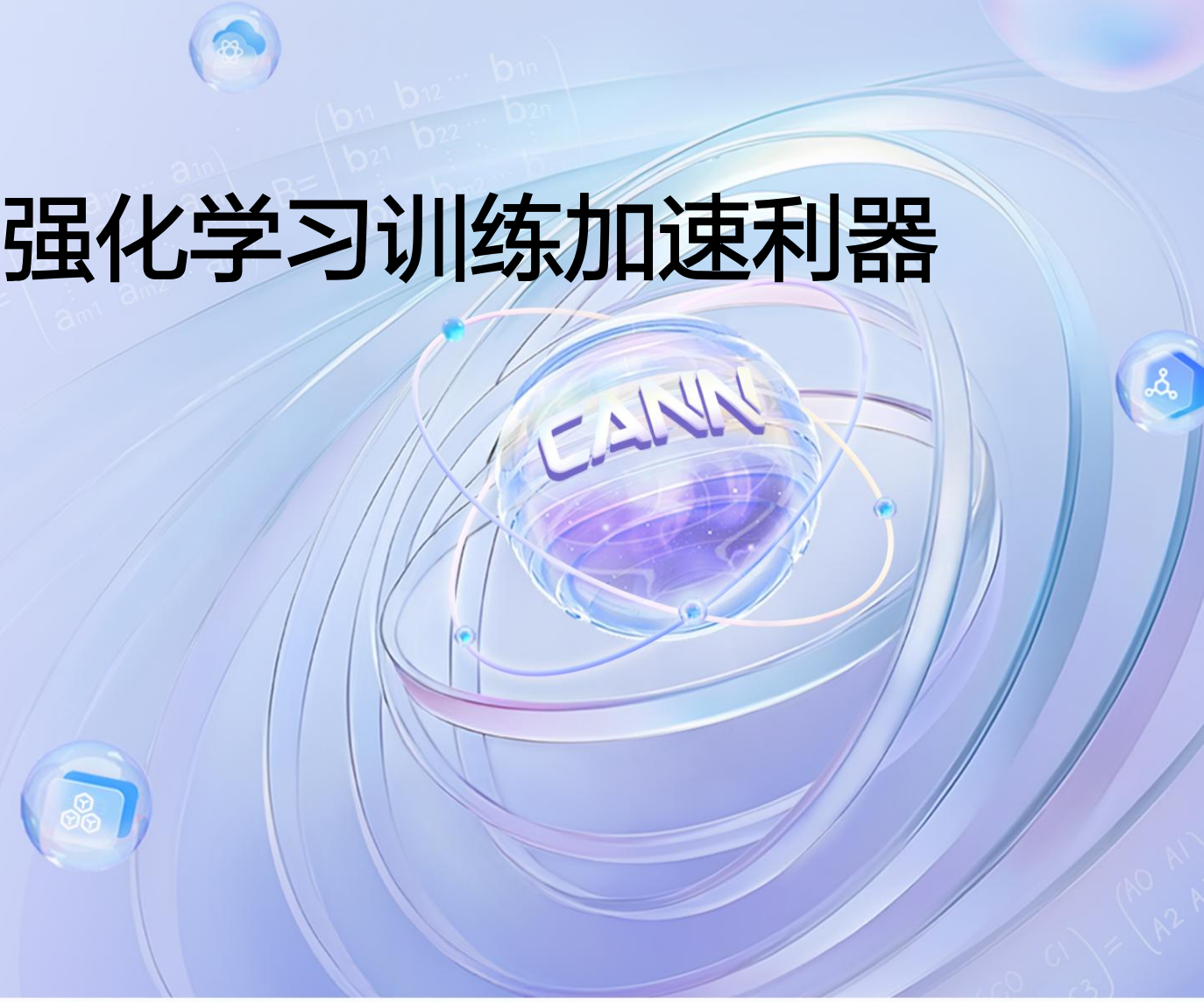
SAM投机推理：长序列强化学习训练加速利器

作者：胡元泉

时间：2025.12.18



CANN

目录

CANN

大语言模型强化学习(RL)训练面临严峻的推理瓶颈

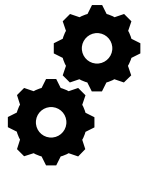
核心问题

- RL后训练场景通常需要进行数百万次的交互式采样（Rollout）来收集和评估数据，这一过程是主要的性能瓶颈。

	Rollout	Training	Weight Update
Moonlight	84%	14%	2%
Qwen2-VL-72B	63%	31%	6%
Kimi-K2	87%	10%	3%

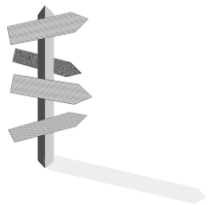
- 投机解码（Speculative Decoding，SD）使用比原始模型小得多的近似模型进行自回归采样，而用大模型并行验证采样结果，能够大幅度提升推理效率。但是传统的SD方法存在一定局限性。

