Alibaba

UMEM: 打破自演进 Agent "死记硬背" 困局的可泛化记忆提取和管理框架

演讲人：兰天

Alibaba ATH MaaS 业务线

目录

CONTENTS

Alibaba

01 / Agent Memory 背景

什么是 & 为什么是 Self-Evolving Memory?

为什么需要 Agent Memory / Self-Evolve Memory

Training-free v.s. Training-based Optimization

参数化微调

优化模型内置参数

优点：
- **泛化性好**：优化后类似任务有泛化性
- **上限高**：持续推高模型能力天花板

缺点：
- **可解释性差**：无法审查学到了什么
- **开销大**：模型尺寸越大训练成本越高
- **影响通用性**：损失模型通用性

Self-Evolve Memory

优化作为模型外挂参数（Memory）

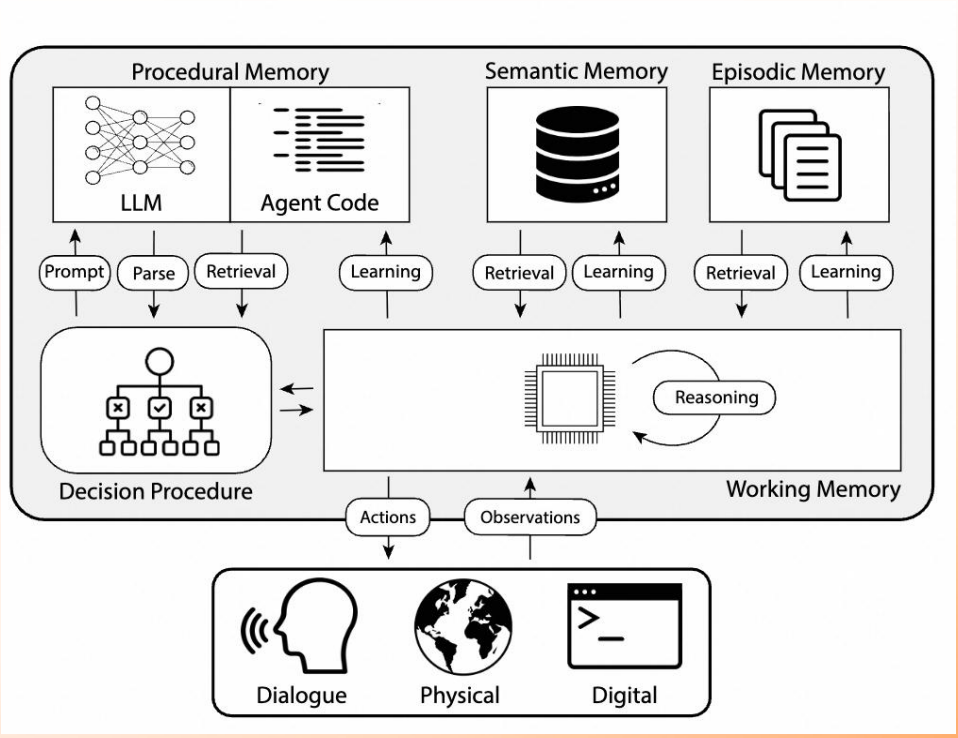
优点：
- **可解释性好**：人类可读的经验记忆
- **开销小**：操作memory无需优化参数
- **通用性保留**：模型冻结通用能力保持

缺点：
- **上限较低**：基于当前模型的固有能力，提升对经验记忆利用能力
- **泛化性：取决于积累经验记忆的内容**

Self-Evolve Memory: 更新外挂 Memory 提升模型下游任务效果
经验记忆的可泛化性是关键

Agent Memory的分类：从Memory的功能分类

Cognitive Architectures for Language Agents (ICML 2024)



