腾讯2面：RL训练为什么会“熵崩溃”?

ttttt DASOU 2025年10月13日 10:03 北京

DASOU
一名算法工程师，分享工作日常和AI干货，专注深度学习。
129篇原创内容

公众号

作者：ttttt；仅用于学术分享；

引自/编辑：丁师兄大模型

https://zhuanlan.zhihu.com/p/1954330684970754139

前两篇文章我们讲了 KL 散度和 GRPO，这篇文章讲一个最近（并非最近，鸽了一段时间了）比较火的概念——熵 (Entropy)。

很多同学只知道 RL 能 work，但为什么 work，模型在 RL 过程中到底学到了什么，似乎总隔着一层纱。

最近的两篇论文《The Entropy Mechanism of Reinforcement Learning for Reasoning Language Models》和《Beyond the 80/20 Rule: High-Entropy Minority Tokens Drive Effective Reinforcement Learning for LLM Reasoning》从“熵”这个统一的视角，探索了 RL 的内在机制。

```
https://arxiv.org/abs/2505.22617
https://arxiv.org/abs/2506.01939
```

这篇文章的目标，就是用大白话和必要的公式，讲一下“熵”到“RL”的探索之路，感受一下 RL 在确定性和不确定性之间的极限拉扯。

01 熵、交叉熵、KL 散度“三位一体”

1. 信息熵（Information Entropy）：衡量“不确定性”的标尺

信息论里的熵，概念其实非常直观。它衡量的是一个系统或一个概率分布的不确定性或者说“惊奇程度”。

举个最经典的例子：扔硬币。

场景 A：一枚极不均匀的硬币，99% 的概率是正面，1% 是反面。你来猜结果，基本无脑猜正面就行。结果揭晓时，你基本不会感到“惊奇”，因为结果非常“确定”。这个系统的不确定性很低，熵就低。

场景 B：一枚完美的均匀硬币，50% 正面，50% 反面。你猜正面还是反面？完全是蒙。结果揭晓时，你会感到更“惊奇”，因为结果非常“不确定”。这个系统的不确定性很高，熵就高。

信息论之父香农（Shannon）用一个优美的公式量化了它。对于一个离散变量X，其概率分布为 P（x），那么它的信息熵 H（P）定义为：

$$H(P) = -\sum_x P(x) \log_2 P(x)$$

- $P(x)$ 是事件 x 发生的概率。
- $\log_2 P(x)$ 可以理解为事件 x 发生所包含的“信息量”或“惊奇程度”（概率越低，信息量越大）。
- 前面的负号是了确保熵值为正。
- 整个公式就是对所有可能事件的“期望信息量”或“期望惊奇程度”进行加权平均。

对于 LLM 来说，在生成第 t 个词时，它实际上是面对一个巨大的概率分布——词汇表里每个词成为下一个词的概率。

如果模型对下一个词非常确定（例如，“一言为定，驷马难……”后面几乎必然是“追”），那么这个概率分布的熵就很低。

如果模型很纠结（例如，一个故事的开头），多个词都有可能，那么熵就很高。

2. 交叉熵（Cross-Entropy）与 KL散度（KL Divergence）：从“理想”到“现实”的代价

既然熵衡量了不确定性，那跟 LLM 的训练有什么关系?

在 LLM 训练中，我们有一个“真实世界的分布”（用 P 表示），也就是人类语言中，给定上文后，下一个词的真实概率分布。

同时，我们还有一个“我们的模型通过训练拟合出来的分布”（用 Q 表示），也就是 LLM 自己预测的、对下一个词的概率分布。

我们的目标，就是让模型 Q 尽可能地去逼近真实世界 P。

怎么衡量这个“逼近”程度呢?