传统SD方案的不足



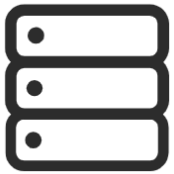
依赖辅助模型：

需要训练和维护一个的用于生成草稿的小模型，可以是额外的模型，也可以在原模型添加auxiliary head，无论哪种都会增加系统复杂度和内存管理成本。



分布漂移风险：

RL 训练中，主模型策略不断更新，辅助模型很难与之保持分布一致，可能影响采样质量和训练效果。

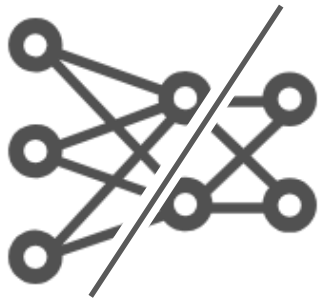


资源占用：

辅助模型还是会占用额外的显存，这在计算资源紧张的训推共卡 (Colocate) RL框架中尤其关键。

CANN

我们使用一种基于后缀自动机的投机解码方案（SAM-Decoding）



无模型 (Model-Free)

无需任何辅助草稿模型，从根本上消除了模型维护成本和分布漂移问题。

基于检索(Retrieval-Based)

利用RL数据中固有的结构化和重复特性。SAM-Decoding 高效地从历史上下文中检索匹配序列作为高质量草稿。

即插即用(Plug-and-Play)

可无缝集成到现有训练框架中，无需修改模型或增加额外内存管理，对现有系统侵入性极低。

SAM数据结构介绍

什么是SAM?

SAM是后缀自动机（Suffix Automation）的简写，是一个能够高效解决多种字符串问题的数据结构，可以理解为对给定字符串的所有子串的压缩表示。它能以线性的时间和空间复杂度构建。

两个核心概念

1. endpos 等价类

所有结束位置（endpos）完全相同的子串被归入同一状态，以此实现对子串信息的高效压缩。

示例：对于字符串 ABCBC，子串 BC 的 endpos 集合是 {2, 4}，C的 endpos 集合也是 {2, 4}，因此它们属于同一个 endpos 等价类。

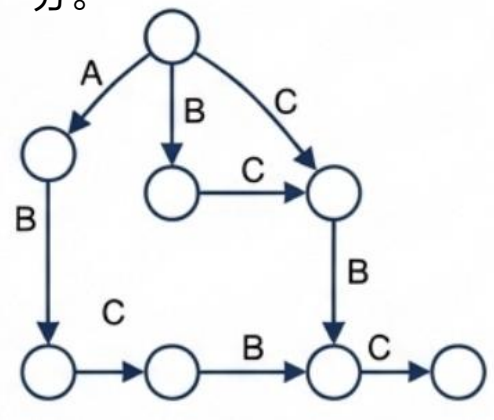
2. 后缀链接 link

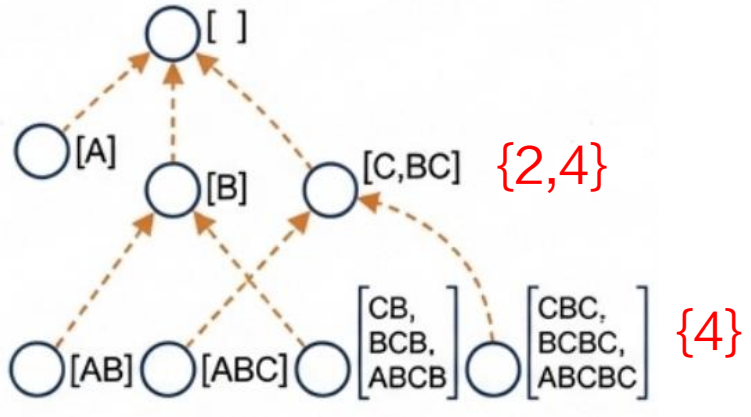
link 指向当前状态中最长子串的“最大真后缀”所在的状态。link 构成了 SAM的核心树形结构，是实现线性时间构建和快速回溯的关键。