抽象程度、可泛化性和构建成本逐渐提高

Agent Memory的分类：从优化Memory的目标分类

层级	优化对象	谁在被学	代表工作
C0	框架类（无需训练优化）	—	Mem0, ExpeL, ReasoningBank, Memp
C1	优化 Memory 检索器 / 路由器	Memory Retriever	RCR-Router, HippoRAG
C2	优化 Executor 对 Memory 管理和使用	Executor Agent	MemAgent, Memory-as-Action
C3	优化 Memory Bank 管理策略	Management Policy Model	Memory-R1, Mem-α
C4	优化 Memory 提炼策略	Extractor Policy Model	EvolveR, Training-free GRPO

Agent Memory的分类：从 Memory（Experience）的泛化性分类

层级	名称	说明	相关工作
L0	无记忆	—	—
L1	Working	对当前样例的 test-time scaling	ParallelMuse, Deep Search Test-time Scaling
L2	Instance-level	记忆对相似样例的泛化	Memp, ReMem
L3	Task Skill	对类似任务的归纳	EvoSkill, SkillRL
L4	Harness	记忆和经验针对具体任务的 harness 进行优化	Meta-harness

提升 Memory or Experience 的泛化性是 Self-Evolve Agent Memoy的核心

UMEM：一套优化记忆提取模型的训练框架，提升提炼记忆的泛化性

02 /

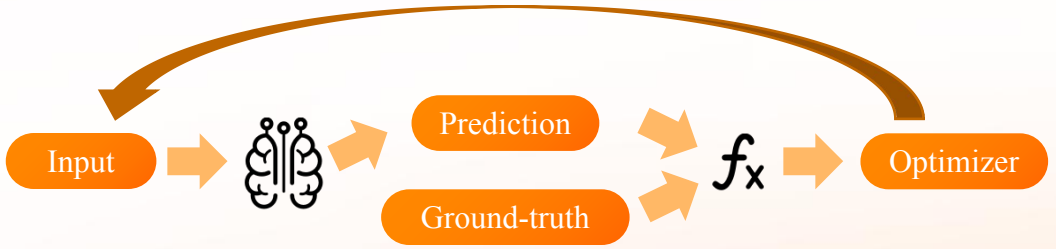
UMEM 框架

联合优化“提取和管理”可泛化的记忆

类比：Self-Evolving Agent Memory V.S. 模型参数微调

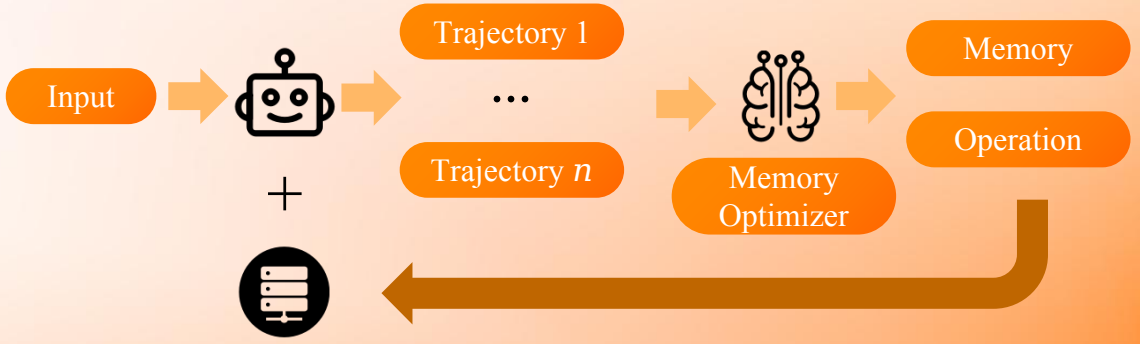
参数化更新模型

- 任务执行 → 前向传播 → 计算梯度 → 反向传播更新参数
- Loss 计算取决于任务的答案



Self-Evolving Agent

- Agent 执行 → 经验轨迹 → Memory Optimizer Model 提炼经验记忆 + 记忆更新操作 → 更新 Memory Bank
- 提炼的 Memory 类比梯度



问题形式化

Self-Evolving Memory: 固定 Executor Agent，优化 Memory Bank

Memory 形式	Memory 检索	轨迹收集（前向推理）	Memory 更新
• Key-Value 形式的 Memory • Key 是 Query • Value 是提炼的 Memory 文本	• Query → Embedding • 相似度检索 Top-K Key-Value Memory	• Executor Agent 推理，收集轨迹信息 • 获取任务完成的反馈信号	• Mem-Optimizer 从轨迹中提取经验 • 生成 Memory 操作 • 更新Memory Bank

任务：训练一个专有的Mem-Optimizer模型
在流式多任务交互过程中，提炼和管理记忆库，实现 Self-Evolve Agent

现有方法的问题及解决方案

UMEM 联合优化可泛化的记忆提炼和管理流程

现有做法

MemRL, Reasoningbank, EvolveR, Memp, ...

