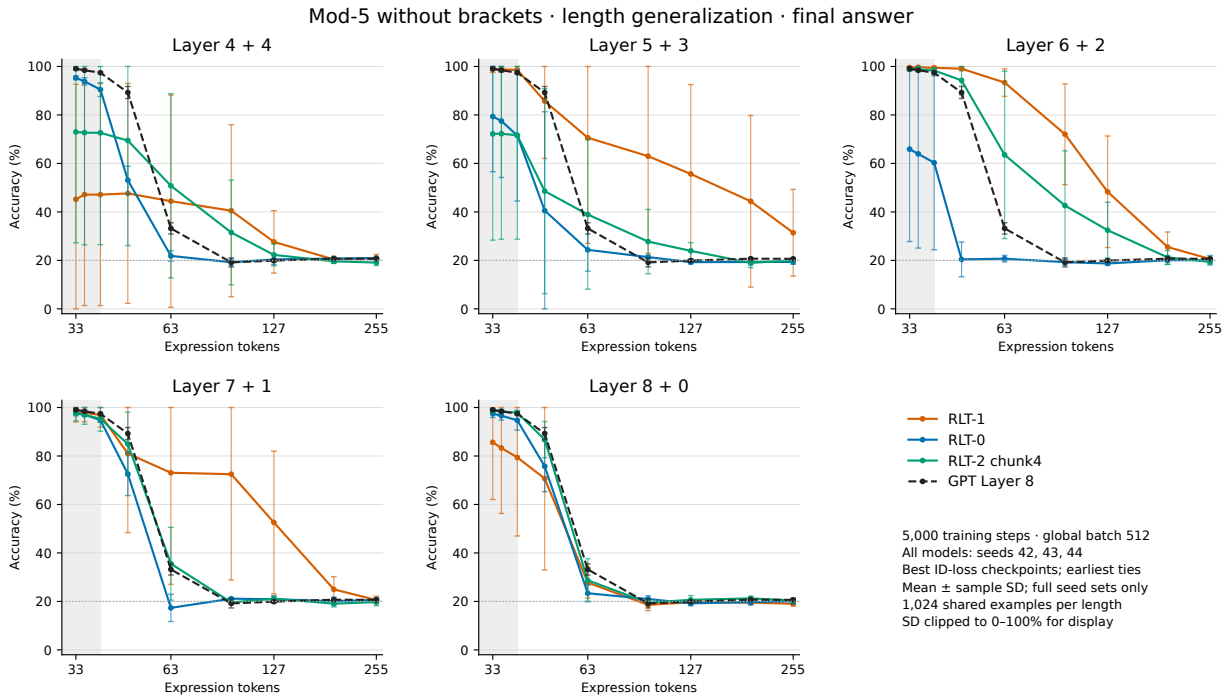


图 26　无括号模五：长度泛化。训练 5,000 步后按最佳分布内损失选择检查点，所有曲线均报告种子 42–44 的均值 ± 样本标准差。误差棒显示范围裁剪至 0–100%。