SAM 主要维护两个集合

next（状态转移）：
按前序匹配字符串，是SAM的自动机部分。

link（后缀链接）：
用于高效回溯，是SAM的结构部分。





SAM 的快速增量构造

每次向当前 SAM 中加入一个新字符时，同时更新 next 和 link，单次更新的时间复杂度为 O(1)。

CANN

SAM-Decoding算法介绍

基于 SAM 的投机推理：

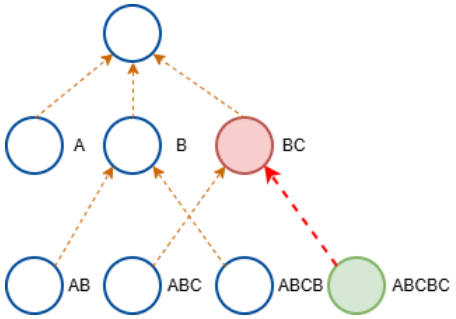
通过 link 回溯和 next 匹配高效定位候选序列

1. **回溯**：从当前序列的最后一个token对应的SAM节点开始，沿着link边向上回溯。

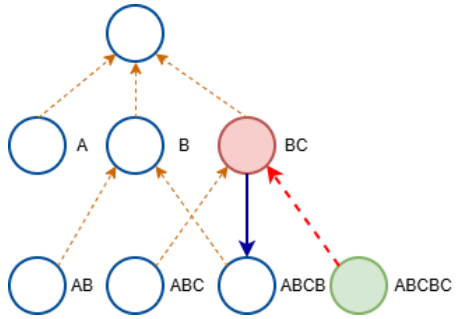
2. **定位**：到达一个能够通过next边转移到下一个token的节点后，停止回溯。该节点的最长子串代表了在前缀子串中出现过的最长后缀。

3. **提取**：根据节点记录的最小结束位置（min_endpos），在原始文本中定位该后缀，并提取其后续的连续token作为候选序列（草稿）。

图解示例：为ABCBC生成下一个草稿



步骤1：当前序列是ABCBC，从绿色节点开始，沿着 link边（红色箭头）向上回溯到BC节点。



步骤2：BC 节点可以经由next边（蓝色实线箭头）转移到下一个状态，回溯停止。

结论：BC是最长匹配后缀，因此提取其后续序列BC作为草稿。

CANN

与其他投机推理方法的比较

- SAM-Decoding：基于检索的非参数化方法。完全通过算法数据结构挖掘上下文中的重复模式。
- EAGLE-3：基于特征外推的 auxiliary head 方法。通过极轻量的“寄生”head 在特征层级进行推测。
- MTP：基于模型本身的自投机方法。将投机能力内化为模型训练目标的一部分。

核心差异对比表

特性 \ 算法	SAM-Decoding	EAGLE-3	MTP
草稿生成方式	借助后缀自动机，从历史文本中，检索“最长后缀 + 后续序列” 。	利用轻量级 EAGLE head 结合树状注意力（Tree Attention）进行特征外推。	在模型结构里加入multi-token heads，一次预测多个未来 token。
额外组件	无额外模型参数；如果使用冷启动需要一个额外的语料库。	需要训练一个依附于主模型的极小网络（参数量通常 <1%）。	需要在模型预训练阶段就集成 multi-token heads。
部署成本/难度	较低：即插即用，无需对模型进行任何微调或附加训练。	中等：需要针对特定主模型训练 Adapter，增加内存管理复杂度。	较高：需要在预训练阶段就加入 MTP 结构，现有模型难以直接获得此能力。
适用场景	在结构性强、重复模式多的场景（如数学、代码、Agent）中表现极佳。	在与EAGLE head训练数据分布相似的场景中有较好的表现。	具有较好的泛化性，能够应对更加多样化的使用场景。

目录

三层校验确保SAM-Decoding的精度无损

理论保障

投机解码通过拒绝采样（Rejection Sampling）机制，对草稿token进行逐一验证。若候选token 符合大模型的预测分布则接受，否则拒绝并由大模型重新生成。理论上，这一过程确保最终输出与大模型直接生成的结果完全一致。

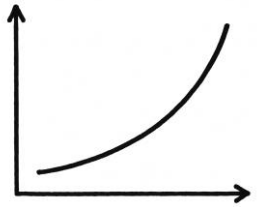
实践中的三层验证体系

1. 算子级精度 (Operator-Level Precision)

验证投机解码使用的底层计算算子与标准解码算子在数值上是否等价。

2. 任务级评测 (Task-Level Evaluation)

检验开启 SAM 后，在标准的下游任务 Benchmark 上模型准确率是否发生变化。



3. 端到端训练 (End-to-End Training)