做法：

- 不训练 Mem-Optimizer: 轨迹中抽取经验和操作
- 优化训练 Mem-Optimizer：试图在“未来任务”上训练优化，**但是未来任务的反馈奖励无法传递给过去的训练过程**

问题：

- 记忆管理策略和记忆提炼环节**脱节**
- **未能成功显式**建模记忆的**可泛化性**

UMEM

Mem-Optimizer 负责提取和管理可泛化记忆

做法：

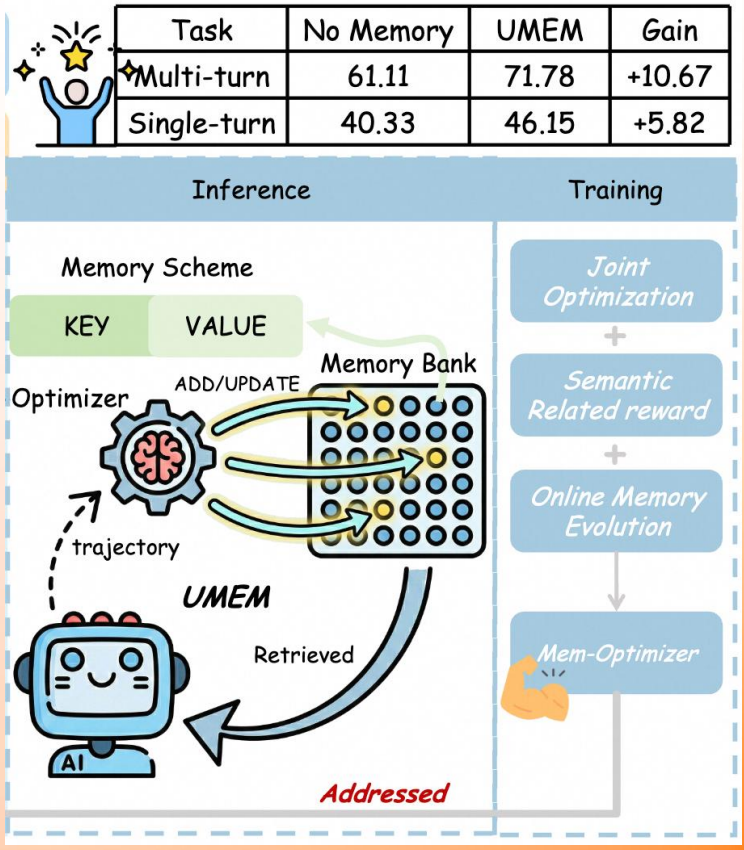
- **联合建模记忆提炼和管理：**端到端生成任务

<experience><value>...</value>

<operation>...</operation></experience>

