一种用于大语言模型高效推理的状态转移框架

Liang Zhang[1,3], **Yu Zhao**[2], **Longyue Wang**[2], **Tianqi Shi**[2], **Weihua Luo**[2], **Kaifu Zhang**[2], **Jinsong Su**[1,3,4*]
[1]厦门大学信息学院，中国 [2]阿里巴巴国际数字商业[3]福建与台湾非物质文化遗产数字化保护与智能处理重点实验室（厦门大学），中国文化和旅游部[4]上海人工智能实验室
lzhang@stu.xmu.edu.cn, jssu@xmu.edu.cn{fengli.zy, wanglongyue.wly, weihua.luowh}@alibaba-inc.com

arXiv:2602.01198v1 [cs.AI] 1 Feb 2026

摘要

尽管长链式思维（思维链）推理显著提升了大语言模型在复杂推理任务上的表现，但生成长思维链序列所需的大量计算与内存开销，限制了其效率和实用性。现有研究通常通过压缩思维链序列来提高大语言模型的推理效率。然而，这种方法与测试时扩展相冲突，限制了大语言模型的推理能力。本文提出一种高效推理框架，将大语言模型的推理过程建模为状态转移过程。具体而言，我们首先采用线性注意力机制来估计大语言模型的推理状态，该状态记录了先前推理步骤中的历史推理信息。随后，大语言模型可基于查询提示和推理状态，高效执行当前推理步骤并更新状态。借助线性注意力，当前推理步骤中的每个词元都能直接从推理状态中检索相关历史推理信息，而无需显式关注先前推理步骤中的词元。由此，注意力的计算复杂度从二次降至线性，显著提升了大语言模型的推理效率。此外，我们提出一种基于状态的推理策略，以缓解噪声推理步骤引发的过度思考问题。在多个数据集和不同模型规模上的大量实验表明，我们的框架不仅提高了大语言模型的推理效率，还提升了其推理性能。

1 引言

思维链（CoT）（Wei 等，2022年；Kojima 等，2022年）已成为提升大语言模型（LLM）在复杂任务上推理能力的核心技术。通过提示模型逐步推理，思维链使大语言模型能够将复杂问题分解为更简单的子任务，从而提升其解决问题的能力（Yao 等，2023；Wang 等，2023；Zhou 等，2023；Wang 等，2024年）。近期研究，包括 OpenAI o1（OpenAI 等，2024年）、QwQ（Team, 2024年）和 DeepSeek-R1（DeepSeek 等，2025年），表明增加思维链长度能够进一步提升大语言模型的推理能力。然而，由于当前大多数大语言模型都基于 Transformer 架构（Vaswani 等，2017b），其注意力机制的计算复杂度会随上下文长度呈二次增长，而 KV 缓存的内存开销则会随上下文长度线性增长。因此，生成较长的思维链会大幅增加大语言模型的计算和内存开销，限制其在复杂推理任务中的实际效率。

为提高大语言模型的推理效率，先前研究采用提示（Han等人，2024年；Ma等人，2025年a）、监督微调（SFT）（Liu等人，2024年；Munkhbat等人，2025年）或强化学习（RL）（Aggarwal等人，2025年；Shen等人，2025年），以促使大语言模型生成更短的思维链序列。然而，这些方法往往会损害

*通讯作者。

大语言模型的推理能力（Jin等人，2024年；Merrill等人，2024年），因为思维链缩短与测试时扩展相冲突（OpenAI等人，2024年）。为保留大语言模型的推理能力，一些研究（Ma等人，2025年b；Kang等人，2025年；Xia等人，2025年）将思维链表述得更加简洁（例如，删除不太重要的词元或使用GPT-4改写），以缩短其长度。然而，简化较长的思维链时，可能会丢失关键推理信息或降低可解释性（Wang等人，2025c）。

本文提出了一种面向大语言模型的高效推理框架，将大语言模型的推理过程建模为状态转移过程。我们将较长的思维链视为由多个推理步骤构成的序列，其中大语言模型在每个步骤中执行特定的思维模式，例如归纳或反思。值得注意的是，每个推理步骤都包含两类信息：大量语言信息，以确保表达流畅；以及有限的推理信息，用于支持后续推理或答案生成（Zhang 等，2025年a；Xia 等，2025年）。因此，我们的框架（图1）首先将先前生成的推理步骤中的推理信息压缩为一个矩阵，称为推理状态矩阵。随后，大语言模型基于查询提示和状态矩阵，高效生成当前推理步骤，并相应地更新状态矩阵。具体而言，当前推理步骤中的词元可以直接从推理状态中检索相关的历史推理信息，而无需显式关注先前步骤中的词元。通过这种方式，我们有效降低了注意力的计算复杂度和 KV缓存的内存开销。重要的是，我们的框架不会缩短或简化大语言模型生成的思维链序列，因此能够保留其推理能力和可解释性。

为了高效地获得状态矩阵，我们设计了混合注意力模块（**MAM**）来替换大语言模型中原有的注意力模块，该模块由Softmax注意力（**SA**）子模块和线性注意力（**LA**）子模块组成。我们将大语言模型原有的注意力模块用作SA 子模块，其中每个词元只能关注查询提示中的词元及其当前推理步骤中的词元。在LA 子模块中，我们采用线性注意力机制（Katharopoulos et al., 2020年）来捕获大语言模型的推理状态。同时，借助线性注意力，每个词元可以直接从推理状态中检索相关的历史推理信息。与卷积神经网络（Krizhevsky et al., 2012年）和Q-Former（Li et al., 2023）等其他方法相比，线性注意力在捕获模型推理状态方面具有以下优势：（1）作为Softmax 注意力的一种变体，线性注意力与其天然兼容，从而降低压缩过程中丢失关键推理信息的风险，并确保模型训练的稳定性与效率。（2）近期研究（Yang et al., 2024b; 2025d）表明，线性注意力的状态更新过程本质上是一种梯度下降学习过程（即测试时训练），揭示了其处理复杂推理任务的强大潜力（Zhu et al., 2025年）。

另一项挑战是，大语言模型经常会产生噪声推理步骤，这些步骤可能误导后续推理步骤，并导致过度思考问题（Chen 等人，2025b；Yang 等人，2025c）。为缓解这一问题，我们提出了一种基于状态的推理策略。鉴于模型的状态转移过程是一个梯度下降过程，我们首先采用动量法累积已完成推理步骤所产生的梯度，从而获得全局梯度。全局梯度指示了大语言模型的全局推理方向。因此，我们利用它来引导大语言模型完成当前推理步骤，确保其不会显著偏离全局方向。

为验证我们框架的有效性，我们在七个广泛使用的基准数据集上进行了实验。实验结果表明，我们的框架不仅提高了大语言模型的推理效率，还提升了其推理性能。广泛的消融研究进一步证明了我们框架中各个组件的有效性。

2 初步介绍

We 首先简要介绍线性注意力的背景，为我们提出的 d
fra框架奠定基础。同时，我们概述框架中思维链序列的具体形式。

2.1 线性注意力

Softmax 注意力（SA）。主流大语言模型，如通义千问 3（Yang 等人，2025a）和 Llama 4（Meta，2025年），采用仅解码器的 Transformer 架构，由多头 Softmax 注意力与随后的前馈网络（FFN）组成的重复模块构成（Vaswani 等人，2017a）。给定输入序列 $\boldsymbol{X}=[x_1,\cdots,x_{|\boldsymbol{X}|}]$，Softmax 注意力可表述如下：

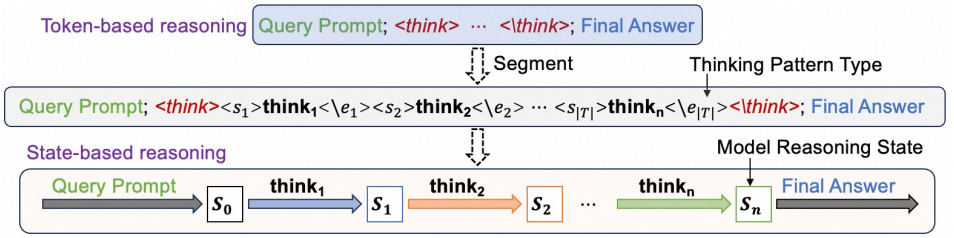


图1：传统基于令牌的推理与基于状态的推理的比较。**思考**$_t$表示一个推理步骤。在基于状态的推理中，大语言模型只需基于查询提示和推理状态 $\boldsymbol{S}_{t-1}$，即可高效生成**思考**$_t$ 。

$$\boldsymbol{o}_t = \frac{\sum_{i=1}^{t} \exp\left(\boldsymbol{q}_t \boldsymbol{k}_i^\top / \sqrt{d}\right) \boldsymbol{v}_i}{\sum_{i'=1}^{t} \exp\left(\boldsymbol{q}_t \boldsymbol{k}_{i'}^\top / \sqrt{d}\right)}; \quad \boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t = x_t \boldsymbol{W}_Q,\ x_t \boldsymbol{W}_K,\ x_t \boldsymbol{W}_V, \tag{1}$$

其中 $\boldsymbol{W}_Q$、 $\boldsymbol{W}_K$、 $\boldsymbol{W}_V$ 为可学习权重矩阵。Softmax 函数的计算复杂度随上下文序列长度呈二次增长。此外，Softmax 注意力高度依赖不断增长的 KV缓存来召回历史信息以进行序列建模，从而导致巨大的内存开销，在长上下文场景中尤为明显。

线性注意力（LA）。线性注意力是 Softmax 注意力的一种变体，旨在降低其计算复杂度和内存开销。这里，我们首先用更简单的核函数 $\phi(\cdot)$ 替换 Softmax 注意力中的指数函数 $\exp(\cdot)$: $\exp(\boldsymbol{q}_t \boldsymbol{k}_i / \sqrt{d}) \rightarrow \phi(\boldsymbol{q}_t)\phi(\boldsymbol{k}_i)^\top$。接下来，基于矩阵乘积的结合律，线性注意力可以写成

$$\boldsymbol{o}_t = \frac{\sum_{i=1}^{t} \phi(\boldsymbol{q}_t)\phi(\boldsymbol{k}_i)^\top \boldsymbol{v}_i}{\sum_{i'=1}^{t} \phi(\boldsymbol{q}_t)\phi(\boldsymbol{k}_{i'})^\top} = \frac{\phi(\boldsymbol{q}_t) \sum_{i=1}^{t} \phi(\boldsymbol{k}_i)^\top \boldsymbol{v}_i}{\phi(\boldsymbol{q}_t) \sum_{i'=1}^{t} \phi(\boldsymbol{k}_{i'})^\top}. \tag{2}$$

此外，近期研究（Sun 等，2023；Yang 等，2024a年）表明，即使将 $\phi(\cdot)$ 设为恒等函数并移除归一化器，线性注意力仍能表现良好：

$$\boldsymbol{o}_t = \boldsymbol{q}_t \sum_{i=1}^{t} \boldsymbol{k}_i^\top \boldsymbol{v}_i = \boldsymbol{q}_t \boldsymbol{S}_t; \quad \boldsymbol{S}_t = \sum_{i=1}^{t} \boldsymbol{k}_i^\top \boldsymbol{v}_i, \tag{3}$$

其中， $\boldsymbol{S}_t$ 是存储历史信息的状态矩阵。借助 $\boldsymbol{S}_t$，线性注意力可以在线性时间内进行序列建模，同时保持恒定的内存开销。

线性注意力（LA）的测试时训练（TTT）视角。近期研究（Sun 等人，2024年；Chen 等人，2025年b）将线性注意力中的状态矩阵 $\boldsymbol{S}_t$ 视为一种快速适应的参数，并通过轻量级梯度下降在每个词元处进行更新。它们进一步为线性注意力引入了一种在线学习目标，并给出了其梯度下降闭式解：

$$\begin{aligned} &\textbf{Objective:} \quad \mathcal{L}(\boldsymbol{S}) = -\langle \boldsymbol{S}\boldsymbol{k}_t, \boldsymbol{v}_t \rangle, \\ &\textbf{SGD update:} \quad \boldsymbol{S}_t = \boldsymbol{S}_{t-1} - \beta \nabla \mathcal{L}_t(\boldsymbol{S}_{t-1}) = \boldsymbol{S}_{t-1} + \beta \boldsymbol{v}_t \boldsymbol{k}_i^\top, \end{aligned} \tag{4}$$

其中 $\langle \cdot,\ \cdot \rangle$ 表示内积， β 为学习率。当 β 设为 1 时，线性注意力的状态更新过程等价于使用学习目标 $\mathcal{L}(\boldsymbol{S})$训练状态矩阵 $\boldsymbol{S}_t$ 。尽管目前的研究尚未提供直接证据证明线性注意力能够提升模型的推理能力，但其理论特性表明它具有巨大的潜力（Zhu 等人，2025年）。