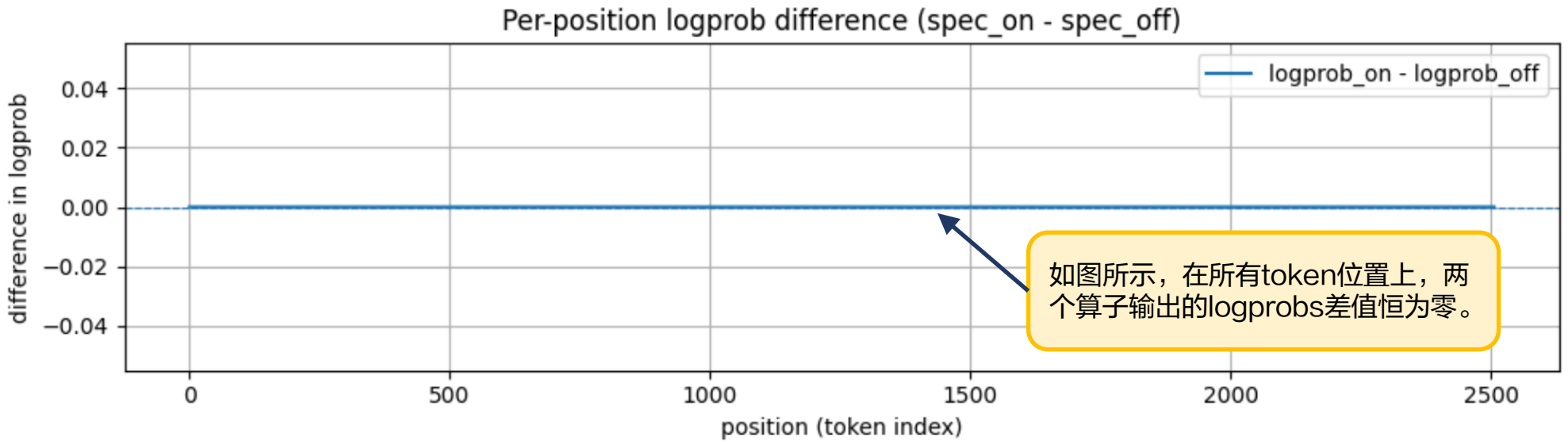
对比开启和关闭 SAM 的完整 RL 训练过程，观察训练动态（如Reward 曲线）是否保持一致。

CANN

验证一：核心算子精度验证

验证方法

在 vLLM-Ascend 中，标准解码使用 torch-npu 的 _npu_paged_attention 算子。投机解码使用 _npu_paged_attention_splitfuse 算子来验证生成的 draft tokens。我们对相同的输入，统计了这两个算子在每个token位置上预测的 logprobs 的差值。



普通解码和投机解码所使用的核心算子之间没有精度差异，为无损加速提供了底层保障。

验证二&三：下游任务准确率不变，端到端 RL 训练曲线吻合

Part 1: 下游任务评测

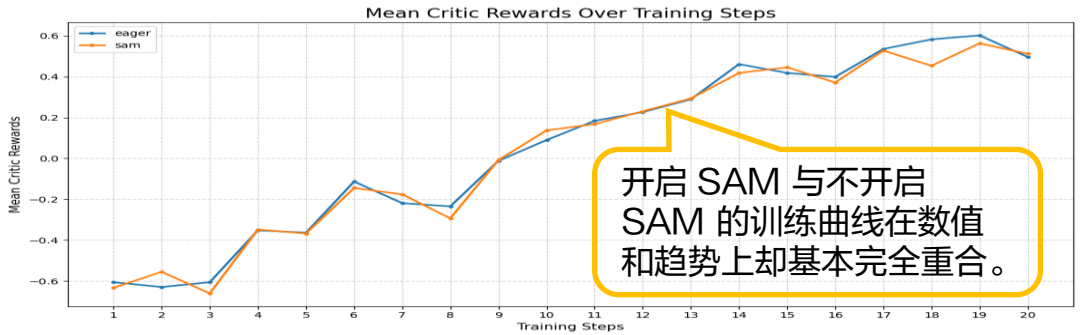
模型与配置：Qwen3-32B，temperature=0.6，pass@1

数据集（Dataset）	不开启SAM（SAM Off）	开启SAM（SAM On）
GSM8k	88.9%	88.7%
AIME2024	66.5%	66.8%