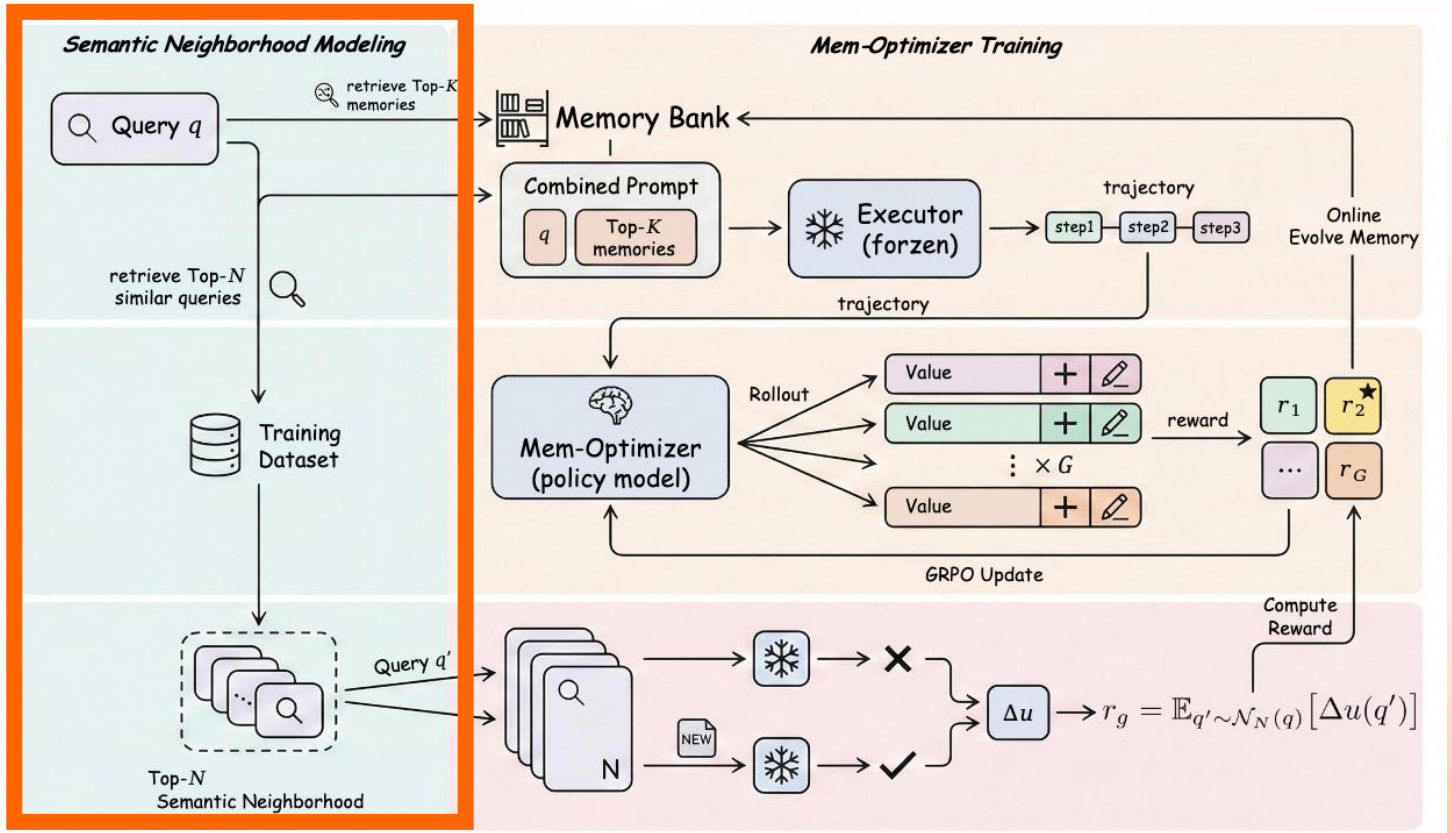
- **直接验证** Memory **可泛化性**
- Mem-Optimizer 与 Memory Bank **共同演化**
- 1/4/14B Mem-Optimizer → Memory 系统

Task	No Memory	UMEM	Gain
Multi-turn	61.11	71.78	+10.67
Single-turn	40.33	46.15	+5.82



UMEM: Unified Memory Extraction and Management Framework for Generalizable Memory

直接优化“未来任务”太难；UMEM 用相似任务近似“未来任务”