2.2 长思维链分段

一个长思维链样本（$\boldsymbol{Q}$， $\boldsymbol{T}$， $\boldsymbol{A}$）通常由三部分组成：查询提示 $\boldsymbol{Q}$、思维序列$\boldsymbol{T}$和最终答案 $\boldsymbol{A}$。给定查询提示 $\boldsymbol{Q}$，大语言模型首先进行复杂推理（$\boldsymbol{T}$），然后生成最终答案 $\boldsymbol{A}$。在推理过程中，大语言模型会采用各种思维模式（例如，反思和结果验证），并使用常见的过渡词元（例如，“等等”、“嗯”）在这些模式之间切换（Yang 等人，2025c；Wang 等人，2025d）。近期研究（Wang 等人，2025年b）

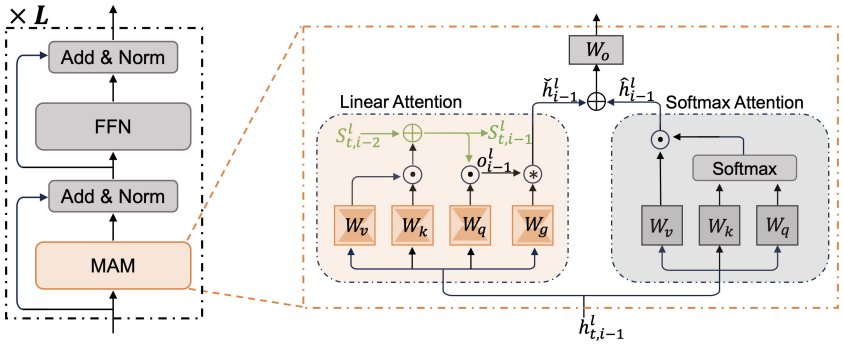


图2：在我们的框架中，大语言模型中的原始 Softmax 注意力模块被我们的混合注意力模块（MAM）取代，从而提高推理效率并降低内存开销。

进一步表明，大语言模型通常会在某些高熵词元处切换思维模式，例如“或者”和“也许”。在此，我们将大语言模型完成一种思维模式的过程称为一个推理步骤。值得注意的是，在执行当前推理步骤时，大语言模型只考虑先前已完成步骤中的推理信息（例如，结论），而不考虑其中的语言信息（例如，语法）。

遵循 Wang 等人（2025b）的方法，我们首先从长思维链训练集中提取高熵转折词元，这些词元出现在句首。接下来，我们使用这些词元将每个训练样本中的思维序列 $\boldsymbol{T}$ 划分为一系列推理步骤。同时，我们对整个训练集中的所有推理步骤进行聚类，每个簇对应一种思维模式（即类型）。最后，对于每种思维模式，我们引入两个特殊标记，将其对应的推理步骤括起来，如图1所示。借助重构后的训练集，训练出的大语言模型能够生成结构更清晰的思维序列。同时，这些特殊标记使我们能够更准确地追踪和控制大语言模型的推理过程。

3 我们的框架

本节将详细介绍我们提出的框架。在该框架中，我们首先设计了一个混合注意力模块（**MAM**），以替代大语言模型中的原始注意力模块，如图 2 所示。借助 MAM，我们可以将大语言模型的推理过程建模为状态转移过程（见第 3.1 节）。随后，我们在第 3.2 节进一步介绍基于状态的推理策略。最后，第 3.3 节介绍我们的训练策略。

3.1 混合注意力模块

我们的框架旨在将大语言模型的推理过程建模为状态转移过程，从而在不牺牲推理能力的前提下提高推理效率。为此，我们设计了一个 MAM 来替代大语言模型中的 softmax 注意力模块，该模块由 Softmax 注意力（**SA**）子模块和线性注意力（**LA**）子模块组成。为避免这种替代造成性能损失，我们采用大语言模型原有的 softmax 注意力模块作为 SA 子模块。不过，在 SA 子模块中，每个词元只能关注查询提示 $\boldsymbol{Q}$中的词元，以及其推理步骤中此前生成的词元。通过这种方式，我们将注意力的计算复杂度从二次 $O(\mathcal{C}^2)$ 降至线性 $O(\mathcal{C})$，并将 KV缓存的内存使用量从线性 $O(\mathcal{C})$ 降至常数 $O(1)$，其中 $\mathcal{C}$ 表示上下文长度。这显著提高了模型的推理效率，尤其是在需要更长思维序列的复杂场景中。此外，LA 子模块采用线性注意力机制来获取大语言模型的推理状态矩阵，该矩阵记录了此前已完成推理步骤中的推理信息。因此，当前推理步骤中的每个词元都可以从状态矩阵中获取相关历史信息，而无需直接关注先前推理步骤中的词元。

鉴于我们的 大语言模型在每个推理步骤中都采用相同的过程生成每个词元，我们以第 t 个推理步骤中第 i 个词元 $x_{t,\,i}$ 的生成为例。具体而言，在第 l 个

层中，我们首先将 $h_{t\ i-1}^{(l-1)}$，，即输入词元 $x_{t,\ i-1}$ 在第 (l-1) 层的输出特征，输入当前层的 MAM。然后，$h_{t\ i-1}^{(l-1)}$，被传递至 MAM 中的两个子模块：

在 SA 子模块中，KV缓存仅保留查询提示词元以及当前推理步骤中先前生成的词元所对应的 KV向量，而已完成推理步骤中的词元所对应的 KV向量则会被移除。通过 softmax 注意力机制（见公式 1），输入词元 $x_{t,\ i-1}$ 会关注 KV缓存中保留的词元，从而得到更新后的特征 $\hat{h}_{t}^{(l)}{}_{,\ i-1}$。

在 LA 子模块中，完成前 $t-1$ 个推理步骤后，我们的模型状态更新为$\boldsymbol{S}_{t-1}^{(l)}$。随后，我们将该状态作为当前推理步骤的初始状态 $\boldsymbol{S}_{t}^{(l)}{}_{,\ 0}=\boldsymbol{S}_{t-1}^{(l)}$ 并逐词元对其进行更新，从而得到当前模型状态 $\boldsymbol{S}_{t}^{(l)}{}_{,\ i-1}=\boldsymbol{S}_{t}^{(l)}{}_{,\ 0}+\sum_{j=1}^{i-1}\boldsymbol{k}_{j}^{(l)^{\top}}\boldsymbol{v}_{j}^{(l)}$。接着，我们利用输入词元 x_{i-1} 的查询向量 $\boldsymbol{q}_{i-1}^{(l)}$，从 $\boldsymbol{S}_{t}^{(l)}{}_{,\ i-1}$中提取相关的历史推理信息$\boldsymbol{o}_{i-1}^{(l)}$：$\boldsymbol{o}_{i-1}^{(l)}=\boldsymbol{q}_{i-1}^{(l)}\boldsymbol{S}_{t}^{(l)}{}_{,\ i-1}$。在当前推理步骤中，模型通常更多地依赖历史推理信息来生成早期阶段的词元，而随着生成的词元增多，这种依赖会逐渐减弱。因此，我们通过门控机制获得该子模块的输出：$\check{h}_{t}^{(l)}{}_{,\ i-1}=\sigma($ $\boldsymbol{W}_{g}h_{t\ i-1}^{(l-1)}{}_{,}$ $)\ *\ \boldsymbol{o}_{i-1}^{(l)}$，其中 $\sigma(\cdot)$ 表示 Sigmoid 函数，$\boldsymbol{W}_{g}$ 是可学习权重（见图 2）。

最后，我们将两个子模块的输出相结合，并应用线性输出层，以得到 MAM 的最终输出：$\tilde{h}_{t}^{(l)}{}_{,\ i-1}=\boldsymbol{W}_{o}(\check{h}_{t}^{(l)}{}_{,\ i-1}+\hat{h}_{t}^{(l)}{}_{,\ i-1})$。如图 2 所示，这两个子模块的结构完全相同，但 LA 子模块额外引入了一个门控权重 $\boldsymbol{W}_{g}$。为控制参数规模，我们通过 LoRA 策略（Hu 等，2022）实现 LA 子模块。

随后，我们使用前馈网络模块处理 $\tilde{h}_{t}^{(l)}{}_{,\ i-1}$，生成模型第 l 层的输出 $h_{t}^{(l)}{}_{,\ i-1}$。重复上述过程 L 次后，我们得到输入词元 x_{t-1} 的最终输出特征 $h_{t}^{(L)}{}_{,\ i-1}$，其中 L 表示模型中的层数。接下来，将 $h_{t}^{(L)}{}_{,\ i-1}$ 输入模型的预测层，以生成下一个词元的预测分布 $p(x)$。最后，我们的模型根据 $x_{i}=\arg\max_{x}p(x)$ 生成第 t 个推理步骤中的第 i 个词元 x_{i}。

按照上述流程，我们的模型会在当前推理步骤中依次生成词元，直到生成该步骤对应思维模式的结束标记 $\langle\backslash e_{t}\rangle$（见图1）。生成 $\langle\backslash e_{t}\rangle$ 后，我们模型在第 l 层的状态矩阵会更新为 $\boldsymbol{S}_{t}^{(l)}{}_{,\ n_{t}}$，我们将其记为 $\boldsymbol{S}_{t}^{(l)}$，并用作下一推理步骤的初始状态，其中 n_{i} 表示第 i 个推理步骤中的词元数量。同时，我们会从 KV缓存中清除当前推理步骤所有词元的 KV向量。

3.2 基于状态的推理策略

在推理过程中，大语言模型常会产生噪声推理步骤，进而误导后续步骤，导致过度思考问题（Chen 等人，2025b；Yang 等人，2025c）。在我们的框架中，此类噪声推理步骤可能会使模型的状态转移偏离正确的推理轨迹，进而产生错误结果（见图3(a)）。为缓解这一问题，我们提出了一种基于状态的推理策略，通过全局推理方向引导模型进行推理。

在推理过程中，我们的模型在第 l 层的状态转移可形式化为：$\boldsymbol{S}_{0}^{l}\rightarrow\boldsymbol{S}_{1}^{l}\rightarrow\cdots\rightarrow\boldsymbol{S}_{|\boldsymbol{T}|}^{l}$，其中 $\boldsymbol{S}_{0}^{l}$ 表示 LA 子模块处理查询提示 $\boldsymbol{Q}$ 中的词元后得到的模型状态。鉴于线性注意力中的状态转移过程是一个梯度下降过程（参见第2.1节），我们将每个推理步骤贡献的总梯度表示为 $[\nabla_{1}^{l},\nabla_{2}^{l},\cdots,\nabla_{|\boldsymbol{T}|}^{l}]$，其中 $\nabla_{t}^{l}=\boldsymbol{S}_{t}^{l}-\boldsymbol{S}_{t-1}^{l}$ 表示第 t 步的推理方向。噪声推理步骤的推理方向往往会与其他步骤的推理方向产生显著偏差。

因此，我们首先通过动量累积将所有先前步骤的推理方向聚合为一个全局推理方向 $\bar{\nabla}_{t-1}^{l}$：$\bar{\nabla}_{t-1}^{l}=(1-{}_{t-1}^{1})\,\bar{\nabla}_{t-2}^{l}+{}_{t-1}^{1}\nabla_{t-1}^{l}={}_{t-1}^{1}\sum_{i=1}^{t-1}\nabla_{i}^{l}$，其中 $\bar{\nabla}_{0}^{l}$初始化为零矩阵。然后，在完成第 t 个推理步骤后，我们使用全局方向 $\bar{\nabla}_{t-1}^{l}$ 校正其推理方向 ∇_{t}^{l}：$\hat{}\ \nabla_{t}^{l}=(1-\alpha)\nabla_{t}^{l}+\alpha\bar{\nabla}_{t-1}^{l}$，其中$\alpha=\max\{\alpha_{\max},{}_{|\boldsymbol{T}|}^{t}\}$，$|\boldsymbol{T}|$ 表示推理步骤的最大数量（默认值：40）。由于

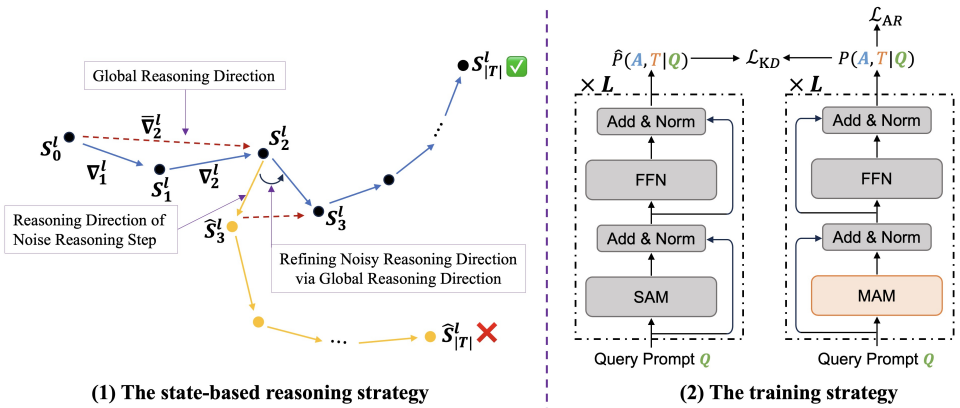


Figure 3：我们的推理与训练策略示意图。SAM 是 softmax 注意力模块 .