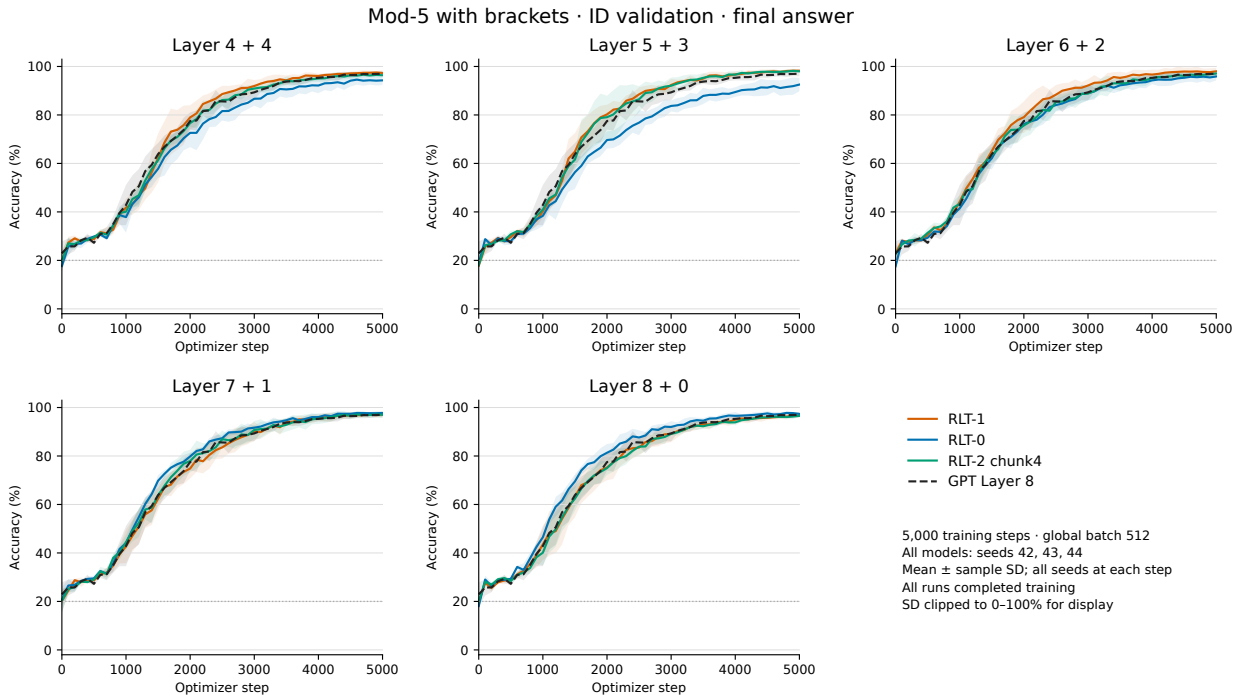


图 27　带括号模五：分布内验证。所有模型均报告种子 42–44 的均值 ± 样本标准差，均训练 5,000 步。阴影裁剪至 0–100%。

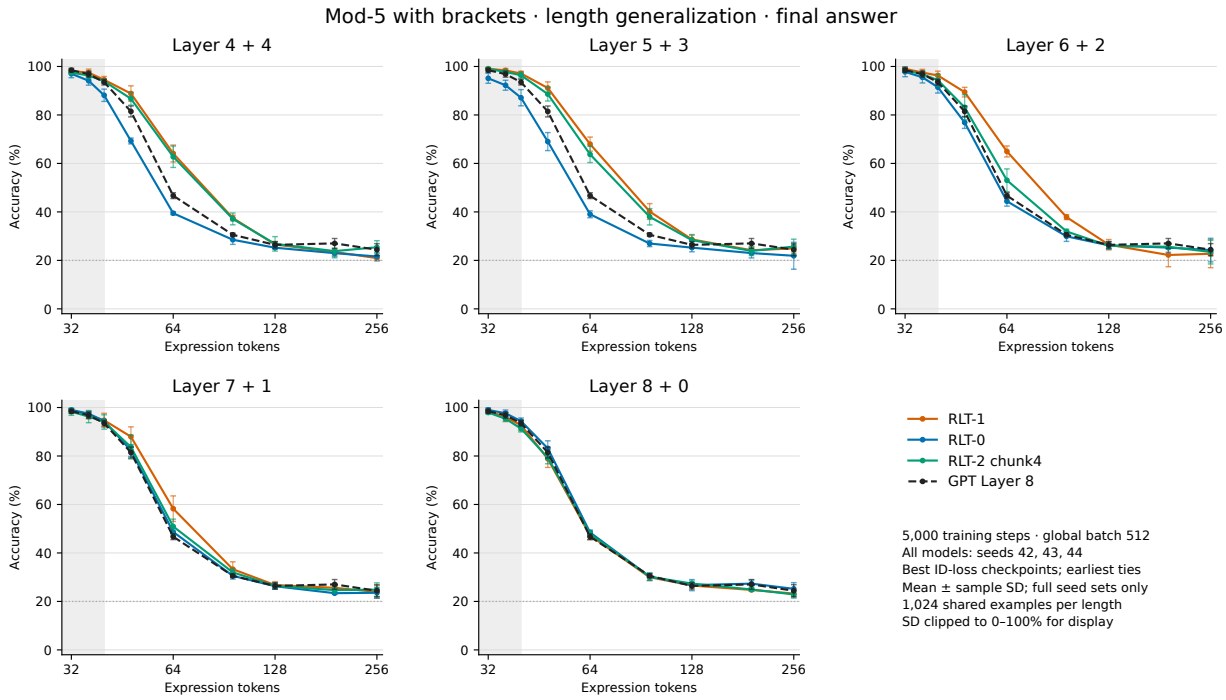


图 28　带括号模五：长度泛化。训练 5,000 步后按最佳分布内损失选择检查点，所有曲线均报告种子 42–44 的均值 ± 样本标准差。误差棒显示范围裁剪至 0–100%。