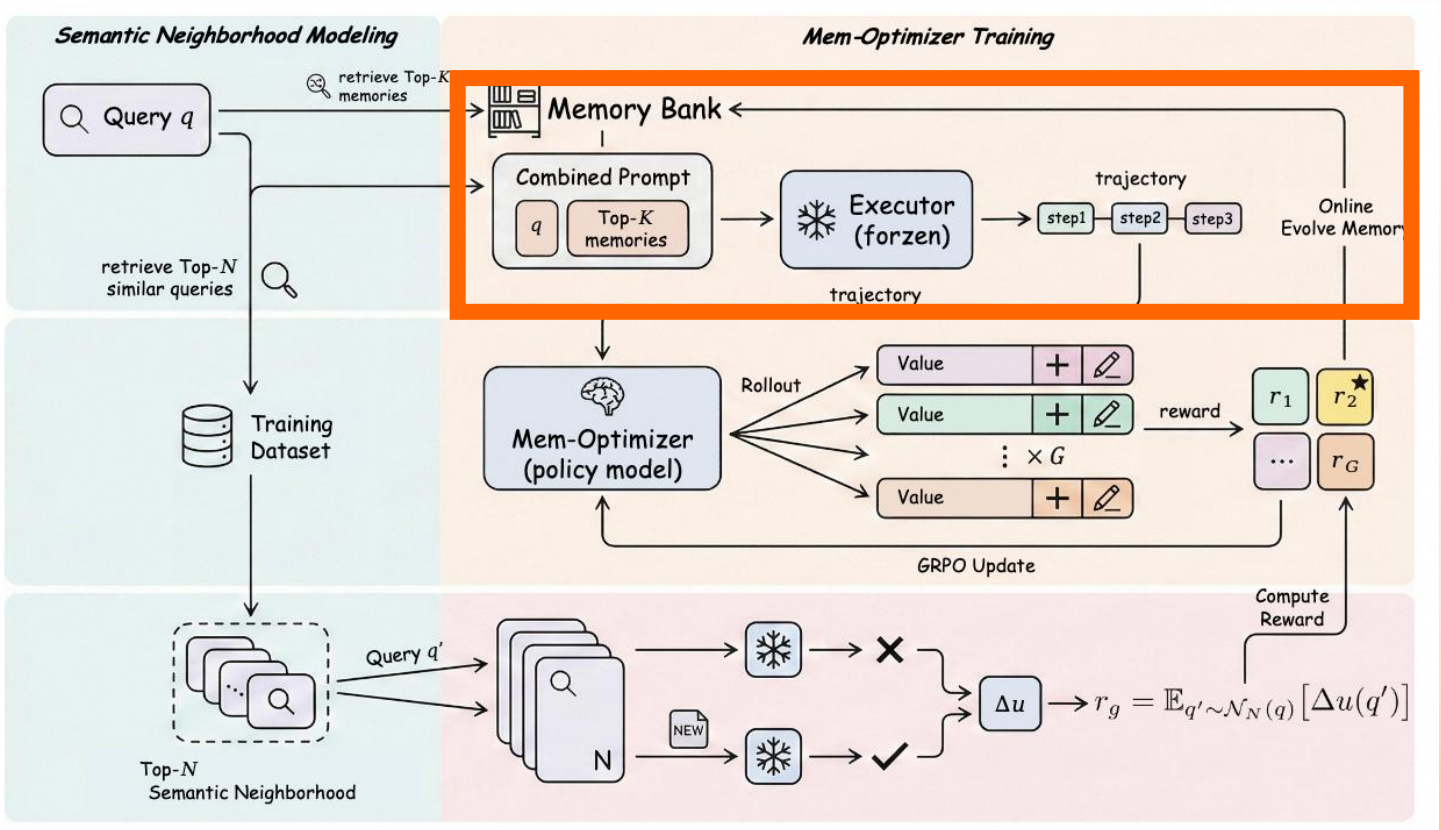


Semantic Neighborhood Modeling

- Query q 映射为语义向量
- 检索 Top-K 相似任务及其 Memory
- 训练过程中持续累积 Memory Bank

UMEM: Unified Memory Extraction and Management Framework for Generalizable Memory

直接优化“未来任务”太难；用相似任务近似“未来任务”