交叉熵（Cross-Entropy）就登场了。它衡量的是，当我们用模型的“有偏认知”（Q）去预测和编码“客观事实”（P）时，所需要的平均信息量。

它的公式是:

$$H(P,Q) = -\sum_x P(x) \log_2 Q(x)$$

可以看到它和熵的公式很像，只是 log 里的概率从 P（x）换成了 Q（x）。

然而由于我们的模型 Q 总是不完美的，所以用它来编码 P 的代价 H（P，Q），必然会比用 P 自己最优的编码代价 H（P）要高。

多出来的这部分代价就是 KL 散度（Kullback-Leibler Divergence）。它精确地量化了两个概率分布之间的“距离”（虽然不是严格意义上的数学距离）。

KL 散度的公式如下:

$$D_{KL}(P||Q) = H(P,Q) - H(P) = \sum_x P(x) \log_2 \frac{P(x)}{Q(x)}$$

我们可以发现交叉熵，信息熵和 KL 散度之间存在以下关系：

$$\text{交叉熵}(H(P,Q)) = \text{信息熵}(H(P)) + \text{KL散度}(D_{KL}(P||Q))$$

在训练 LLM 时，我们通常使用最小化交叉熵损失函数作为训练目标。

但是我们接触到的真实数据（各种训练语料）是固定的，所以真实世界的信息熵 H（P）是一个我们无法改变的常数。

因此，我们的优化目标就等价于最小化模型预测 Q 和真实数据 P 之间的 KL 散度。

这就是交叉熵损失函数在整个深度学习领域如此流行的根本原因。它让我们有了一个明确的优化目标：让模型拟合的分布无限接近于真实分布。

02 微观的“Token 熵”与宏观的“策略熵”

有了上述基础，我们终于可以来看这两篇论文了。一个非常有意思的地方是，这两篇论文虽然都在谈论熵，但它们观察熵的尺度是不同的。

《The Entropy Mechanism》关注的是宏观的、全局的“策略熵”（Policy Entropy）。

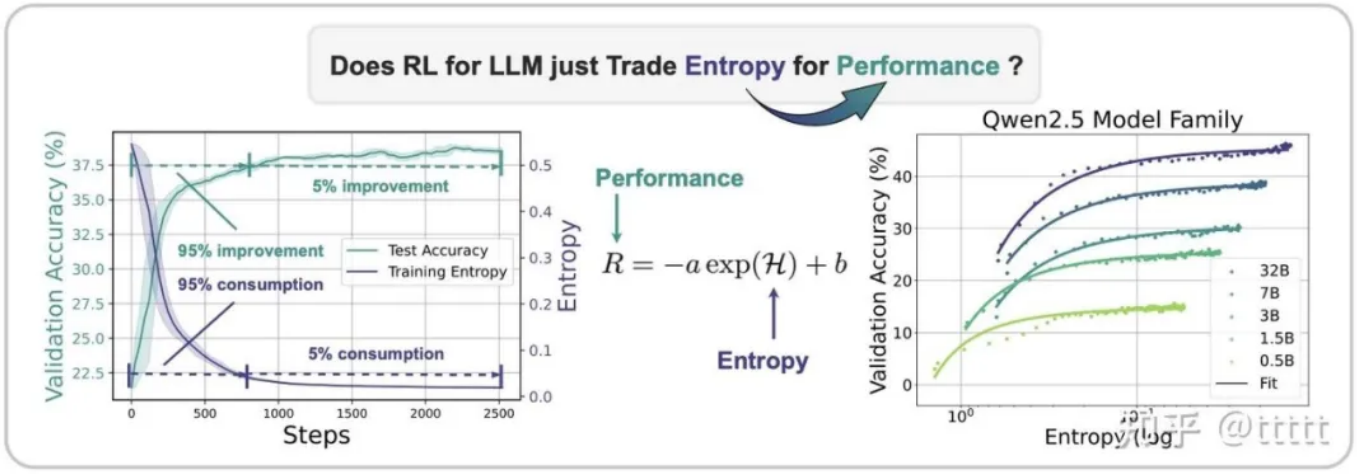
它关心的是模型在 RL 训练过程中的整体健康状况，特别是策略熵是否会过早“崩溃”。

《Beyond the 80/20 Rule》关注的是微观的、局部的“Token 级熵”（Token-level Entropy）。它把熵当作诊断工具，去寻找推理链条中那些最关键的“分叉路口”。