J.12 固定学习率调度下的加法反馈系数

我们仅改变 RLT-1 的反馈系数 α，在每种配比和种子下保持初始化、架构、训练数据和优化器调度不变。所有模型在混合 1–8 位加法上训练 2,000 步，全局批量为 512，微批量为 32，使用 TBPTT128。学习率经过 200 步预热达到 10^{-4}，随后衰减至 5×10^{-6}。所有配比均测试 $\alpha = 0.03$，除 $7+1$ 外的配比另测 $\alpha = 0.01$，每种设置使用种子 42、43、44。比较对象为相同种子下 $\alpha = 0.1$ 的 RLT-1 和 GPT Layer 8，共 45 个模型。

每个模型按最低分布内验证损失选择检查点，同分取最早步骤；$7+1$、$\alpha = 0.1$、种子 44 的模型选中第 1,900 步，其余 44 个模型均选中第 2,000 步。图 29 与表 11 报告 9 至 32 位共七种长度的教师强制答案 token 准确率，每个长度共用 256 个样本，结果为三个种子的均值 ± 样本标准差。计数包含答案格式和 EOS，排除提示与填充位置。

所测试的反馈系数减小方案中，没有一种能在某个配比的所有测试位数上都提高准确率。在相同配比和位数下与 $\alpha = 0.1$ 的 63 组比较中，37 组均值上升、26 组下降，变化幅度为 -3.86 至 $+5.60$ 个百分点；只有三组差值同时超过两者的样本标准差。对 $8+0$，$\alpha = 0.01$ 使 12 位均值从 $29.49 \pm 4.38\%$ 升至 $32.21 \pm 5.13\%$，16 位均值则从 $24.92 \pm 0.39\%$ 变为 $24.66 \pm 4.88\%$。32 位时，RLT-1 各模型均值为 14.44%–15.91%，GPT Layer 8 为 $16.84 \pm 1.45\%$。

表 11 加法反馈系数消融，种子 42–44。采用相同学习率调度训练 2,000 步后，所选位数的教师强制答案 token 准确率（%），为三个种子的均值 ± 样本标准差。每个长度共用 256 个样本，数据仓库提供全部七种位数。

Model	α	9 digits	12 digits	16 digits	32 digits
RLT-1 Layer 4 + 4	0.01	47.25 ± 4.04	33.11 ± 4.56	25.93 ± 1.88	14.44 ± 0.17
RLT-1 Layer 4 + 4	0.03	48.75 ± 5.34	32.00 ± 2.07	26.22 ± 0.74	14.74 ± 1.25
RLT-1 Layer 4 + 4	0.1	46.77 ± 1.34	30.72 ± 0.58	25.28 ± 1.48	15.30 ± 0.44
RLT-1 Layer 5 + 3	0.01	57.96 ± 0.67	32.36 ± 2.93	23.01 ± 5.22	15.29 ± 2.19
RLT-1 Layer 5 + 3	0.03	56.30 ± 2.41	32.60 ± 5.61	23.54 ± 3.75	14.66 ± 1.49
RLT-1 Layer 5 + 3	0.1	58.51 ± 8.12	34.97 ± 4.54	24.20 ± 5.26	14.89 ± 2.27
RLT-1 Layer 6 + 2	0.01	56.52 ± 9.87	30.97 ± 4.64	26.35 ± 2.98	15.21 ± 1.46
RLT-1 Layer 6 + 2	0.03	63.85 ± 10.15	30.39 ± 2.85	27.18 ± 4.26	14.86 ± 0.65
RLT-1 Layer 6 + 2	0.1	58.25 ± 10.29	29.22 ± 2.28	23.92 ± 1.58	15.74 ± 1.58
RLT-1 Layer 7 + 1	0.03	66.43 ± 2.53	30.10 ± 2.29	26.27 ± 1.42	15.91 ± 0.63
RLT-1 Layer 7 + 1	0.1	68.05 ± 3.64	29.80 ± 4.03	26.45 ± 4.31	15.59 ± 1.57
RLT-1 Layer 8 + 0	0.01	62.20 ± 9.85	32.21 ± 5.13	24.66 ± 4.88	15.45 ± 0.66
RLT-1 Layer 8 + 0	0.03	59.18 ± 4.47	28.81 ± 3.57	25.92 ± 1.10	14.90 ± 0.54
RLT-1 Layer 8 + 0	0.1	62.13 ± 7.75	29.49 ± 4.38	24.92 ± 0.39	15.31 ± 0.93
GPT Layer 8	–	58.61 ± 2.44	26.84 ± 3.69	26.22 ± 1.54	16.84 ± 1.45