更多的先前推理步骤会带来更准确的全局推理方向，因此我们将校正系数 α 线性增大至预设阈值 $\alpha_{\max}$。这样一来，我们的模型便能在推理初期充分探索不同的推理方向，随后逐渐收敛至全局推理方向。最后，我们使用校正后的推理方向 $\hat{\nabla}_t^l$ 更新模型状态，$\boldsymbol{S}_t^l = \boldsymbol{S}_{t-1}^l + \hat{\nabla}_t^l$，从而减轻噪声步骤对后续步骤的负面影响。

为增强推理过程中思维模式的多样性，我们使用指示思维模式的特殊标记，引导模型在连续步骤中采用不同的思维模式。由此，我们可以进一步增强模型的稳健性与总体性能。

3.3 模型训练

我们使用第2.1节构建的长思维链训练集来训练模型。为在保持大语言模型原有推理能力的同时提高训练效率，我们仅微调新加入的 LA 子模块以及与思维模式对应的特殊标记的参数。如图 3(b) 所示，我们用两项损失联合优化模型：（1）模型在训练样本上的自回归损失 $\mathcal{L}_{\mathrm{AR}}$，以及（2）基础模型（带有 softmax 注意力模块的原始大语言模型）与我们所提出模型之间的知识蒸馏损失 $\mathcal{L}_{\mathrm{KD}}$。最终，训练目标定义为 $\mathcal{L} = \mathcal{L}_{\mathrm{AR}} + \beta \mathcal{L}_{\mathrm{KD}}$，其中 β 表示一个超参数。

具体而言，我们首先将一个长思维链样本输入我们的模型，以获得在查询提示 $\boldsymbol{Q}$ 条件下、关于思维序列 $\boldsymbol{T}$ 和最终答案 $\boldsymbol{A}$ 的预测概率分布 $P(\boldsymbol{A}, \boldsymbol{T} | \boldsymbol{Q})$。为保持训练与测试之间的一致性，我们在 SA 子模块中应用定制掩码矩阵，以确保每个词元仅关注查询提示 $\boldsymbol{A}$ 中的词元及其对应推理步骤中的词元。接下来，我们的自回归损失项 $\mathcal{L}_{\mathrm{AR}}$ 定义如下：$\mathcal{L}_{\mathrm{AR}} = -\log P(\boldsymbol{A}, \boldsymbol{T} | \boldsymbol{Q})$。

与此同时，将同一长思维链样本输入基础模型，以产生预测概率 $\hat{}$ 分布 $P(\boldsymbol{A}, \boldsymbol{T} | \boldsymbol{Q})$。值得注意的是，我们的模型是在基础模型之上构建的，方法是将其 softmax 注意力模块替换为我们的 MAM。在基础模型中，每个词元都会关注所有先前的词元。随后，我们使用 $P(\boldsymbol{A}, \boldsymbol{T} | \boldsymbol{Q})$ 与 $\hat{} P(\boldsymbol{A}, \boldsymbol{T} | \boldsymbol{Q})$ $\hat{}$ 之间的 Kullback–Leibler（KL）散度作为我们的蒸馏损失项 $\mathcal{L}_{\mathrm{KD}} = \mathbf{KL}(P(\boldsymbol{A}, \boldsymbol{T} | \boldsymbol{Q}) || P(\boldsymbol{A}, \boldsymbol{T} | \boldsymbol{Q}))$。$\mathcal{L}_{\mathrm{KD}}$ 损失项旨在有效训练我们的 LA 子模块，以捕获全局推理信息。

4 实验

4.1 实验设置

实现细节。 我们的实验主要在Qwen-2.5系列模型（Hui等人，2024年）上进行。同时，我们采用第2.1节所述方法，从OpenR1-Math-220K中抽取95K条高质量样本来构建我们的训练集。我们在初始化Qwen-2.5时使用

方法	GSM8K		MATH-500		AMC23		AIME24		AIME25		AVG.	
	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	ReL↓
Qwen2.5-1.5B 系列												
我们的方法（基础）	80.1											
+H2O	76.1	2116	71.7	75.4 3835	128.5	55.0 6230	209.7	13.3 13921	462.2	10.0 12802	428.6	46.0 260.1
+SepLLM	77.3	2230	75.8	76.0 3890	130.3	57.5 6573	220.2	13.3 13690	452.6	13.3 12690	423.9	47.5 260.6
LightThinker	78.8	2109	69.6	77.6 4060	134.8	60.5 6312	214.6	16.7 13827	461.6	13.3 12934	434.6	50.1 263.0
INFTYTHINK	79.5	2503	89.3	78.0 4750	170.2	62.5 7605	263.5	16.7 16318	566.2	16.7 15020	501.0	50.7 318.0
Ours	**82.1**											
Qwen2.5-7B 系列												
我们的方法（基础）	89.4											
+H2O	83.5	2730	92.0	82.6 4389	147.1	77.5 6843	209.7	30.0 13274	448.9	23.3 13204	443.7	59.4 268.3
+SepLLM	84.9	2582	92.3	84.2 4244	144.3	80.0 6511	220.2	33.3 13100	484.7	23.3 12972	435.9	61.1 275.5
LightThinker	85.6	2779	90.0	84.8 4321	145.2	80.0 6790	214.6	33.3 13307	449.0	26.7 13250	443.9	62.1 268.5
INFTYTHINK	87.9	3237	117.0	86.2 5050	184.8	82.5 8061	254.3	36.7 15094	508.7	30.0 15448	530.5	64.7 319.0
Ours	**90.9**											

表1：我们的模型与基线方法在数学推理基准上的结果。

其对应的DeepSeek-R1（DeepSeek等人，2025年）蒸馏版本，然后在训练集上对其进行训练，以获得**我们的基础模型**。最后，我们的框架建立在该基础模型之上。

基线方法。 我们将本框架与两类基线方法进行比较：高效推理方法和KV缓存压缩方法。在第一类中，**LightThinker**（Zhang等人，2025年a）使用可学习的特殊标记来压缩已完成推理步骤的推理信息，从而提升大语言模型的推理效率。**INFTYTHINK**（Yan等人，2025年）将先前推理步骤中的信息压缩为简洁摘要。在第二类中，**H2O**（Zhang等人，2023）采用贪心驱逐策略，仅保留一小部分重击令牌和近期词元的KV向量，从而压缩KV缓存。**SapLLM**（Chen等人，2025年a）仅将初始、邻近和分隔标记的KV向量存入大语言模型的KV缓存。这些基线方法均使用LoRA进行训练，且可训练参数数量与我们的模型相同。

数据集与指标。 我们在七个基准测试上评估本框架，其中包括五个数学推理基准（GSM8K（Cobbe等人，2021年）、MATH-500（Hendrycks等人，2021年）、AMC23（美国数学竞赛，2023）、AIME24（美国邀请数学考试，2024年）、AIME25（美国邀请数学考试，2025年））、一个科学推理基准（GPQA Diamond（Rein等人，2024年））和一个代码推理基准（HumanEval（Chen等人，2021年））。我们使用贪心解码生成目标大语言模型的输出。我们报告三项指标，以评估各种方法的性能与效率：（1）准确率（**准确率**）表示目标模型生成正确答案的百分比；（2）词元数量（**词元**）指思维链序列的平均长度；（3）推理延迟（**ReL**）定义为每个样本的平均推理时间。此外，我们在单块NVIDIA A100 GPU上以批大小为1测量推理延迟，同时启用FlashAttention-2（Dao，2023）与BF16（Kalamkar等人，2019）。

4.2 主要结果

数学基准测试上的实验结果如表1所示。如平均值列所示，我们的框架在推理效率方面优于所有基线方法，并取得了最佳总体性能。仔细分析这些结果后，我们得出以下几项结论：

首先，我们的模型在推理速度方面显著优于基础模型，尤其是在需要更长思维链的基准测试上。在 AIME24基准测试上，与基础模型相比，我们的模型推理速度提升了 16−18%。这主要归因于相较于基础模型的 softmax 注意力模块，我们的 MAM 具有更低的计算复杂度。此外，在所有基准测试中，我们的模型性能始终优于基础模型。这些结果凸显了线性注意力机制在实现高效推理方面的巨大潜力。

其次，除 INFTYTHINK 外，基线方法与我们的模型在每个基准测试中生成的思维链序列长度相近，因为它们是在同一数据集上训练的。尽管这些基线方法的推理效率高于基础模型，但其推理性能却更差。例如，在 AIME25基准测试中，LightThinker 相较于基础模型将推理效率提升了 10–12%，但性能下降了 3–6 分。这些结果表明，朴素的 KV缓存压缩方法可能会丢失重要的推理信息，从而限制模型性能。

方法	GSM8K	MATH-500	美国数学竞赛23	美国邀请数学考试24	美国邀请数学考试25	平均值。
Ours	**82.1**	**81.2**	**67.0**	**26.7**	**23.3**	**56.2**
(1)不含 基于状态的推理策略	79.3	78.8	65.0	20.0	20.0	52.6
(2)不含 思维模式多样性	80.9	80.0	65.0	23.3	20.0	53.8
(3)不含 蒸馏损失 $\mathcal{L}_{\text{KD}}$	77.0	77.6	62.5	20.0	16.7	50.8
(4)不含 模型推理状态	76.4	76.2	62.5	20.0	16.7	46.8
(5)不含 门控机制	80.1	79.4	65.0	23.3	23.3	54.2

表 2：我们的模型在数学推理基准上的消融结果。

第三，与其他基线方法不同，INFTYTHINK 使用先前推理步骤的文本摘要来记录其推理信息。尽管这种方法增强了可解释性，但由于需要生成更长的思维链序列，它降低了推理效率。此外，这种离散的摘要方式可能会遗漏模型内部部分隐式推理信息，导致 INFTYTHINK 的性能略低于基础模型。然而，我们通过利用模型的推理状态来记录推理信息，有效避免了这两个问题，使我们的模型在效率和性能方面均优于 INFTYTHINK。

4.3 消融研究

我们进一步通过从框架中移除不同组件，开展了大量消融研究，以考察它们各自的影响。如表 2 所示，我们仅从性能角度将模型与以下变体进行比较，因为它们的推理效率相同。

不含基于状态的推理策略。 在此变体中，我们从框架中移除了基于状态的推理策略。我们观察到，该变体在所有基准测试上的平均性能下降了 3.54 个百分点。这些结果表明，我们的推理策略确实能够增强大语言模型应对噪声推理步骤的鲁棒性，从而提升其总体性能。此外，这也说明，经过适当处理后，我们的模型状态能够直观地反映推理步骤的质量，这为未来的探索（例如过程奖励强化学习）提供了有趣的启示。

不使用思维模式多样性。 我们通过相应的特殊标记，引导大语言模型在相邻的推理步骤中使用不同的思维模式，从而确保推理过程中思维模式的多样性。为检验这一做法的有效性，我们从框架中移除了该操作。如**第 (2) 行**所示，该变体在所有基准测试上的性能均持续下降。这些发现表明，更多样的思维模式能够增强大语言模型的探索能力，从而提高其生成答案的准确率。

蒸馏损失 $\mathcal{L}_{\text{KD}}$。 如第 3.3 节所述，我们使用 $\mathcal{L}_{\text{KD}}$ 训练 LA 子模块，使其能够更好地捕获全局推理信息。在该变体中，训练目标不包含 $\mathcal{L}_{\text{KD}}$ 。从表 2 中的**第 (3) 行** 可以看出，该变体在所有基准测试上的性能均下降了 3.6–6.6 个百分点。主要原因是，自回归损失 $\mathcal{L}_{\text{AR}}$ 可能难以训练 LA 子模块，使其捕获推理步骤之间的全局关联。