（1）RL 中的“熵崩溃”现象

《The Entropy Mechanism》这篇论文首先抛出了一个在 RL 训练 LLM（特别是推理任

务）时普遍存在的痛点：熵崩溃（Entropy Collapse）。



在训练早期，模型的熵会急剧下降，说明模型性能的提升是通过牺牲熵换来的

模型在 RL 训练的早期阶段，策略的熵会急剧下降。模型迅速地对某些特定的答案或路径变得“过度自信”，其行为多样性（探索能力）急剧丧失。

这种“熵崩溃”的直接后果就是性能饱和。模型因为失去了探索能力，早早地就锁死在了一个次优解上，无论你再怎么加大训练，奖励（Reward）都很难再提升。

论文发现了一个经验公式：$R = -a \cdot e^{H} + b$，其中R是奖励，H是熵。这个公式明确告诉我们：**奖励的提升，是用熵的消耗换来的。**当熵（探索能力）耗尽时，性能的提升也就到头了。

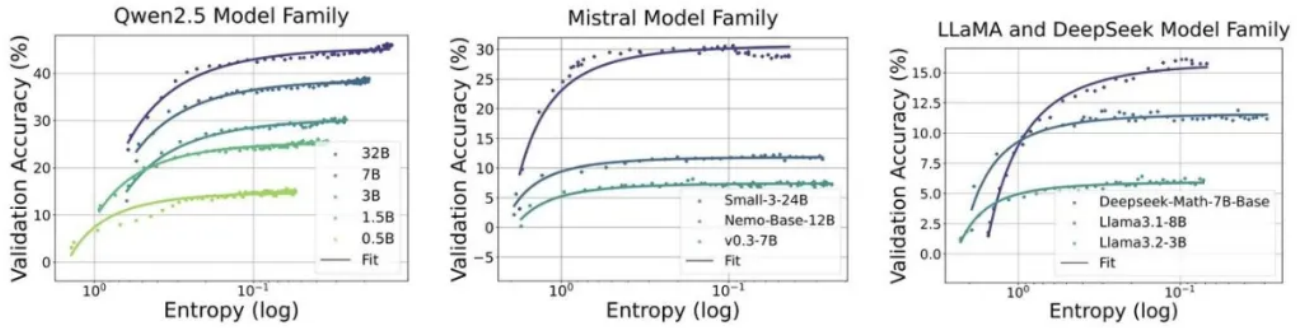
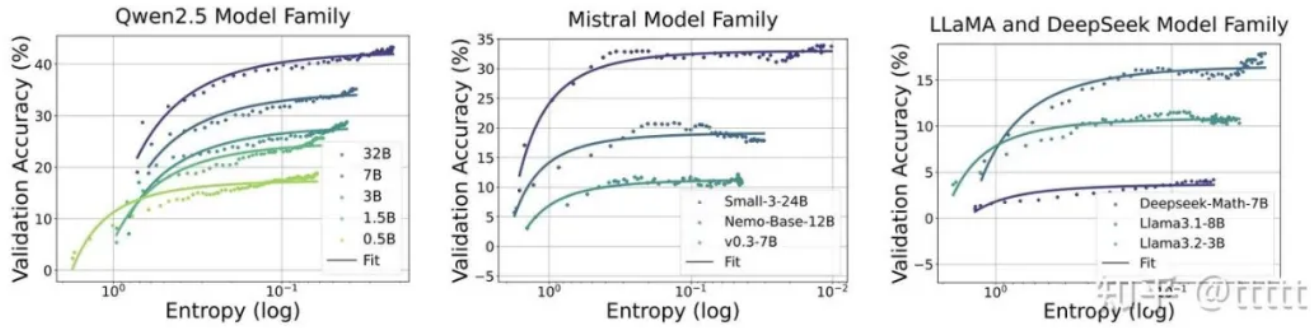


Figure 3: Fitting curves between policy entropy and validation performance on math task. We conduct validation every 4 rollout steps until convergence.



很多模型都表现都符合这个公式

那么，为什么会发生“熵崩溃”？论文从数学上给出了一个解释。作者推导出，策略熵的变化与一个关键因素——动作概率和优势函数（Advantage）的协方差——有关（反比关系）。

原文中的公式比较复杂，感兴趣的读者可以自行拜读。这里提供一个通俗易懂的说法（在数学上不一定严谨）。

简单来说：

- 当模型选择一个高概率的动作（token），而这个动作又带来了高奖励（高Advantage）时，强化学习算法会大力强化这个选择。
- 这种“强强联合”的更新，会导致这个高概率动作的概率变得更高，其他动作的概率被压制，从而使得整个概率分布的熵急剧下降。