开启 SAM 投机解码不影响模型在数学推理任务上的准确率。

Part 2: 端到端 RL 后训练

模型与配置：Qwen3-32B，DAPO算法，DAPO-MATH-17K数据集，固定随机种子



结合三层验证，可以判断 SAM 投机解码在 RL 后训练场景的精度是严格无损的。

目录

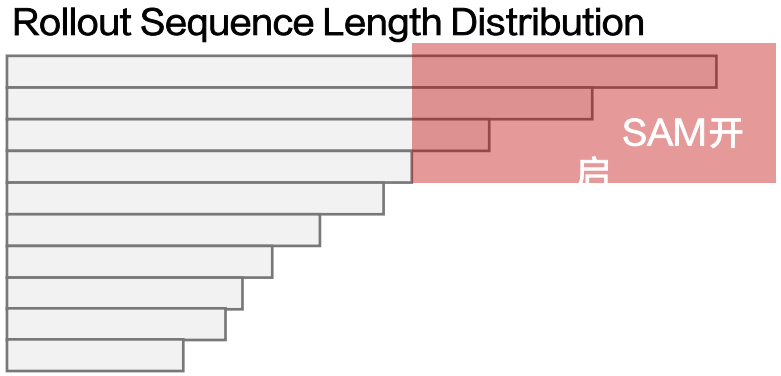
投机解码工程优化

最大化SAM-Decoding在RL训练中的加速收益，关键在于：
①**提高接受率**；②**减少额外耗时**。

1. 自适应Batch Size开关

机制：仅当vllm engine当前处理的请求数量小于等于阈值时，才激活投机解码。

目的：①在请求数量数减少、序列变长的rollout后期开启投机，能够提高接受率；②避免在大batch size下由于算子膨胀导致验证开销超过收益。



2. 拒绝采样加速

问题：vllm-ascend原生实现的拒绝采样包含多层Python循环，额外耗时超过50ms, 完全掩盖了投机收益。

解决方法：通过纯PyTorch代码向量化 (Vectorization)优化，消除循环，利用NPU并行计算能力，将该模块耗时减少到原来的十分之一。

Before

缓慢的串行Python循环

After

高速的并行Tensor操作

CANN

实测结果：SAM投机解码为Qwen3系列模型RL训练带来高达10%的端到端加速

测试场景：Eager模式，DAPO长序列RL训练，数据集DAPO-MATH-17k，最大回复长度34k

Qwen3-32B Dense 性能数据

基础配置：2机16卡32die，TP=8，
gen_batch_size=96
SAM相关配置：自适应开关阈值为8，投机个数为3

指标	不开启SAM	开启SAM	收益
单轮平均推理时间/s	3904.22	3548.62	10.02%
单步总推理时间/s	4159.06	3793.29	9.64%
单步总时间/s	4730.8	4374.9	8.13%

Qwen3-235B MoE 性能数据

基础配置：8机64卡128die，TP=4，gen_batch_size=128
SAM相关配置：自适应开关阈值为8，投机个数为3

指标	不开启SAM	开启SAM	收益
单轮平均推理时间/s	7102.92	6467.52	9.82%
单步总推理时间/s	14287.54	12811.98	11.52%
单步总时间/s	15441.45	13960.98	10.60%

SAM在Rollout长尾阶段表现尤为突出，可实现超过35%的局部加速

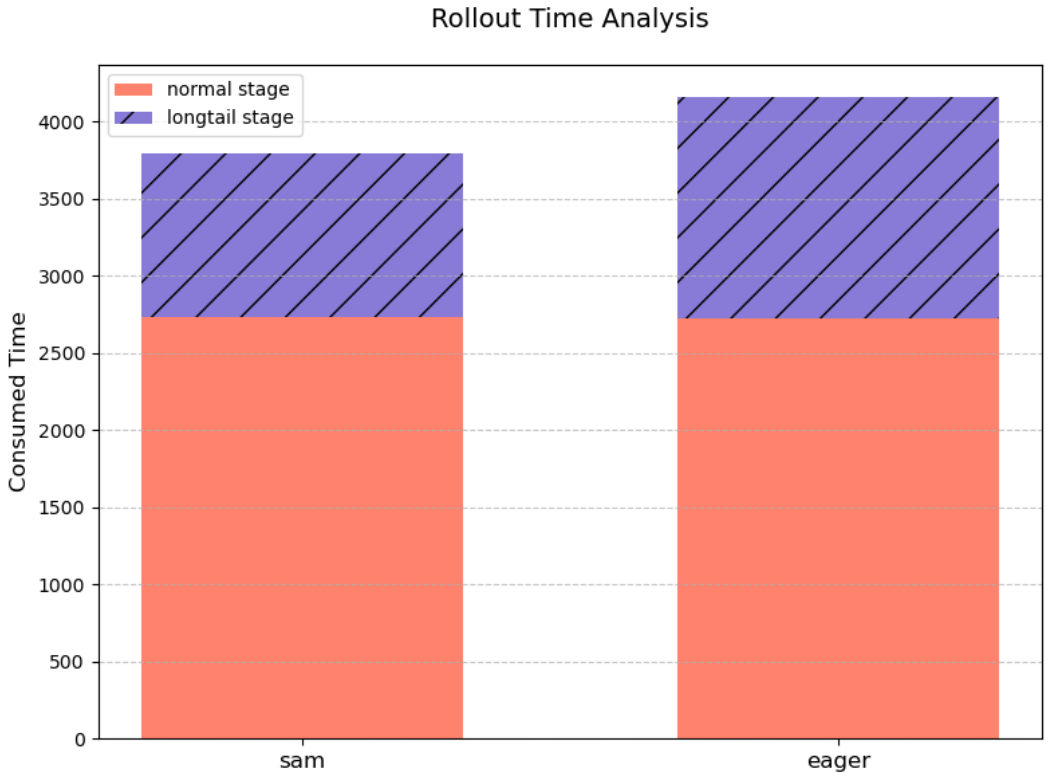
什么是Rollout长尾阶段?