不使用模型推理状态。在这一变体中，我们将框架中的模型状态替换为大小为 64 的滑动窗口，以存储历史推理信息。尽管这种方法也兼容大语言模型中原有的 Softmax 注意力，但与其他变体相比，该变体的性能下降最为显著。这主要是因为这种方法的推理能力不如线性注意力强。

不使用门控机制。这一变体从 LA 子模块中移除了门控机制，并直接将 MAM 中两个子模块的输出相加。这一改动导致所有基准测试上的性能略有下降。这证实，我们的门控机制能够更精确地控制各词元所需的历史推理信息量。

4.4 进一步分析

计算与内存开销分析。我们通过实验进一步比较了模型与基础模型在不同思维链长度下的计算效率和内存效率。实验结果如图 4(a) 所示。虽然在思维链较短时，我们的模型与基础模型的推理效率相近，但当

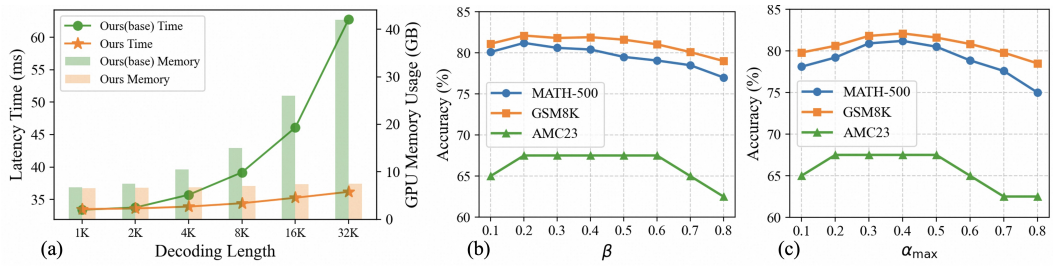


图4：(a) 展示了我们的模型和基础模型的计算效率与内存效率。(b) 和 (c) 分别展示了我们的模型在超参数 β 和 $\alpha_{\max}$ 取不同值时的性能。这些实验均在 Qwen2.5-1.5B 上进行。

思维链长度超过 4K 后，我们的模型便显著优于基础模型。具体而言，当思维链长度达到 32K 时，我们的模型推理速度比基础模型快 40% 以上。此外，我们的模型在不同思维链长度下的内存使用量几乎保持不变，而基础模型的内存使用量则随思维链长度线性增长。从理论上讲，当禁用 FlashAttention-2 时，我们的模型在计算效率和内存效率方面的优势会更加显著。

超参数分析。我们还研究了两个关键超参数 β 和 $\alpha_{\max}$ 对模型性能的影响。如图4(b)–(c)所示，我们的模型对这两个超参数不太敏感。同时，当 β 和 $\alpha_{\max}$ 分别设为 0.2 和 0.4 时，模型性能最佳。我们进一步分析这两个超参数取值的选择如下：

- β 表示蒸馏损失项 $\mathcal{L}_{\text{KD}}$ 的权重。将 β 设得过小（例如 0.1）会使我们的线性注意力模块无法有效捕获全局推理信息；而设得过大（例如 0.8）则会限制模型超越基础模型的潜力。

- 在基于状态的推理策略中，α 决定了对噪声推理步骤的推理方向进行修正的幅度。因此，将 α 设得过低（例如，0.1）会增加模型对噪声推理步骤的敏感度，而设得过高（例如，0.8）则会限制大语言模型的探索能力。

- 此外，在适当的范围内（例如，0.2−0.6），模型的性能对这些超参数并不敏感。

更多基线方法的分析。近期，许多研究（Cheng 等，2025年；Wu 等，2025年；Aggarwal 和 Welleck，2025年；Li 等，2025年a；Fatemi 等，2025年；Li 等，2025年b；Lan 等，2025年b；Lin 等，2025年；Yuan 等，2025年；Dai 等，2025年a；Xiang 等，2025年；Qi 等，2025年a；Zhang 等，2025年b；Wang 等，2025年a；Yong 等，2025年；Zhao 等，2025年；Chen 等，2025年c；Qi 等，2025年b；Yang 等，2025年b）致力于缩短思维链长度，以提高大语言模型的推理效率。然而，我们的方法旨在降低大语言模型中注意力机制的计算复杂度和存储复杂度，从而加快令牌生成速度。这表明，无论思维链长度如何，我们的方法都能提高大语言模型的推理效率。因此，这些基线方法与我们的方法相互正交且兼容。这里，我们选取四种开源基线方法作为示例，以验证我们的方法与这些基线方法的兼容性：基于任务算术的模型合并（**TAMM**）（Wu 等，2025年）、**SBT-D**（Zhao 等，2025年）、**L1-Max**（Aggarwal 和 Welleck，2025年），以及 Anytime-Reasoning（**AR**）（Qi 等，2025年b）。如表3所示，我们的方法通过加快大语言模型的令牌生成速度，进一步提升了这些基线方法的推理效率。此外，基于强化学习的基线方法（例如 AR）能为我们的方法提供更强的基础模型，从而进一步提高其性能上限。

在附录 A 中，我们进一步分析了框架的域泛化能力，以及 LA 子模块中的门控机制。**在附录 C 中**，我们对框架作出更直观的解释，并进一步探讨其潜力。

方法	GSM8K		MATH-500		AMC23		AIME24		AIME25		AVG.	
	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	ReL↓
Qwen2.5-1.5B 系列												
我们的方法（基础）	80.1											
+TAMM	78.0	1212	43.4	73.2	2265	82.1	55.0	2924	105.6	16.7	5702	215.8
+SBT-D	77.8	1170	43.1	72.8	2105	76.7	57.5	3215	114.9	16.7	6409	246.4
Ours	82.1											
+TAMM	82.1	2116	69.7	81.2	3812	125.8	67.5	6460	213.2	26.7	13610	440.13
+SBT-D	79.9	1081	36.2	76.8	2110	70.3	62.5	3193	101.4	23.3	6458	212.6
令牌预算	2k		4k		6k		13k		12k			
基础版+L1-Max	80.5	2077	77.4	79.0	4091	144.8	62.5	6103	233.5	20.0	13108	514.5
基础版+AR	79.8	2061	75.8	79.4	4083	143.6	62.5	6097	232.0	23.3	13108	514.5
我们的方法+L1-Max	82.7	2103	69.5	81.4	3962	129.1	65.0	6025	199.2	26.7	12980	415.2
我们的方法+AR	82.7	2084	68.6	82.0	4079	131.2	67.5	6101	200.5	30.0	12980	415.2
令牌预算	1k		2k		3k		6.5k		6k			
基础版+L1-Max	77.8	1161	41.6	76.8	2171	80.3	57.5	3206	124.6	16.7	6606	260.7
基础版+AR	78.8	1105	39.8	78.2	2075	77.8	60.0	3112	124.7	20.0	6587	260.3
我们的方法+L1-Max	79.4	1130	36.3	79.2	2066	68.6	62.5	3147	103.1	23.3	6691	208.0
我们的方法+AR	81.9	1092	35.1	80.4	2107	69.1	65.0	3107	101.6	26.7	6603	207.4

表3：我们的方法与基线方法结合后在数学推理基准上的结果。

5 相关工作

高效推理。尽管较长的思维链增强了大语言模型的推理能力，但也带来了显著的计算开销。为缓解这一挑战，现有的思维链压缩方法大致可分为三类。**第一类**利用提示（Han等人，2024年；Ma等人，2025年a）、监督微调（SFT）（Liu等人，2024年；Munkhbat等人，2025年）和强化学习（RL）（Aggarwal等人，2025年；Shen等人，2025年；Dai等人，2025年b），鼓励大语言模型生成更短的思维链序列。由于此类思维链缩减与测试时扩展相冲突（OpenAI等人，2024年），这些方法往往会损害大语言模型的推理能力（Jin等人，2024年；Merrill等人，2024年）。**第二类**将多个文本标记中的信息压缩到单个隐藏向量中，把较长的离散思维链序列转换为较短的连续序列，以提高大语言模型的推理效率（Hao等人，2024年；Su等人，2025年）。然而，此类方法通常需要改变大语言模型的输入/输出模式，这会增加其发生灾难性遗忘的风险，并限制其性能。**第三类**用更精简的文本表达思维链，在缩短篇幅的同时保留大语言模型的推理能力（Ma等人，2025年b；Kang等人，2025年；Xia等人，2025年）。不过，在精简较长思维链时，这些方法可能会丢失关键信息，或降低可解释性（Wang等人，2025年c）。

线性注意力。 为了解决 Softmax 注意力的二次计算成本（Zhang et al., 2018; 2022），研究者提出了许多线性注意力方法，以提高训练和推理效率。早期研究探讨了用于线性注意力的核函数（Kasai et al., 2021; Peng et al., 2021）。随后，一些研究将门控机制引入线性注意力，以更有效地控制模型状态中的信息（Yang et al., 2024a; Qin et al., 2024; Lan et al., 2025a）。近期研究为线性注意力的测试时训练属性提供了理论证据（Yang et al., 2024b; Sun et al., 2024; Liu et al., 2025; Chen 等人，2025b）。这些研究致力于探索线性注意力的在线学习目标，以增强其序列建模能力。

6 结论

本文提出了一种高效推理框架，将大语言模型的推理过程建模为状态转移过程，以提升其推理效率。在该框架中，我们设计了一个 MAM，用于替代大语言模型中的原始注意力模块。MAM 采用线性注意力机制，捕获存储先前推理步骤中推理信息的模型状态。利用模型状态，我们可以显著降低注意力的计算复杂度和内存开销，从而提高大语言模型的推理效率。此外，我们设计了一种基于状态的推理策略，以避免由噪声推理步骤引起的过度思考问题。在多个基准测试和不同模型规模上的大量实验表明，我们的框架具有高效性和稳健性。

致谢

本项目获得了中国国家重点研发计划（编号：2022ZD0160501）、中国福建省自然科学基金（编号：2024J011001）和厦门市公共技术服务平台项目（编号：3502Z20231043）的支持。我们还感谢审稿人提出的富有洞察力的意见。

伦理声明

本研究遵循 ICLR 道德准则。本研究未涉及人体受试者或动物实验。所有使用的数据集，包括 OpenR1-Math-220K、GSM8K、MATH-500、AMC23、AIME24、AIME25、GPQA Diamond 和 HumanEval，均按照相关使用规范获取，确保未侵犯隐私。我们在研究过程中谨慎避免偏见或歧视性结果。研究未使用任何个人身份信息，也未开展可能引发隐私或安全问题的实验。我们致力于在整个研究过程中保持透明与诚信。

可复现性声明

我们已尽一切努力确保本文所述结果可复现。论文详细介绍了实验设置，包括训练步骤、模型配置和硬件细节。此外，我们使用 OpenR1-Math-220K、GSM8K、MATH-500 和 HumanEval 等公开数据集，以确保评估结果一致且可复现。我们相信，这些措施将使其他研究人员能够复现我们的工作，并进一步推动该领域的发展。

大语言模型的使用

在撰写本文的过程中，我们仅使用大语言模型（LLM）辅助写作和润色文本，包括提升表达清晰度、语法正确性和可读性。我们未使用大语言模型生成科学内容、设计实验、进行分析或得出结论。本文报告的所有技术构想、实验和结果均完全由作者完成。

参考文献

Pranjal Aggarwal 和 Sean Welleck。L1：通过强化学习控制推理模型的思考时长。载于*COLM* 论文集，2025年。

Pranjal Aggarwal、Sean Welleck、Pranjal Aggarwal 和 Sean Welleck。L1：控制 ar理模型通过强化学习进行思考的时长。*arXiv:2503.04697*，2025年。

美国邀请数学考试。美国邀请数学考试（美国邀请数学考试）2024年第一场和第二场，2023。网址 https://huggingface.co/datasets/Maxwell-Jia/AIME_2024。

美国邀请数学考试。美国邀请数学考试（美国邀请数学考试）2025年第一场和第二场，2025年。网址 https: //huggingface.co/datasets/opencompass/AIME2025。

美国数学竞赛。美国数学竞赛，2023。网址 https://artofproblemsolving.com/wiki/index.php/AMC_Problems_and_Solutions。

Guoxuan Chen、Han Shi、Jiawei Li、Yihang Gao、Xiaozhe Ren、Yimeng Chen、Xin Jiang、Zhenguo Li、Weiyang Liu 和 Chao Huang。SepLLM：通过将一个片段压缩为一个分隔符来加速大语言模型。在 *ICML* 会议论文集 中，2025年a。