在 RL 训练初期，模型很容易找到一些“低垂的果实”，即一些简单、高回报的捷径。

于是模型疯狂地在这些路径上进行自我强化，导致协方差持续为正，熵一路狂跌，最终“熵崩溃”，探索能力耗尽。

为了解决这个问题，论文提出了 Clip–Cov 和 KL–Cov 等方法，核心思想就是限制那些高协方差 token 的更新幅度。

翻译成大白话就是：“我知道你这个选择又自信又正确，但你先别太激动，悠着点更新，给别的可能性留点机会。”

Clip–Cov：

$$L_{\text{Clip-Cov}}(\theta) = \begin{cases} \mathbb{E}_t \left[\dfrac{\pi_\theta(y_t \mid \mathbf{y}_{<t})}{\pi_{\theta_{\text{old}}}(y_t \mid \mathbf{y}_{<t})} A_t \right], & t \notin I_{\text{clip}} \\ 0, & t \in I_{\text{clip}} \end{cases}$$

随机选择一小部分具有正协方差的 token，将其梯度分离，阻止其更新。

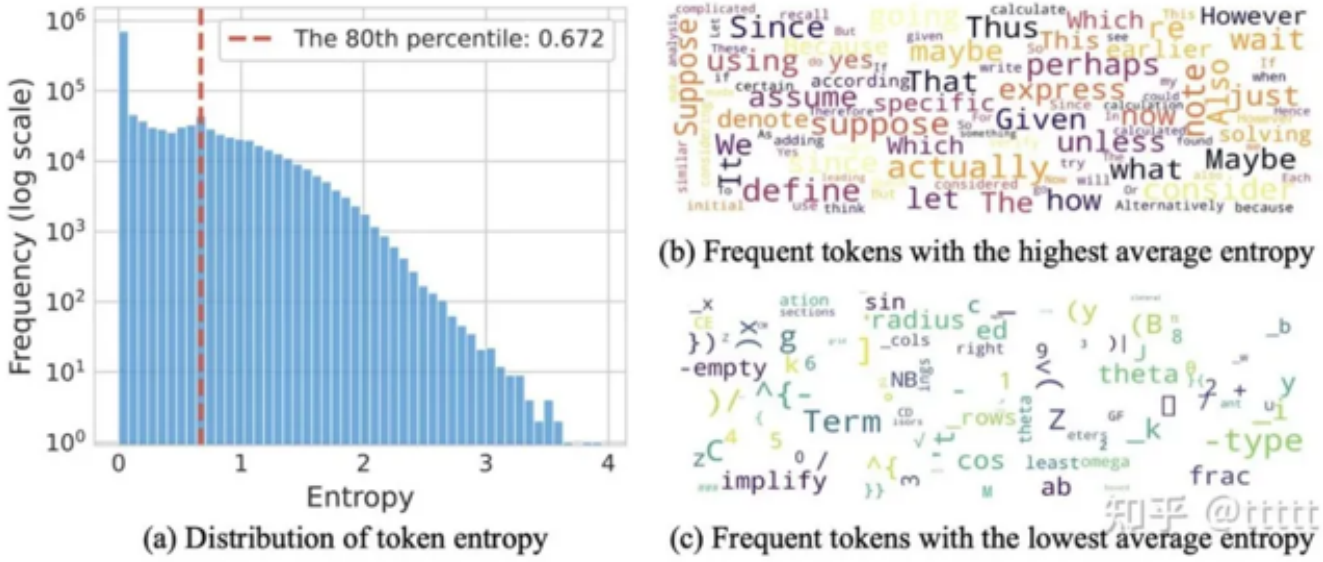
KL–Cov：

$$L_{\text{KL-Cov}}(\theta) = \begin{cases} \mathbb{E}_t \left[\dfrac{\pi_\theta(y_t \mid \mathbf{y}_{<t})}{\pi_{\theta_{\text{old}}}(y_t \mid \mathbf{y}_{<t})} A_t \right], & t \notin I_{\text{KL}} \\ \mathbb{E}_t \left[\dfrac{\pi_\theta(y_t \mid \mathbf{y}_{<t})}{\pi_{\theta_{\text{old}}}(y_t \mid \mathbf{y}_{<t})} A_t - \beta \mathbb{D}_{\text{KL}} \left(\pi_{\theta_{\text{old}}}(y_t \mid \mathbf{y}_{<t}) \,\|\, \pi_\theta(y_t \mid \mathbf{y}_{<t}) \right) \right], & t \in I_{\text{KL}} \end{cases}$$