在同步RL框架中，由于生成长度不均，Rollout后期往往只剩下少数长序列请求在运行，总体的耗时由这些长尾请求决定（木桶效应）。

SAM自适应开关的作用

我们的自适应开关正是为这个阶段设计的。当请求数低于阈值，SAM投机解码被激活。此时动态SAM中包含了丰富的上下文，接受率显著提高，加速效果最明显。

我们单独统计了这一阶段的加速收益，如右图紫色部分所示，SAM在其使能的长尾阶段耗时显著减小，性能达到非投机的1.35倍。



数据表明SAM的接受率与序列长度正相关，天然适配长序列推理场景

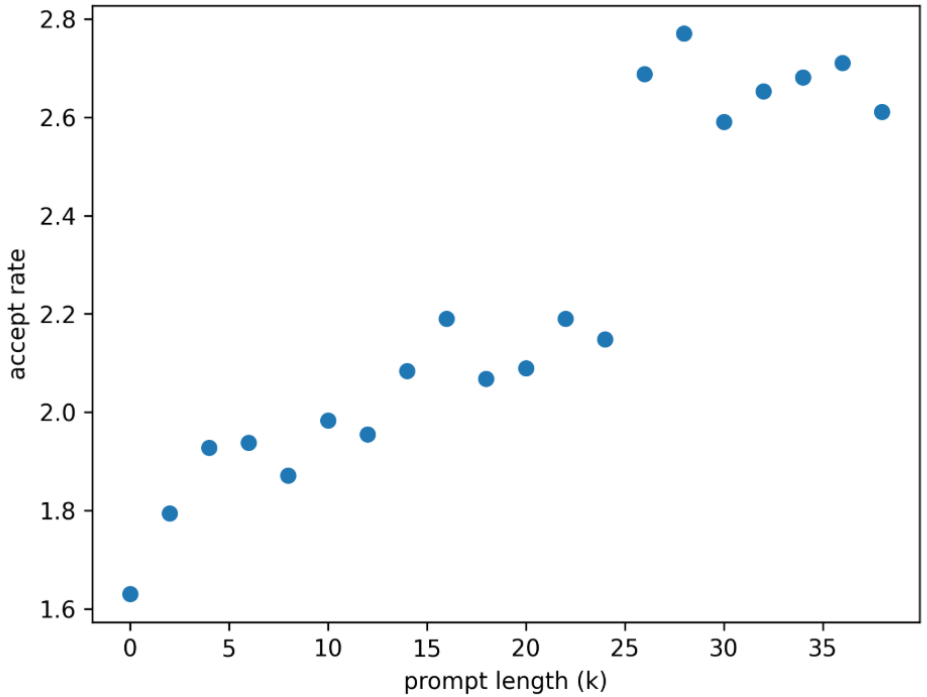
实验设置

- 模型：Qwen3-8B
- 数据集：Math500
- SAM配置：投机token 数=3
- 指标：平均接受长度 (Mean Acceptance Length)，包含1个 bonus token