Mark Chen、Jerry Tworek、Heewoo Jun、Qiming Yuan、Henrique Ponde De Oliveira Pinto、Jared Kaplan、Harri Edwards、Yuri Burda、Nicholas Joseph、Greg Brockman 等。评估基于代码训练的大语言模型。*arXiv:2107.03374*，2021年。

Xingyu Chen、Jiahao Xu、Tian Liang、Zhiwei He、Jianhui Pang、Dian Yu、Linfeng Song、Qiuzhi Liu、Mengfei Zhou、Zhuosheng Zhang 等。别对 2+ 3= 想太多? 论类 o1 大语言模型的过度思考。载于*ICML* 论文集，2025年b。Zigeng Chen、Xinyin Ma、Gongfan Fang、Ruonan Yu 和 Xinchao Wang。VeriThinker：通过学习验证提升推理模型效率。*arXiv:2505.17941*，2025c。Xiaoxue Cheng、Junyi Li、Zhenduo Zhang、Xinyu Tang、Wayne Xin Zhao、Xinyu Kong 和 Zhiqiang Zhang。激励双过程思维，实现高效大语言模型推理。*arXiv:2505.16315*，2025年。Karl Cobbe、Vineet Kosaraju、Mohammad Bavarian、Mark Chen、Heewoo Jun、Lukasz Kaiser、Matthias Plappert、Jerry Tworek、Jacob Hilton、Reiichiro Nakano 等。训练验证器以求解数学应用题。*arXiv:2110.14168*，2021年。Muzhi Dai、Shixuan Liu 和 Qingyi Si。面向高效推理的稳定强化学习。*arXiv:2505.18086*，2025年a。Muzhi Dai、Chenxu Yang 和 Qingyi Si。S-GRPO：通过强化学习在推理模型中实现早退。*arXiv:2505.07686*，2025年b。Tri Dao。FlashAttention-2：借助更优的并行化与工作划分，实现更快的注意力计算。*arXiv:2307.08691*，2023。DeepSeek、Daya Guo、Dejian Yang、Haowei Zhang、Junxiao Song、Ruoyu Zhang、Runxin Xu、Qihao Zhu、Shirong Ma、Peiyi Wang、Xiao Bi 等。DeepSeek-R1：通过强化学习激发大语言模型的推理能力。*arXiv:2501.12948*，2025年。Mehdi Fatemi、Banafsheh Rafiee、Mingjie Tang 和 Kartik Talamadupula。通过强化学习实现简洁推理。*arXiv:2504.05185*，2025年。Tingxu Han、Zhenting Wang、Chunrong Fang、Shiyu Zhao、Shiqing Ma 和 Zhenyu Chen。具备词元预算意识的大语言模型推理。*arXiv:2412.18547*，2024年。Shibo Hao、Sainbayar Sukhbaatar、DiJia Su、Xian Li、Zhiting Hu、Jason Weston 和 Yuandong Tian。训练大语言模型在连续潜在空间中进行推理。*arXiv:2412.06769*，2024年。Dan Hendrycks、Collin Burns、Saurav Kadavath、Akul Arora、Steven Basart、Eric Tang、Dawn Song 和 Jacob Steinhardt。使用MATH数据集衡量数学问题求解能力。载于*NIPS* 论文集，2021年。Edward J Hu、Yelong Shen、Phillip Wallis、Zeyuan Allen-Zhu、Yuanzhi Li、Shean Wang、Lu Wang、Weizhu Chen 等。LoRA：大语言模型的低秩适应。载于*ICLR* 论文集，2022年。Binyuan Hui、Jian Yang、Zeyu Cui、Jiaxi Yang、Dayiheng Liu、Lei Zhang、Tianyu Liu、Jiajun Zhang、Bowen Yu、Keming Lu 等。Qwen2.5技术报告。*arXiv:2409.12186*，2024年。Mingyu Jin、Qinkai Yu、Dong Shu、Haiyan Zhao、Wenyue Hua、Yanda Meng、Yongfeng Zhang 和 Mengnan Du。推理步骤长度对大语言模型的影响。载于*ACL* 论文集，2024年。Dhiraj Kalamkar、Dheevatsa Mudigere、Naveen Mellempudi、Dipankar Das、Kunal Banerjee、Sasikanth Avancha、Dharma Teja Vooturi、Nataraj Jammalamadaka、Jianyu Huang、Hector Yuen 等。BFLOAT16在深度学习训练中的研究。*arXiv:1905.12322*，2019年。Yu Kang、Xianghui Sun、Liangyu Chen 和 Wei Zou。C3OT：在不损害效果的前提下生成更短的思维链。载于*AAAI* 论文集，2025年。Jungo Kasai、Hao Peng、Yizhe Zhang、Dani Yogatama、Gabriel Ilharco、Nikolaos Pappas、Yi Mao、Weizhu Chen 和 Noah A Smith。将预训练Transformer微调为循环神经网络。载于*EMNLP* 论文集，2021年。

Angelos Katharopoulos、Apoorv Vyas、Nikolaos Pappas 和 Franc¸ois Fleuret。Transformer 是 RNN：采用线性注意力的快速自回归 Transformer。在 *ICML* 论文集，2020年。Takeshi Kojima、Shixiang Shane Gu、Machel Reid、Yutaka Matsuo 和 Yusuke Iwasawa。大语言模型是零样本推理器。在 *NIPS* 论文集，2022年。Alex Krizhevsky、Ilya Sutskever 和 Geoffrey E Hinton。使用深度卷积神经网络进行 ImageNet 分类。在 *NIPS* 论文集，2012年。Disen Lan、Weigao Sun、Jiaxi Hu、Jusen Du 和 Yu Cheng。Liger：将大语言模型线性化为门控循环结构。在 *ICML* 论文集，2025年a。Zhibin Lan、Liqiang Niu、Fandong Meng、Jie Zhou 和 Jinsong Su。UME-R1：探索推理驱动的生成式多模态嵌入。*arXiv:2511.00405*，2025年b。Chen Li、Nazhou Liu 和 Kai Yang。自适应组策略优化：迈向稳定训练和令牌高效推理。*arXiv:2503.15952*，2025年a。Junnan Li、Dongxu Li、Silvio Savarese 和 Steven Hoi。BLIP-2：使用冻结图像编码器和大语言模型进行引导式语言-图像预训练。在 *ICML* 论文集，2023。Zheng Li、Qingxiu Dong、Jingyuan Ma、Di Zhang、Kai Jia 和 Zhifang Sui。Selfbudgeter：用于高效大语言模型推理的自适应令牌分配。*arXiv:2505.11274*，2025年b。Junhong Lin、Xinyue Zeng、Jie Zhu、Song Wang、Julian Shun、Jun Wu 和 Dawei Zhou。Plan and Budget：大语言模型推理的有效高效测试时扩展。*arXiv:2505.16122*，2025年。Bo Liu、Rui Wang、Lemeng Wu、Yihao Feng、Peter Stone 和 Qiang Liu。Longhorn：状态空间模型是摊销在线学习器。在 *ICLR* 论文集，2025年。Tengxiao Liu、Qipeng Guo、Xiangkun Hu、Cheng Jiayang、Yue Zhang、Xipeng Qiu 和 Zheng Zhang。语言模型能学会跳过步骤吗？在 *NIPS* 论文集，2024年。Wenjie Ma、Jingxuan He、Charlie Snell、Tyler Griggs、Sewon Min 和 Matei Zaharia。推理模型无需思考也能发挥作用。*arXiv:2504.09858*，2025年a。Xinyin Ma、Guangnian Wan、Runpeng Yu、Gongfan Fang 和 Xinchao Wang。CoT-valve：长度可压缩的思维链调优。*arXiv:2502.09601*，2025年b。William Merrill、Ashish Sabharwal、William Merrill 和 Ashish Sabharwal。带思维链的 Transformer 的表达能力。在 *ICLR* 论文集，2024年。AI Meta。Llama 4 系列：原生多模态人工智能创新新时代的开端。*https://ai. meta. com/blog/llama-4-multimodal-intelligence/*，查阅于，2025年。Tergel Munkhbat、Namgyu Ho、Seo Hyun Kim、Yongjin Yang、Yujin Kim 和 Se-Young Yun。自训练引导大语言模型产生简洁推理。*arXiv:2502.20122*，2025年。Aaron OpenAI、Jaech、Adam Kalai、Adam Lerer、Adam Richardson、Ahmed El-Kishky、Aiden Low、Alec Helyar、Aleksander Madry、Alex Beutel、Alex Carney 等。OpenAI o1 系统卡。*arXiv:2412.16720*，2024年。Hao Peng、Nikolaos Pappas、Dani Yogatama、Roy Schwartz、Noah A Smith 和 Lingpeng Kong。随机特征注意力。在 *ICLR* 论文集，2021年。Penghui Qi、Zichen Liu、Tianyu Pang、Chao Du、Wee Sun Lee 和 Min Lin。通过预算相对策略优化优化随时推理。*arXiv:2505.13438*，2025年a。Penghui Qi、Zichen Liu、Tianyu Pang、Chao Du、Wee Sun Lee 和 Min Lin。通过预算相对策略优化优化随时推理。*arXiv:2505.13438*，2025年b。Zhen Qin、Songlin Yang、Weixuan Sun、Xuyang Shen、Dong Li、Weigao Sun 和 Yiran Zhong。HGRN2：带状态扩展的门控线性循环神经网络。在 *COLM* 论文集，2024年。

David Rein、Betty Li Hou、Asa Cooper Stickland、Jackson Petty、Richard Yuanzhe Pang、Julien Di- rani、Julian Michael 和 Samuel R Bowman。Gpqa：研究生水平的、可通过 Google 验证的问题与答案基准。在*COLM* 论文集，2024年。Yi Shen、Jian Zhang、Jieyun Huang、Shuming Shi、Wenjing Zhang、Jiangze Yan、Ning Wang、Kai Wang、Zhaoxiang Liu 和 Shiguo Lian。Dast：面向大规模推理模型的难度自适应慢思考。*arXiv:2503.04472*，2025年。DiJia Su、Hanlin Zhu、Yingchen Xu、Jiantao Jiao、Yuandong Tian 和 Qinqing Zheng。令牌分类：混合潜在令牌和文本令牌以改进语言模型推理。*arXiv:2502.03275*，2025年。Yu Sun、Xinhao Li、Karan Dalal、Jiarui Xu、Arjun Vikram、Genghan Zhang、Yann Dubois、Xinlei Chen、Xiaolong Wang、Sanmi Koyejo 等。在测试时学习（学习如何学习）：具有表达力强的隐藏状态的循环神经网络。*arXiv:2407.04620*，2024年。

Yutao Sun、Li Dong、Shaohan Huang、Shuming Ma、Yuqing Xia、Jilong Xue、Jianyong Wang 和 Furu Wei。保持网络：面向大语言模型的 Transformer 后继者。*arXiv:2307.08621*，2023。

Qwen 团队。QwQ：深入思考未知的边界。2024年。

Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。Attention Is All You Need。载于 *NIPS* 论文集，2017a。

Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。Attention Is All You Need。载于 *NIPS* 论文集，2017b。

Ante Wang、Linfeng Song、Ye Tian、Baolin Peng、Lifeng Jin、Haitao Mi、Jinsong Su 和 Dong Yu。自洽性提升数学推理的校准效果。*arXiv* 预印本 *arXiv:2403.09849*，2024年。

Ante Wang、Linfeng Song、Ye Tian、Baolin Peng、Dian Yu、Haitao Mi、Jinsong Su 和 Dong Yu。LiteSearch：采用动态探索预算进行数学推理的高效树搜索。收录于*AAAI* 论文集，2025年a。

Shenzhi Wang、Le Yu、Chang Gao、Chujie Zheng、Shixuan Liu、Rui Lu、Kai Dang、Xionghui Chen、Jianxin Yang、Zhenru Zhang 等。超越 80/20 法则：高熵少数令牌推动大语言模型推理的有效强化学习。*arXiv:2506.01939*，2025年b。

Xuezhi Wang、Jason Wei、Dale Schuurmans、Quoc Le、Ed Chi、Sharan Narang、Aakanksha Chowdh- ery 和 Denny Zhou。自洽性提升语言模型中的思维链推理。收录于*ICLR* 论文集，2023。