对协方差排名前 k 的 token，增加 KL 散度正则化。

这篇论文从宏观上指出了问题（熵崩溃）并给出了病因（协方差机制），但它留下了一个更深的问题：熵的价值在不同地方是相同的吗？

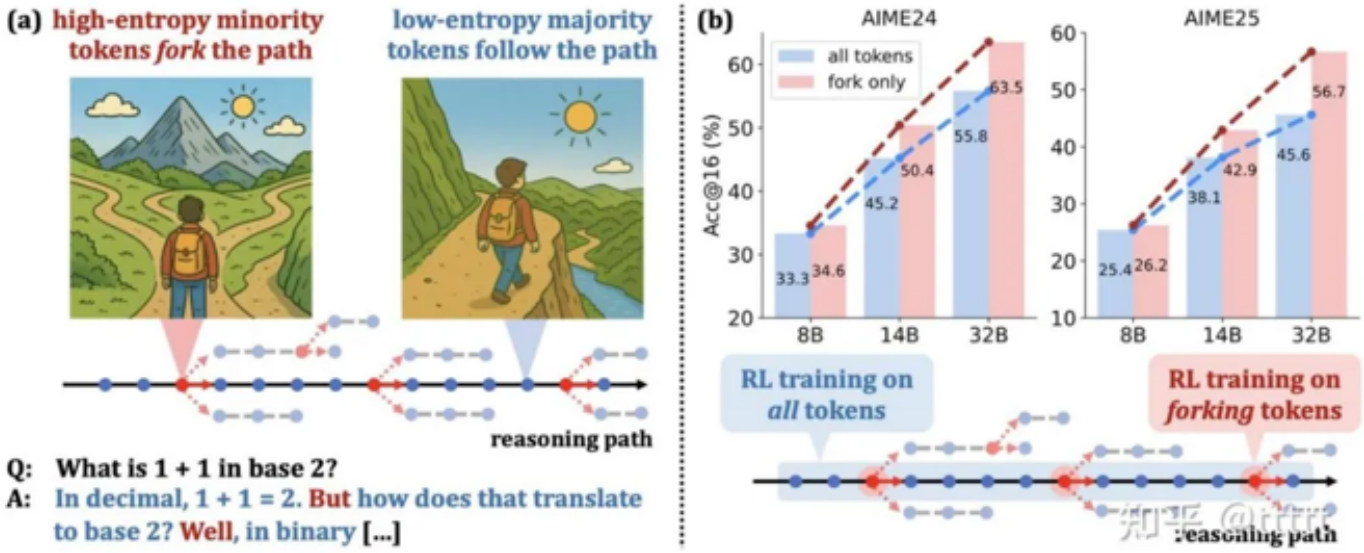
（2）抓住“关键少数”



80% 的 token 熵都低于 0.672。可以看到低熵 token 基本上都是各种符号，高熵 token 基本上是一些转折词

研究者发现了一个惊人的“二八定律”：

- 在一个完整的推理链中，大约 80% 的 Token 都是低熵的。这些词通常是模板化的、衔接性的、或者是确定性的计算步骤（比如 1+1 后的 =2）。模型生成这些 token 都时候有很高的置信度。
- 而只有大约 20% 的 Token 是高熵的。这些词是思维链真正的“分叉路口”（forking tokens）。在这些关键节点上，模型感到不确定，存在多个合理的推理方向。例如，在解一道应用题时，选择先计算哪个中间变量，就是一个典型的高熵决策点。