核心观察

随着Prompt长度的增加，SAM能够构建更丰富的后缀自动机，从而提升接受率。这一特性解释了为何SAM-Decoding在长序列RL后训练场景中能取得显著收益。

接受率与序列长度关系图

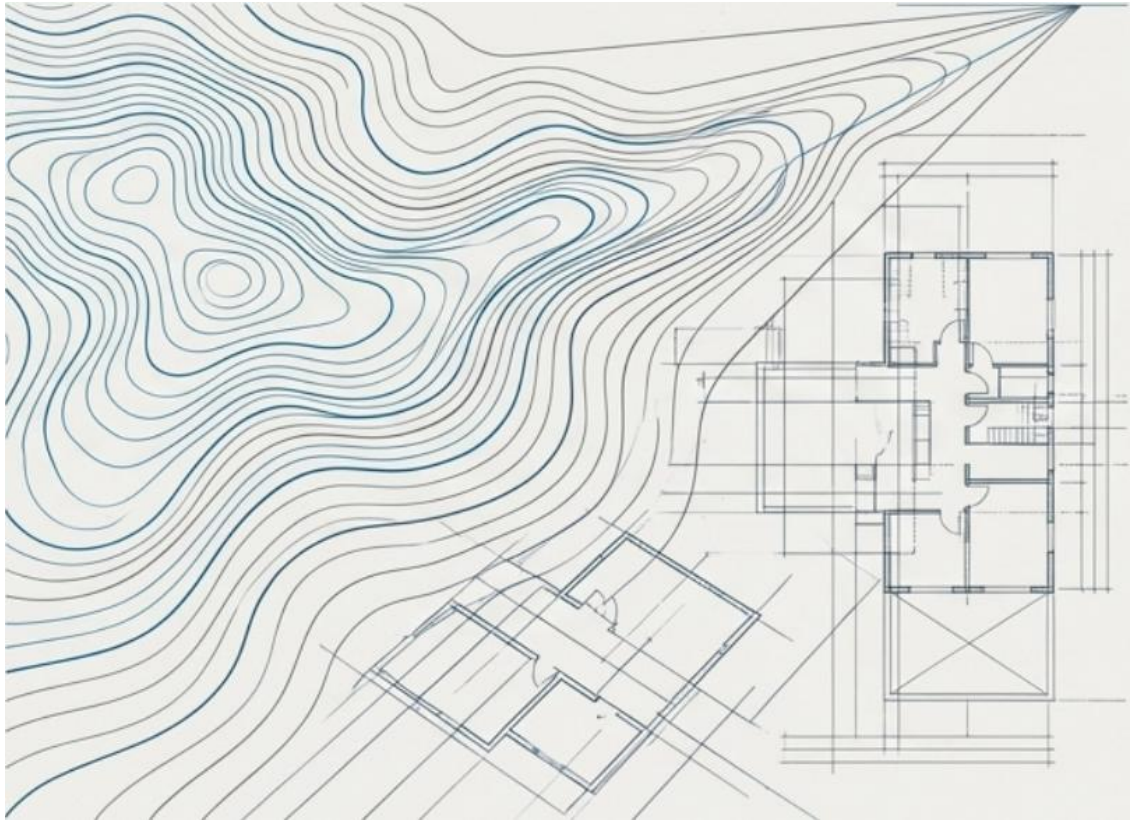


目录

总结与展望

核心贡献

- SAM-Decoding为LLM强化学习提供了一个即插即用且无损的加速方案。
- 证明了传统算法和数据结构层面的创新（如后缀自动机）可以作为模型参数方法强有力的补充。
- 提供了一个生产环境验证的长序列RL加速实践样例。



未来展望

- 探索更高效的投机策略，以提高接受率。
- 尝试将投机算子与图模式结合，提高算子执行效率。

cann-recipes-train

➢ 本实践已经开源至 gitcode 代码仓：https://gitcode.com/cann/cann-recipes-train/blob/master/llm_rl/qwen3/README.md

RL后训练执行

在本样例代码根目录下启动Qwen3-235B-A22B的RL后训练。

```
# 请注意，以下bash启动脚本中的内容需要手动配置
# source脚本路径：  根据实际CANN安装目录调整
# MASTER_ADDR：  ray集群主节点的IP地址，每个节点的脚本配置一致
# SOCKET_IFNAME：  集群中各节点自己的网卡名，可通过ifconfig命令查看
# VLLM_DP_SIZE：  推理阶段DP配置，按推理模型切分和总卡数计算

bash ray_start_npu.sh TRAIN_SCRIPT ENV_SCRIPT
# 示例： bash ray_start_npu.sh ./internal/train_grpo_qwen3_235b_128die_random_init.sh ./internal/qwen3_235b_env.sh
# 如果不需要额外的环境变量配置，则不需要该参数，示例：bash ray_start_npu.sh ./internal/train_grpo_qwen3_32b_32die_true_weight.sh
```

可在 `ray_start_npu.sh` 启动训练时添加参数，实现随机权重训练GRPO算法、真实权重训练GRPO算法、真实权重训练DAPO算法，对应修改如下：

基础模型	训练	训练启动脚本	训练配置脚本	环境变量配置脚本
Qwen3-235B-A22B	随机权重训练 GRPO算法	ray_start_npu.sh	./internal/train_grpo_qwen3_235b_128die_random_init.sh	./internal/qwen3_235b_env.sh
Qwen3-235B-A22B	真实权重训练 GRPO算法	ray_start_npu.sh	./internal/train_grpo_qwen3_235b_128die_true_weight.sh	./internal/qwen3_235b_env.sh
Qwen3-235B-A22B	真实权重训练 DAPO算法	ray_start_npu.sh	./internal/train_dapo_qwen3_235b_128die_true_weight.sh	./internal/qwen3_235b_env.sh
Qwen3-32B	真实权重训练 GRPO算法	ray_start_npu.sh	./internal/train_grpo_qwen3_32b_32die_true_weight.sh	-
Qwen3-32B	真实权重训练 DAPO算法	ray_start_npu.sh	./internal/train_dapo_qwen3_32b_32die_true_weight.sh	-

目录

- Qwen3系列模型 RL训练...
  - 概述
    - Qwen3-235B-A22B
    - Qwen3-32B
  - 硬件要求
  - 基于Dockerfile构建环境
  - 数据集准备
  - 模型权重准备
    - Qwen3-235B-A22B
    - Qwen3-32B
  - RL后训练执行
  - 附录
    - 文件说明
    - 手动准备环境

Preview Code 修改追溯 23.25 KB 历史 对比 源码 复制 IDE

SAM投机推理：长序列强化学习训练加速利器

针对大语言模型强化学习训练（RL-training）中的海量交互式采样耗时瓶颈，传统的投机解码（Speculative Decoding）技术依赖辅助模型，而且存在策略更新导致的分布漂移风险。本文提出在 RL 训练中引入一种基于**后缀自动机**（SAM）的无模型（Model-Free）投机解码方案，该方法无需任何辅助模型，利用 RL 数据（如数学推理、代码生成）中固有的结构化重复特性生成候选序列检索。本文结合自适应 batch 调度与向量化拒绝采样等工程优化，在Qwen3系列模型（32B/235B）上完成了 SAM 投机解码在 RL 后训练场景的端到端验证。实践表明，SAM 在保证精度严格无损的前提下，显著降低了 Rollout 推理延迟，特别是在长尾阶段获得了超过 35% 的加速收益，相关代码已在cann-recipes-train 全部开源。