Yibo Wang、Li Shen、Huanjin Yao、Tiansheng Huang、Rui Liu、Naiqiang Tan、Jiaxing Huang、Kai Zhang 和 Dacheng Tao。《R1-Compress：通过分块压缩与搜索实现长链式思维压缩》。*arXiv:2505.16838*，2025c。Yue Wang、Qiuzhi Liu、Jiahao Xu、Tian Liang、Xingyu Chen、Zhiwei He、Linfeng Song、Dian Yu、Juntao Li、Zhuosheng Zhang 等。《思绪纷乱：论类 o1 大语言模型的思考不足》。*arXiv:2501.18585*，2025d。Jason Wei、Xuezhi Wang、Dale Schuurmans、Maarten Bosma、Fei Xia、Ed Chi、Quoc V Le、Denny Zhou 等。《思维链提示能够激发大语言模型的推理能力》。载于*NIPS* 论文集，2022年。Han Wu、Yuxuan Yao、Shuqi Liu、Zehua Liu、Xiaojin Fu、Xiongwei Han、Xing Li、Hui-Ling Zhen、Tao Zhong 和 Mingxuan Yuan。《通过模型合并解锁高效的大语言模型由长到短的推理》。*arXiv* 预印本 *arXiv:2503.20641*，2025年。Heming Xia、Chak Tou Leong、Wenjie Wang、Yongqi Li 和 Wenjie Li。《TokenSkip：大语言模型中的可控思维链压缩》。*arXiv:2502.12067*，2025年。Violet Xiang、Chase Blagden、Rafael Rafailov、Nathan Lile、Sang Truong、Chelsea Finn 和 Nick Haber。《恰到好处的思考：利用自适应长度惩罚强化学习实现高效推理》。*arXiv* 预印本 *arXiv:2506.05256*，2025年。

Yuchen Yan、Yongliang Shen、Yang Liu、Jin Jiang、Mengdi Zhang、Jian Shao 和 Yueting Zhuang。《INFTYTHINK：突破大语言模型长上下文推理的长度限制》。*arXiv:2503.06692*，2025年。An Yang、Anfeng Li、Baosong Yang、Beichen Zhang、Binyuan Hui、Bo Zheng、Bowen Yu、Chang Gao、Chengen Huang、Chenxu Lv 等。《Qwen3 技术报告》。*arXiv:2505.09388*，2025年a。Chenxu Yang、Qingyi Si、Mz Dai、Dingyu Yao、Mingyu Zheng、Minghui Chen、Zheng Lin 和 Weiping Wang。《测试时提示干预》。*arXiv* 预印本 *arXiv:2508.02511*，2025年b。Chenxu Yang、Qingyi Si、Yongjie Duan、Zheliang Zhu、Chenyu Zhu、Qiaowei Li、Zheng Lin、Li Cao 和 Weiping Wang。《推理模型的动态提前退出》。*arXiv:2504.15895*，2025c。Songlin Yang、Bailin Wang、Yikang Shen、Rameswar Panda 和 Yoon Kim。《采用硬件高效训练的门控线性注意力变换器》。载于*ICML* 论文集，2024a。Songlin Yang、Bailin Wang、Yu Zhang、Yikang Shen 和 Yoon Kim。《利用增量规则沿序列长度并行化线性变换器》。载于*NIPS* 论文集，2024b。Songlin Yang、Jan Kautz 和 Ali Hatamizadeh。《门控增量网络：利用增量规则改进 mamba2》。载于*ICLR* 论文集，2025d。

Shunyu Yao、Dian Yu、Jeffrey Zhao、Izhak Shafran、Tom Griffiths、Yuan Cao 和 Karthik Narasimhan。思维树：利用大语言模型进行审慎问题求解。见 *NIPS* 论文集，2023。

Xixian Yong、Xiao Zhou、Yingying Zhang、Jinlin Li、Yefeng Zheng 和 Xian Wu。思考还是不思考？从信息论视角探究大型推理模型中的思考效率。*arXiv:2505.18237*，2025年。

Hang Yuan、Bin Yu、Haotian Li、Shijun Yang、Christina Dan Wang、Zhou Yu、Xueyin Xu、Weizhen Qi 和 Kai Chen。并非所有词元都是思考中所需的。*arXiv:2505.17827*，2025年。

Biao Zhang、Deyi Xiong 和 Jinsong Su。通过平均注意力网络加速神经Transformer。见*ACL* 论文集，2018。

Biao Zhang、Deyi Xiong、Yubin Ge、Junfeng Yao、Hao Yue 和 Jinsong Su。《Aan+：用于加速神经Transformer的广义平均注意力网络》。人工智能研究期刊，2022年。

Jintian Zhang、Yuqi Zhu、Mengshu Sun、Yujie Luo、Shuofei Qiao、Lun Du、Da Zheng、Huajun Chen 和 Ningyu Zhang。《Lightthinker：逐步压缩思考》。*arXiv:2502.15589*，2025年a。Xuechen Zhang、Zijian Huang、Chenshun Ni、Ziyang Xiong、Jiasi Chen 和 Samet Oymak。《让小语言模型成为高效推理器：干预、监督、强化》。*arXiv:2505.07961*，2025年b。Zhenyu Zhang、Ying Sheng、Tianyi Zhou、Tianlong Chen、Lianmin Zheng、Ruisi Cai、Zhao Song、Yuandong Tian、Christopher R´e、Clark Barrett 等。《H2o：用于大语言模型高效生成式推理的重击者预言机》。载于*NIPS* 论文集，2023。Haoran Zhao、Yuchen Yan、Yongliang Shen、Haolei Xu、Wenqi Zhang、Kaitao Song、Jian Shao、Weiming Lu、Jun Xiao 和 Yueting Zhuang。《通过自我刹车调优，让大语言模型摆脱过度思考》。*arXiv:2505.14604*，2025年。Denny Zhou、Nathanael Sch¨arli、Le Hou、Jason Wei、Nathan Scales、Xuezhi Wang、Dale Schuurmans、Claire Cui、Olivier Bousquet、Quoc Le 等。《从少到多提示助力大语言模型实现复杂推理》。载于*ICLR* 论文集，2023。Rui-Jie Zhu、Tianhao Peng、Tianhao Cheng、Xingwei Qu、Jinfa Huang、Dawei Zhu、Hao Wang、Kai-wen Xue、Xuanliang Zhang、Yong Shan 等。《潜在推理综述》。*arXiv:2507.06203*，2025年。

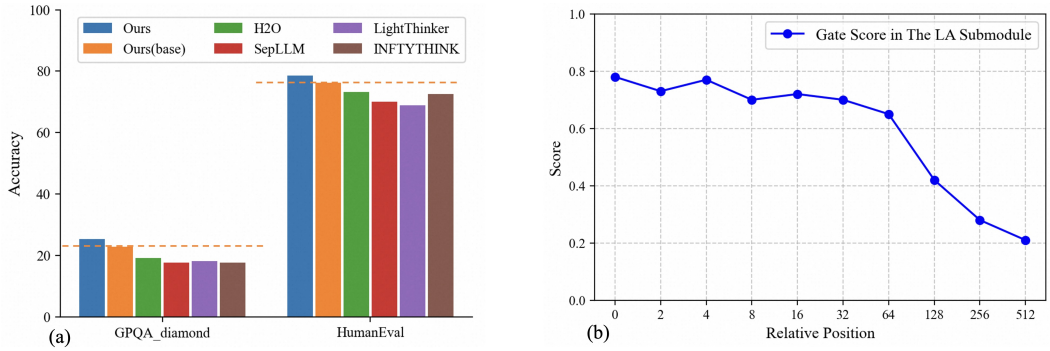


图 5：(a) 展示了我们的模型与基线方法在 GPQA Diamond 和 HumanEval 基准测试上的表现。(b) 展示了 LA 子模块中每个推理步骤内不同位置词元的平均门控分数。这些实验均在 Qwen2.5-7B 上进行。

A 进一步分析

域泛化分析。如第 4.1 节所述，我们的训练集仅包含数学推理数据。为考察框架的域泛化能力，我们进一步在科学推理基准 GPQA Diamond 和代码推理基准 HumanEval 上测试模型。如图 5(a) 所示，我们的模型在这两个基准测试中始终优于所有基线方法。这些结果揭示了我们框架的几项优势：（1）我们的训练策略有效避免了目标领域训练对基础模型在其他领域表现造成的负面影响；（2）我们的框架能够有效地将目标领域中的性能和效率优势迁移到其他领域。

门控分数分析。在此，我们进一步探究每个推理步骤中不同位置的词元需要多少历史推理信息。为此，我们计算了 LA 子模块中这些词元的平均门控分数。如图 5(b) 所示，推理步骤中较早出现的词元通常比之后出现的词元更需要来自模型状态的历史推理信息。主要原因是，后续词元可以通过 SA 子层直接从较早的词元获取更多信息，因此对模型状态的依赖较低。

B 实现细节

我们在训练集上将模型和基线方法均训练 2 轮，批量大小为 32。学习率设为 $2e{-}4$，预热比例为 0.03。学习率采用余弦衰减策略，逐渐降至零。我们将模型和基线方法在 Qwen2.5-1.5B 上的可学习 LoRA 参数量设为约 66.5M，在 Qwen2.5-7B 上设为约 147.3M。为控制训练成本，我们将训练集中长思维链样本的最大长度限制为 16,384 个词元。同时，我们使用 8 块 NVIDIA A100 GPU 训练模型和基线方法，每块显存为 80GB。更多实现细节如下：

- **首先**，为了尽量减少思维类型之间的冲突，我们采用包含 128 个簇的 K 均值聚类算法，为每个步骤标注思维类型。较多的簇可以有效降低将具有不同思维模式的推理步骤分配到同一簇的可能性。值得注意的是，鉴于推理步骤数量庞大（140 万），我们采用 MiniBatch K-Means 聚类来提高聚类效率。同时，我们使用余弦相似度作为距离度量，从而使推理步骤在各个簇中的分布更加均衡（最多：1.61 万；最少：4300）。

- **第二**，我们采用 LoRA 策略来实现线性注意力模块，其中秩 r 用于控制可学习参数的数量。具体而言，我们将 W_q、W_k、W_v 的秩r 设为 512，并将 W_g 的秩 r 设为 16。

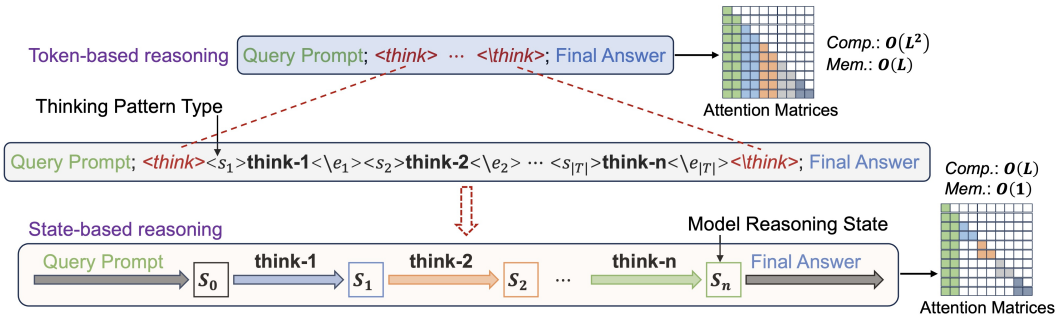


图 6：传统基于令牌的推理与基于状态的推理的比较。**思考**$_t$表示一个推理步骤。**计算量** 和 **内存** 分别表示计算复杂度和内存复杂度，其中 L 表示上下文长度。在基于状态的推理中，大语言模型仅基于查询提示和推理状态 $\boldsymbol{S}_{t-1}$即可高效生成**思考**$_t$ 。

• **第**三，在线性注意力中，模型状态矩阵的维度 $R^{d\times d}$ d $S_t{=}S_{t-1}{+}k_t^T v_t$ ∈ 对应于大语言模型中键向量和值向量的维度，而后者由模型架构本身决定。

• **第三，**由于仅训练新引入的参数（线性注意力模块和特殊标记），我们的方法的训练成本降低至基础模型的**47.7%** 。具体而言，在 8 块 NVIDIA A100 GPU 上，使用 Qwen-2.5-1.5B 模型配置时，基础模型每轮训练需要 8.6 小时，而我们的方法需要 4.1 小时。

C 我们的方法的进一步分析

在图 6 中，我们详细比较了我们的方法（基于状态的推理）与朴素大语言模型（基于令牌的推理）的推理过程。对于朴素大语言模型，我们向其提供一个查询，随后它自回归地生成完整的思维链，其中每个词元都会关注此前生成的所有词元。同时，所有已生成词元的键向量和值向量都必须存储在 KV缓存中。因此，在这种设置下，注意力计算复杂度随思维链长度呈二次增长 L （即 O（L^2）），而内存复杂度呈线性增长（即O（L））。