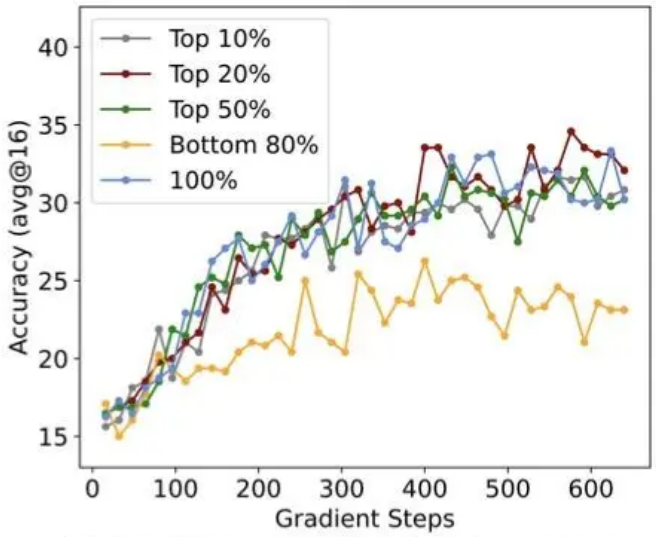
左图是普通 token 和 forking token。右图是普通 RL 和只训练 forking token 的区别

这篇文章的核心论点是：RL for Reasoning 的有效性，几乎完全来自于对这 20% 高熵“关键少数”的优化。

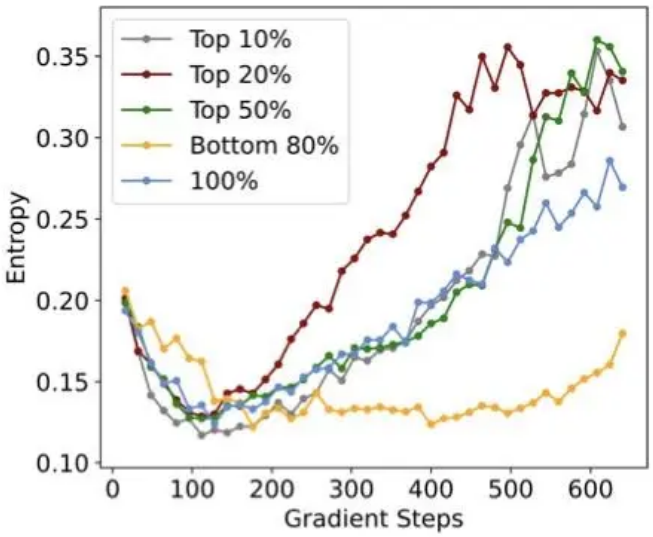
作者团队做了以下实验：

- **全量更新：**正常的 RL 训练，对所有 token 的梯度都进行更新。
- **只更新高熵 Token：**只计算并更新那 20% 高熵 token 产生的梯度，直接丢弃 80% 低熵 token 的梯度。
- **只更新低熵 Token：**作为对比，只更新 80% 低熵 token 的梯度。

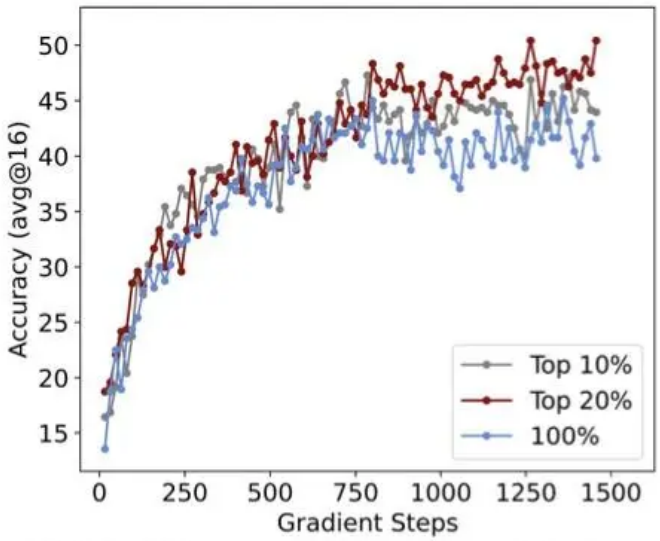
结果如下：



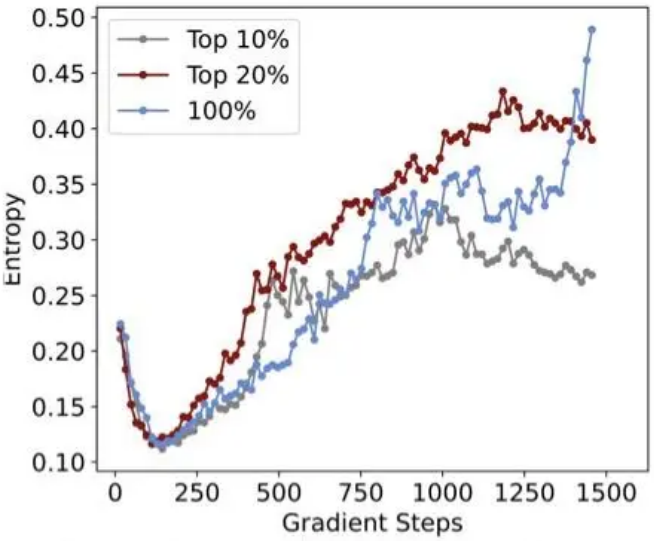
(a) AIME’24 scores trained from Qwen3-8B base.