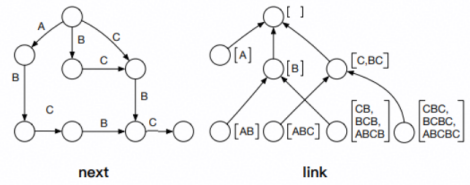
1. 背景介绍

1.1 SAM 原理

SAM（suffix automaton，后缀自动机）是一个能够高效解决许多字符串问题的数据结构。直观上，字符串的 SAM 可以理解为给定字符串的 **所有子串** 的压缩形式。SAM 主要维护两个重要的集合：

1. 结束位置 `endpos`：考虑字符串 S 的任意非空子串 T，我们记 `endpos(T)` 为字符串 S 中 T 的所有结束位置的集合。例如，对于字符串 ABCBC 我们有 `endpos(BC)` ={2,4}。在 SAM 中，所有满足 `endpos` 集合相同的字串被归入同一个**状态**，也被称为 `endpos` **等价类**。`endpos` 等价类的划分使得 SAM 可以以 O(|S|) 的空间复杂度存储所有子串信息。
2. 后缀链接 `link`：对于 SAM 中任意非初始状态 u，u 中的最长子串为 w，则 `link(u)` 指向的状态对应于 w 的后缀中与它的 `endpos` 集合不同且最长的那个，即 u 所代表所有子串的**最大真后缀**所在状态。后缀链接构成了 SAM 的核心树形结构，被称为**后缀链接树**，它使得 SAM 可以在 O(|S|) 的线性时间内构造完成，因为每次添加新字符时，只需要通过后缀链接快速定位和更新相关状态。

下图展示了基于字符串 ABCBC 构建的 SAM：



next link

目录

- SAM投机推理：长序列强...
  - 1. 背景介绍
    - 1.1 SAM 原理
    - 1.2 和其他投机方法...
    - 1.3 SAM 接受率
    - 1.4 SAM 在 RL 后训...
  - 2. RL 训练精度验证
    - 2.1 算子精度
    - 2.2 下游评测
    - 2.3 端到端后训练
  - 3. RL 训练性能验证
    - 3.1 性能优化
      - 3.1.1 自适应开关
      - 3.1.2 拒绝采样加速
    - 3.2 RL 训练实测收益
      - 3.2.1 Qwen3-32B
      - 3.2.2 Qwen3-23...
      - 3.2.3 客户场景验证
      - 3.2.4 Rollout 长...
  - 4. 总结与展望

https://gitcode.com/cann/cann-recipes-train/blob/master/llm_rl/qwen3/README.md

https://gitcode.com/cann/cann-recipes-train/blob/master/docs/llm_rl/sam_decoding.md

cann-recipes系列仓库

- cann-recipes希望通过提供拿来即用的算法模型样例，给到开发者最需要的指导：
- 如何快速上手？如何复现业界SOTA模型？如何榨干NPU性能？
- 内容涵盖：
  - 全场景算法样例：覆盖大模型、多模态、空间智能、具身智能各领域的算法样例
  - 高性能模型复现：基于CANN深度优化的主流模型训练与推理脚本
  - 特性优秀实践：针对CANN新特性的使用指南与性能调优技巧
- 目前已开源四个仓如下，正在紧锣密鼓持续建设中，清程极智、中科院等生态开发者也在积极贡献

仓名	定位	典型样例
cann-recipes-infer	推理样例	DSv3.2/Kimi-K2-thinking 0-day支持发布及高性能优化 HunyuanVideo/Wan2.2发布
cann-recipes-train	训练样例	Qwen3-MoE 32K长序列/DS-R1 RL训练样例
cann-recipes-embodied-intelligence	具身智能样例	Pi0推理样例
cann-recipes-spatial-intelligence	空间智能样例	Hunyuan3D/VGGT推理

cann-recipes-sig小组

cann-recipes交流群

欢迎广大开发者体验并参与贡献，如有疑问可通过issue、SIG或者cann-recipes交流群联系我们！

Thank you.

社区愿景：打造开放易用、技术领先的AI算力新生态

社区使命：使能开发者基于CANN社区自主研究创新，构筑根深叶茂、跨产业协同共享共赢的CANN生态

Vision: Building an Open, Easy-to-Use, and Technology-leading AI Computing Ecosystem

Mission: Enable developers to independently research and innovate based on the CANN community and build a win-win CANN ecosystem with deep roots and cross-industry collaboration and sharing.

上CANN社区获取干货

关注CANN公众号获取资讯

https://gitcode.com/cann

CANN