在我们的框架中，大语言模型逐步生成完整的思维链，以推理步骤为单位：**思考**$_1$ $\rightarrow \cdots \rightarrow$ **思考**$_n$。与此同时，我们采用线性注意力机制来估计大语言模型在推理过程中的状态 $\boldsymbol{S}$。如图 6（底部）所示，随着大语言模型推理的推进（**思考**$_1$ $\rightarrow \cdots \rightarrow$ **思考**$_n$），其状态会持续转变（$\boldsymbol{S}_0 \rightarrow \cdots \rightarrow \boldsymbol{S}_n$）。因此，在我们的框架中，大语言模型的推理过程被建模为一个状态转移过程。重要的是，我们模型的状态发挥着两个关键作用：

• **首先**，状态记录/存储已完成推理步骤的历史推理信息。因此，我们的模型仅基于当前状态（$\boldsymbol{S}_{t-1}$）和查询，就能生成下一个推理步骤（**思考**$_t$），而无需依赖任何先前已完成的步骤（**思考**$_{<t}$）。通过这种方式，我们将大语言模型中注意力的计算复杂度降低至线性， O（L）。由于后续推理不需要先前已完成的推理步骤，因此会从 KV缓存中移除这些步骤中词元的键向量和值向量。由此，大语言模型中注意力的存储复杂度降低至O(1)。

• **其次**，模型状态可以被视为对大语言模型的知识的定量表征。因此，在我们的基于状态的推理策略中，我们纠正噪声步骤的推理方向，以防止其污染大语言模型的知识（状态），从而提升推理性能。此外，我们正在研究如何利用模型状态对推理步骤的过程奖励进行无监督估计，以提高大语言模型在强化学习训练中的效率和效果。

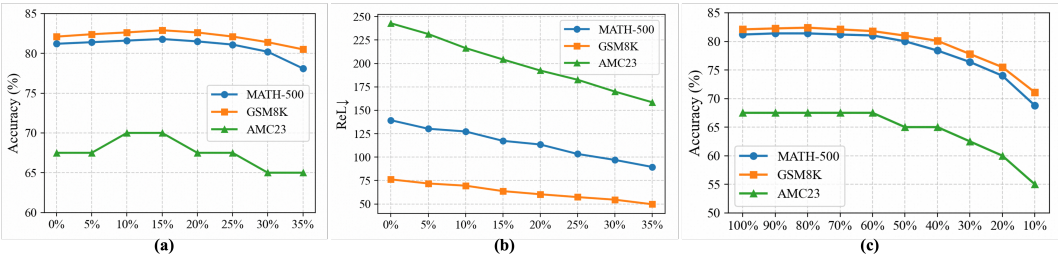


图 7：(a) 和 (b) 展示了随着移除的推理步骤比例变化，我们的模型的性能和推理延迟。(c) 展示了移除不同比例的训练样本后，我们的模型的性能。这些实验均在 Qwen2.5-1.5B 上进行。

在此，我们还开展了一项分析实验，以进一步验证模型状态的潜力。具体而言，对于训练集中的每个样本，通过计算全局推理方向 ∇ 与该步骤的推理方向 ∇_t 之间夹角的余弦值（即 $\cos(\nabla, \nabla_t)$），来估计每个推理步骤的过程奖励。接着，我们从每个样本中移除过程奖励较低的推理步骤，以缩短思维链长度。最后，我们在图 7 中展示实验结果。我们发现，移除不到 20% 的步骤后，模型的性能有所提升。这表明，这种简单的过程奖励能够反映推理步骤的质量。此外，当移除超过 25% 的步骤时，模型性能开始下降。这可能是因为思维链中的关键步骤被移除，限制了模型的推理能力。同时，缩短思维链长度进一步提升了模型的推理效率。

此外，我们进一步利用模型状态估计训练样本质量，从而提高方法的数据效率。具体而言，我们首先将某个训练样本中各推理步骤的过程奖励取平均，计算该样本的全局过程奖励。全局过程奖励反映了样本中包含噪声推理步骤的程度。随后，我们丢弃全局过程奖励较低的训练样本，并重新训练模型。如图 7(c) 所示，保留 60% 的高质量训练样本时，模型的性能并未下降。与此同时，移除 60% 的训练样本（即保留 40%）仅会导致性能略微下降。

D 思维链分段方法的泛化分析

事实上，我们的长思维链分段方法也是一个通用框架。该框架包含两项主要操作：

- **分段，** 旨在将长思维链切分为一系列推理步骤。
- **标注，** 旨在识别每个推理步骤所依据的思维类型。

与此同时，该框架遵循两项核心原则：

- **首先，** 分段后的推理步骤应保持较低的语言依赖性，以减轻 KV 缓存清理造成的信息丢失。
- **其次，** 应尽量减少思维类型之间的冲突，以避免类型不兼容带来的不利影响。

该框架的主要目标是增强对大语言模型推理过程的可控性。在此设置下，我们可以利用与思维类型相关联的特殊标记，来监控和控制大语言模型的推理过程。

在资源充足的情况下，可以手动进行思维链分段和思维类型标注，从而获得更优的性能。另一种可行的方法是采用强大的大语言模型（如 GPT-4o）来执行分段和标注。

方法	MATH-500			AMC23			AIME24		
Qwen2.5-1.5B 系列									
贪心解码	Acc↑	Tok	RC率↓	Acc↑	Tok	RC率↓	Acc↑	Tok	RC率↓
我们的方法（基础）	78.8	3958	0.31	62.5	6392	0.40	20.0	13765	0.50
Ours	81.2	3812	0.18	67.5	6460	0.23	26.7	13610	0.31
采样解码	pass@1↑	Tok	RC率↓	pass@1↑	Tok	RC率↓	pass@1↑	Tok	RC率↓
我们的方法（基础）	80.3	3780	0.19	63.6	6607	0.28	22.5	13880	0.26
Ours	83.5	3652	0.06	69.4	6533	0.08	29.1	13809	0.09

表 4：我们的模型和基线方法在不同解码策略下的结果。RC率表示在所有错误问题中，模型因重复生成而未能得到最终答案的问题所占的比例。

方法	GSM8K		MATH-500		AMC23		AIME24		AIME25		AVG.	
	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	词元 ReL↓	准确率↑	ReL↓
Qwen2.5-7B 系列($\beta = 0.4, \alpha = 0.3$)												
我们的方法												
Ours	91.6											

表5：我们的方法与基线方法在数学推理基准上的结果。

鉴于资源限制，我们采用了一种更为经济的策略：(1) 使用出现在句首的高熵转折词元进行分段，确保每个推理步骤在语义上保持独立；(2) 通过聚类算法（K-means）进行思维类型标注，同时设置较大的簇数量（128），以尽量减少类型间冲突。尽管如此，我们的模型在数学领域仍取得了出色的性能，并能有效泛化至科学领域和代码领域。

此外，我们所使用的聚类算法的具体细节如下：

- 鉴于推理步骤数量庞大（140 万），我们使用 MiniBatch K-Means 来提高聚类效率。
- 我们使用了更多的簇（128 个），以有效降低将具有不同思维模式的推理步骤归入同一簇的概率。
- 我们使用余弦相似度作为距离度量，使推理步骤在各簇之间实现均衡分布（最大值：16.1k；最小值：4.3k）。这有效降低了大语言模型对思维模式“过拟合”或“欠拟合”的风险。

E 模型性能的进一步分析

解码策略分析。正如 DeepSeek-R1 DeepSeek 等（2025）所观察到的，基于温度的采样策略可以有效减少重复生成的比例，从而提升大语言模型的推理性能。此处，我们采用了 DeepSeek-R1 采样策略（温度=0.6，top-k=0.95，运行 32 次），并在表 4 中列出了 pass@1。我们观察到，我们的模型和基础模型的性能均有所提升。其主要原因在于，这种采样方法显著降低了重复生成的比例（即，*RC*率）。为避免高估我们的模型性能并影响研究工作的可复现性，我们在主要实验中采用了更加确定性的贪心解码方法。

性能提升幅度分析。值得注意的是，我们的模型所取得的性能提升并非微不足道：

- **首先**，在 Qwen-2.5-1.5B 场景下，我们的模型相比基础模型平均提升了 4.7 个百分点。
- **其次**，尽管超参数是为 Qwen-2.5-1.5B 选择的，我们的模型在 Qwen-2.5-7B 配置下仍比基础模型平均高出 2.4 分。自

然，进行更有针对性的超参数调优将进一步提升我们的模型性能（提高 4.0 分），使其在 Qwen-2.5-7B 上表现更佳（见表5）。

- **第**三，我们的工作的主要目标是提高大语言模型的推理效率，而性能提升则是意外之喜。

F 案例研究

在图 8 和图 9 中，我们展示了两个案例研究，以说明模型的推理过程。我们注意到，这两个案例分别涉及 8 种和 9 种不同的思维类型。此外，这两个案例还共享若干思维类型：

- <**类型_6**>**：** 问题理解。
- <**类型_25**>**：** 问题解释。
- <**类型_61**>**：** 代数求解。
- <**类型_12**>**：** 解答验证。
- . . .

我们使用 GPT-4o 获得这些思维类型的解释。此外，我们观察到许多推理类型是通用的，并不局限于数学领域，例如<**类型 6**>、 <**类型 25**>和 <**类型 12**>。这也间接说明了我们框架的领域泛化能力。

Question:
Task B-1.3.\n\nA ship traveling along a river has covered $24 \\mathrm{~km}$ upstream and $28 \\mathrm{~km}$ downstream. For this journey, it took half an hour less than for traveling $30 \\mathrm{~km}$ upstream and $21 \\mathrm{~km}$ downstream, or half an hour more than for traveling $15 \\mathrm{~km}$ upstream and $42 \\mathrm{~km}$ downstream, assuming that both the ship and the river move uniformly.\n\nDetermine the speed of the ship in still water and the speed of the river.