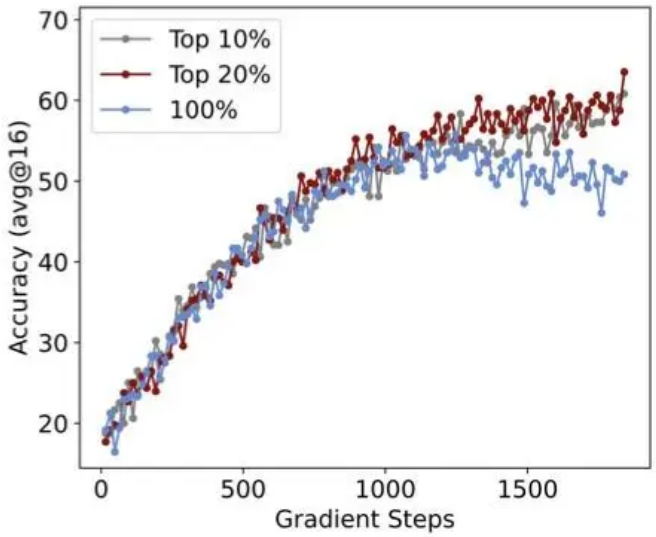
(b) Overall entropy trained from Qwen3-8B base.



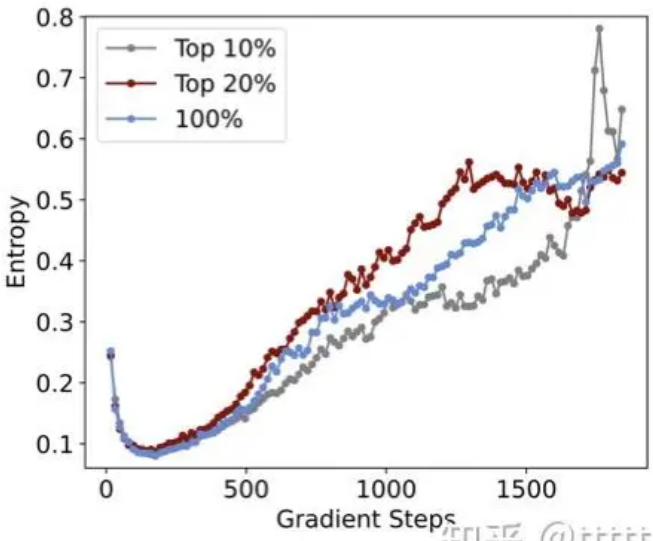
(c) AIME’24 scores trained from Qwen3-14B base.



(d) Overall entropy trained from Qwen3-14B base.



(e) AIME’24 scores trained from Qwen3-32B base.



(f) Overall entropy trained from Qwen3-32B base.

Qwen 系列的结果

作者发现：

- “只更新高熵 Token”的策略，其性能与“全量更新”相当，甚至在某些更强的模型上表现更好
- 而“只更新低熵 Token”的策略，性能则出现了断崖式下跌。

这个实验证明了：RL 并不是在机械地加强一整条“正确答案”的路径。它真正的作用，是帮助模型学会在那些充满不确定性的关键决策点，如何做出更优的选择。

那 80% 的低熵部分，模型在 SFT 阶段已经学得很好了，再用 RL 去“用力”，反而是浪费计算资源，甚至可能破坏模型的语言流畅性。

03 总结

RL 是多样性和准确性，探索能力和基础能力之间的权衡。

RL 期间如果不进行熵的控制，模型会陷入“熵崩溃”，过早丧失探索能力，导致性能无法提升。

作者：ttttt

来源：https://zhuanlan.zhihu.com/p/1954330684970754139

END