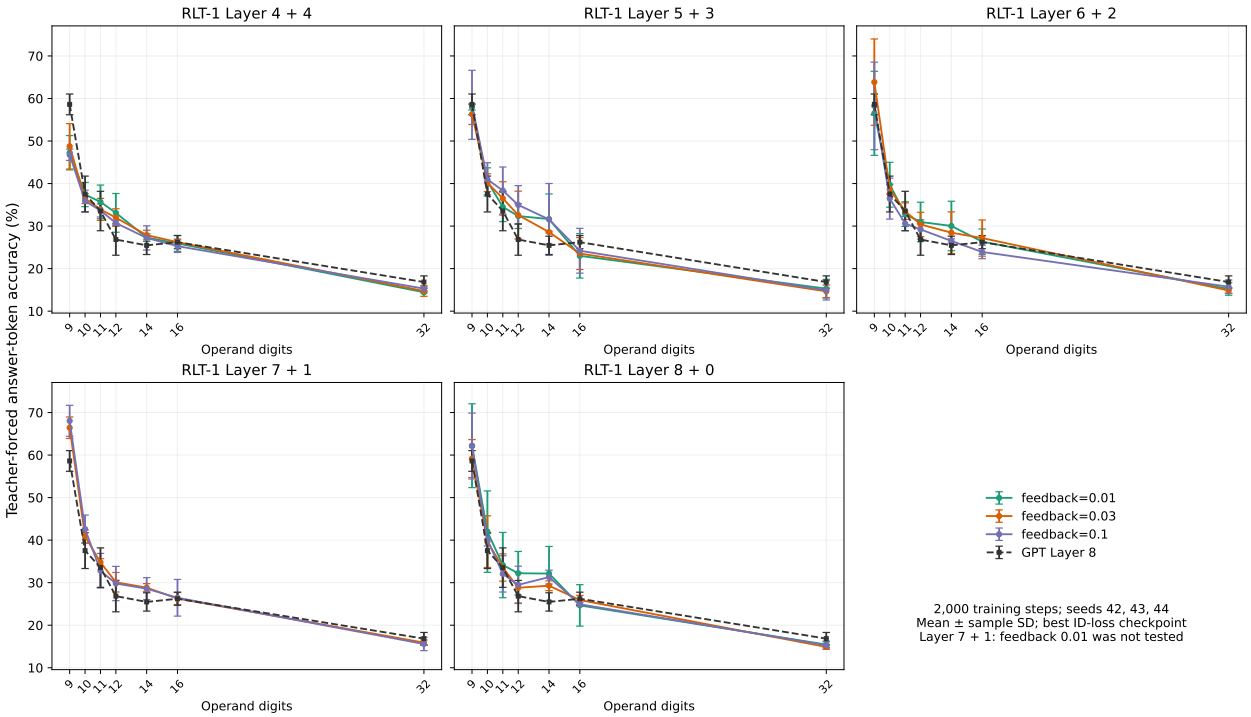


图 29　不同反馈系数下的加法泛化。每个子图固定一种 RLT-1 配比，比较已测试的反馈系数与 GPT Layer 8；$7+1$ 未测试 $\alpha = 0.01$。点和误差棒表示种子 42、43、44 的均值 ± 样本标准差；所有模型训练 2,000 步，学习率调度相同。准确率为教师强制答案 token 准确率，每个长度共用 256 个样本。