Reasoning sequence
<think> **<type_6>** Okay, so I need to find the speed of the ship in still water and the speed of the river. Let me start by recalling that when a ship is moving upstream, its effective speed is the speed of the ship minus the speed of the river. Conversely, when moving downstream, its effective speed is the ship's speed plus the river's speed.\n\n**</type_6> <type_25>** Let me denote the speed of the ship in still water as \\(v \\) (in km/h) and the speed of the river as \\(r \\) (also in km/h). Then, the upstream speed would be \\(v - r \\), and the downstream speed would be \\(v + r \\).\n\nThe problem mentions three different journeys:\n\n1. 24 km upstream and 28 km downstream, taking half an hour less than the second journey.\n2. 30 km upstream and 21 km downstream, which takes half an hour more than the first journey.\n3. 15 km upstream and 42 km downstream, which takes half an hour less than the first journey.\n\nWait, actually, the problem states: \"For this journey, it took half an hour less than for traveling 30 km upstream and 21 km downstream, or half an hour more than for traveling 15 km upstream and 42 km downstream...\". **</type_25> <type_91>** \n\nHmm, let me parse that again. The first journey (24 km upstream, 28 km downstream) took half an hour less than the journey with 30 km upstream and 21 km downstream. Alternatively, it took half an hour more than the journey with 15 km upstream and 42 km downstream. So, there are two comparisons here:\n\n- Time for 24 up + 28 down = Time for 30 up + 21 down - 0.5 hours\n- Time for 24 up + 28 down = Time for 15 up + 42 down + 0.5 hours\n\nTherefore, we can set up two equations based on these time differences.\n\nFirst, let me express the time taken for each journey. Time is equal to distance divided by speed. **</type_91> <type_78>** So, for the first journey (24 km up, 28 km down):\n\nTime1 = \\(\\frac{24}{v - r} + \\frac{28}{v + r} \\)\n\nFor the second journey (30 km up, 21 km down):\n\nTime2 = \\(\\frac{30}{v - r} + \\frac{21}{v + r} \\)\n\nAccording to the problem, Time1 = Time2 - 0.5\n\nSimilarly, for the third journey (15 km up, 42 km down):\n\nTime3 = \\(\\frac{15}{v - r} + \\frac{42}{v + r} \\)\n\nAnd here, Time1 = Time3 + 0.5\n\nSo, we have two equations:\n\n1. \\(\\frac{24}{v - r} + \\frac{28}{v + r} = \\frac{30}{v - r} + \\frac{21}{v + r} - 0.5 \\)\n2. \\(\\frac{24}{v - r} + \\frac{28}{v + r} = \\frac{15}{v - r} + \\frac{s42}{v + r} + 0.5 \\). \n\n</**type_78**> <**type_47**> Okay, so now I need to solve these two equations for \\(v \\) and \\(r \\).\n\nLet me first work on the first equation:\n\n\\(\\frac{24}{v - r} + \\frac{28}{v + r} = \\frac{30}{v - r} + \\frac{21}{v + r} - 0.5 \\)\n\nLet me subtract \\(\\frac{24}{v - r} + \\frac{28}{v + r} \\) from both sides, but maybe it's better to bring all terms to the left side:\n\n\\(\\frac{24}{v - r} + \\frac{28}{v + r} - \\frac{30}{v - r} - \\frac{21}{v + r} + 0.5 = 0 \\)\n\nSimplify the terms:\n\nFor the upstream terms: \\(24 - 30 = -6 \\), so \\(\\frac{-6}{v - r} \\)\n\nFor the downstream terms: \\(28 - 21 = 7 \\), so \\(\\frac{7}{v + r} \\)\n\nSo, the equation becomes:\n\n\\(\\frac{-6}{v - r} + \\frac{7}{v + r} + 0.5 = 0 \\) \n\nSimilarly, for the second equation:\n\n\\(\\frac{24}{v - r} + \\frac{28}{v + r} - \\frac{15}{v - r} - \\frac{42}{v + r} - 0.5 = 0 \\)\n\nSimplify:\n\nUpstream terms: \\(24 - 15 = 9 \\), so \\(\\frac{9}{v - r} \\)\n\nDownstream terms: \\(28 - 42 = -14 \\), so \\(\\frac{-14}{v + r} \\)\n\nThus:\n\n\\(\\frac{9}{v - r} - \\frac{14}{v + r} - 0.5 = 0 \\)\n\n </**type_47**> <**type_37**> Now, we have two equations:\n\n1. \\(-\\frac{6}{v - r} + \\frac{7}{v + r} + 0.5 = 0 \\)\n2. \\(\\frac{9}{v - r} - \\frac{14}{v + r} - 0.5 = 0 \\). </type_66><type_66> \n\nLet me denote \\(x = \\frac{1}{v - r} \\) and \\(y = \\frac{1}{v + r} \\). Then, the equations become:\n\n1. \\(-6x + 7y + 0.5 = 0 \\) \u21d2 \\(-6x + 7y = -0.5 \\)\n2. \\(9x - 14y - 0.5 = 0 \\) \u21d2 \\(9x - 14y = 0.5 \\)\n\nNow, we have a system of linear equations:\n\nEquation (1): \\(-6x + 7y = -0.5 \\)\n\nEquation (2): \\(9x - 14y = 0.5 \\) </**type_47**> <**type_61**> \n\nLet me solve this system using the elimination method. Let's multiply Equation (1) by 2 to make the coefficients of y terms opposites:\n\nEquation (1) multiplied by 2: \\(-12x + 14y = -1 \\)\n\nNow, add this to Equation (2):\n\n\\((-12x + 14y) + (9x -14y) = -1 + 0.5 \\)\n\nSimplify:\n\n\\(-3x = -0.5 \\)\n\nThus, \\(x = (-0.5)/(-3) = 0.5/3 \u2248 0.1667 \\)\n\nBut exact value is \\(x = 1/6 \\)\n\nNow, substitute \\(x = 1/6 \\) back into Equation (1):\n\n\\(-6*(1/6) + 7y = -0.5 \\)\n\nSimplify:\n\n\\(-1 + 7y = -0.5 \\)\n\nAdd 1 to both sides:\n\n7y = 0.5\n\nThus, \\(y = 0.5/7 = 1/14 \u2248 0.0714 \\)\n\nSo, \\(x = 1/6 \\) and \\(y = 1/14 \\)\n\nRecall that \\(x = \\frac{1}{v - r} = 1/6 \\) \u21d2 \\(v - r = 6 \\)\n\nSimilarly, \\(y = \\frac{1}{v + r} = 1/14 \\) \u21d2 \\(v + r = 14 \\)\n\nTherefore, we have two equations:\n\n1. \\(v - r = 6 \\)\n2. \\(v + r = 14 \\)\n\nAdding these equations:\n\n\\((v - r) + (v + r) = 6 + 14 \\)\n\nSimplify:\n\n\\(2v = 20 \\) \u21d2 \\(v = 10 \\)\n\nSubstituting back into \\(v + r = 14 \\):\n\n\\(10 + r = 14 \\) \u21d2 \\(r = 4 \\). \n\nSo, the speed of the ship in still water is 10 km/h, and the speed of the river is 4 km/h. </**type_61**> <**type_12**> \n\nLet me verify these results by plugging them back into the original time equations.\n\nFirst journey: 24/(10-4) + 28/(10+4) = 24/6 + 28/14 = 4 + 2 = 6 hours.\n\nSecond journey: 30/(10-4) + 21/(10+4) = 30/6 + 21/14 = 5 + 1.5 = 6.5 hours. So, 6.5 - 0.5 = 6 hours, which matches the first journey. Good.\n\nThird journey: 15/(10-4) + 42/(10+4) = 15/6 + 42/14 = 2.5 + 3 = 5.5 hours. Then, 5.5 + 0.5 = 6 hours, which also matches the first journey. Perfect.\n\nTherefore, the solution is correct.\n\n**Final Answer**\nThe speed of the ship in still water is \\boxed{10} km/h and the speed of the river is \\boxed{4} km/h.\n </**type_12**> </think>

图 8：**案例-1** <类型 i> 和 <类型 i>分别表示与第 i种思维类型相对应的推理步骤的起始词元和结束词元。

Question:
3. (6 points) A construction company was building a tunnel. When $\\frac{1}{3}$ of the tunnel was completed at the original speed, they started using new equipment, which increased the construction speed by $20 \\%$ and reduced the working hours to $80 \\%$ of the original. As a result, it took a total of 185 days to complete the tunnel. If they had not used the new equipment and continued at the original speed, it would have taken $\\qquad$ days to complete the tunnel.

Reasoning sequence:
<think> **<type_6>** Okay, let me try to figure out this problem. So, a construction company was building a tunnel. When 1/3 of the tunnel was completed at the original speed, they started using new equipment. This new equipment increased their construction speed by 20% and reduced their working hours to 80% of the original. The total time taken with the new equipment was 185 days. We need to find out how many days it would have taken if they hadn't used the new equipment and continued at the original speed. \n\n **</type_6>** **<type_25>**Hmm. Let me break this down. Let's denote the total length of the tunnel as 1 unit for simplicity. So, they completed 1/3 of the tunnel at the original speed. Then, the remaining 2/3 was completed with the new equipment, which made them 20% faster and working 80% of the original hours per day.\n\nWait, the problem says \"reduced the working hours to 80% of the original.\" So, their daily working hours are now 80% of what they used to be. But their speed increased by 20%. So, the rate at which they construct the tunnel per day is affected both by the speed increase and the reduction in working hours.n\nFirst, let's define some variables. Let's let the original construction speed be \"v\" units per day. **</type_25>** **<type_45>** Wait, but actually, maybe it's better to think in terms of time. Let me think. Suppose the original rate is such that without any changes, they would take T days to complete the tunnel. So, total work is 1 tunnel, original time is T days. Therefore, original rate is 1/T tunnels per day.\n\nBut when they started, they completed 1/3 of the tunnel at the original speed, then switched to the new equipment for the remaining 2/3. The total time taken was 185 days. We need to find T.\n\nWait, but maybe we need to relate the original time to the actual time taken with the changes. Let me approach step by step. \n\n **</type_45>** <**type_70**> First, original total time would be T days. But they didn't complete the tunnel in T days; instead, they changed their equipment partway. Let's think about the two phases: Phase 1 is building 1/3 of the tunnel at original speed, and Phase 2 is building 2/3 at the new speed with reduced working hours.\n\nLet me define the original rate as R (tunnel per day). Then, the original time to complete the whole tunnel would be T = 1/R days.\n\nIn Phase 1, they built 1/3 of the tunnel at rate R. So, the time taken for Phase 1 is (1/3)/R = 1/(3R) days.\nIn Phase 2, they have 2/3 of the tunnel left. Their speed increases by 20%, so the new speed is R + 20% of R = 1.2R. However, their working hours are reduced to 80% of the original. If the original working hours per day are H, then the new working hours are 0.8H. But how does this affect the construction rate? \n\n **</type_70>** <**type_78**> Wait, speed is in tunnel per day, which is dependent on how much they work each day. If their speed was originally R tunnels per day with H hours per day, then their rate per hour is R/H tunnels per hour. When they increase their speed by 20%, that would be 1.2*(R/H) tunnels per hour. But they also reduce their working hours to 0.8H. Therefore, the new daily rate is 1.2*(R/H) * 0.8H = 1.2*0.8*R = 0.96R tunnels per day. **</type_78>** **<type_101>** Wait, is that correct?\n\nWait, maybe. Let me clarify. The original rate is R tunnels per day, which is based on working H hours per day. So, per hour, they complete R/H tunnels. If their speed (efficiency) increases by 20%, then their per hour rate becomes 1.2*(R/H). However, they are now working only 0.8H hours per day, so the new daily rate is 1.2*(R/H) * 0.8H = 1.2*0.8*R = 0.96R. So, their daily rate actually decreases to 0.96R? Wait, that seems counterintuitive. If they work faster but fewer hours, the net effect is a decrease in daily rate? **</type_101>** **<type_37>** Hmm.\n\nWait, but 20% faster in speed, but 20% less time working. Let me check the math. If original rate is R per day (H hours). Then per hour rate is R/H. After 20% increase, per hour rate is 1.2R/H. Then, working 0.8H hours per day, so daily rate is 1.2R/H * 0.8H = 0.96R. So yes, the daily rate becomes 0.96R. So even though they are working faster, the reduction in hours leads to a lower daily rate. Therefore, the time taken for Phase 2 is (2/3)/0.96R days.\n\nBut total time is 185 days, so:\n\nTime for Phase 1 + Time for Phase 2 = 185\n\nWhich is:\n\n1/(3R) + (2/3)/(0.96R) = 185\n\nWe can solve for R, then T = 1/R is the original total time. \n\n **</type_37>** **<type_61>** Let me compute that.\n\nFirst, let's write the equation:\n\n1/(3R) + (2/3)/(0.96R) = 185\n\nCombine the terms:\n\n[1/3 + (2/3)/0.96] / R = 185\n\nCalculate the numerator:\n\n1/3 + (2/3)/0.96\n\nFirst compute (2/3)/0.96:\n\n2/3 divided by 0.96 = (2/3)*(100/96) = (2/3)*(25/24) = (50/72) = 25/36 \u2248 0.6944\n\nThen add 1/3: 1/3 \u2248 0.3333, so total \u2248 0.6944 + 0.3333 \u2248 1.0277\n\nSo numerator \u2248 1.0277\n\nTherefore, 1.0277 / R = 185\n\nSo R \u2248 1.0277 / 185 \u2248 0.005555 tunnels per day\n\nTherefore, original time T = 1/R \u2248 1 / 0.005555 \u2248 180 days. \n\nWait, so is T 180 days? Let me check the exact calculation instead of approximating.\n\nLet me compute 1/3 + (2/3)/0.96 exactly.\n\nFirst, (2/3)/0.96:\n\n0.96 is 24/25, so dividing by 24/25 is multiplying by 25/24.\n\nThus, (2/3)*(25/24) = (50/72) = 25/36.\n\nThen 1/3 is 12/36, so 12/36 + 25/36 = 37/36.\n\nTherefore, numerator is 37/36.\n\nSo the equation is (37/36)/R = 185\n\nHence, R = 37/(36*185) = 37/(6660) = 1/180 per day.\n\nTherefore, T = 1/R = 180 days.\n\nSo the original time would be 180 days. **</type_61>** **<type_12>** Wait, so the answer is 180 days? Let me confirm.\n\nIf original time is 180 days, then original rate is 1/180 per day.\n\nPhase 1: 1/3 of the tunnel at 1/180 per day. Time = (1/3)/(1/180) = 60 days.\n\nPhase 2: 2/3 of the tunnel at the new rate. The new rate is 0.96*(1/180) = 0.96/180 = 0.005333... per day.\n\nTime for Phase 2: (2/3)/0.005333... = (2/3)/(0.005333...) \u2248 (2/3)/ (8/1500)) Because 0.005333 is 8/1500. \n\nWait, 0.005333... is 8/1500? Let me check. 8 divided by 1500 is 0.005333..., yes.\n\nSo (2/3) divided by (8/1500) = (2/3)*(1500/8) = (3000/24) = 125 days.\n\nWait, 60 days + 125 days = 185 days, which matches the given total time. Therefore, the original time T is indeed 180 days. So the answer is 180.\n\nTherefore, if they had not used the new equipment, it would have taken 180 days.\n\n**Final Answer**\n\\boxed{180}\n **</type_12>** </think>

图 9：**案例-2：** <类型 i>_和 <类型 i> 分别表示与第 i 种思维类型对应的推理步骤的开始和结束词元。