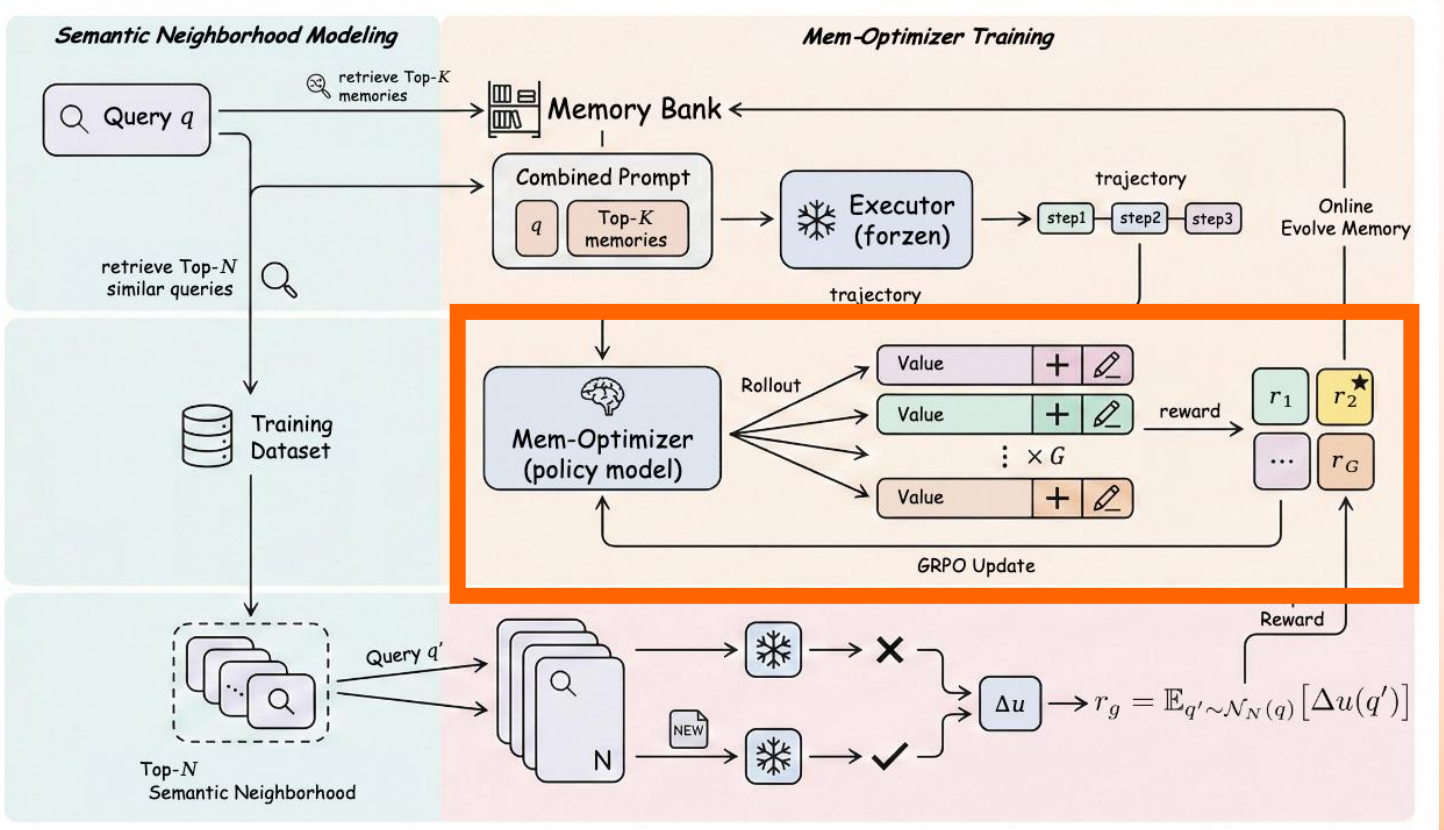


轨迹收集

- 拼接 Query q 和 Top-K 检索记忆
- Memory-Augment Execution: Agent 推理得到若干轨迹数据
- Executor Agent 不参与训练

UMEM: Unified Memory Extraction and Management Framework for Generalizable Memory

直接优化“未来任务”太难；用相似任务近似“未来任务”



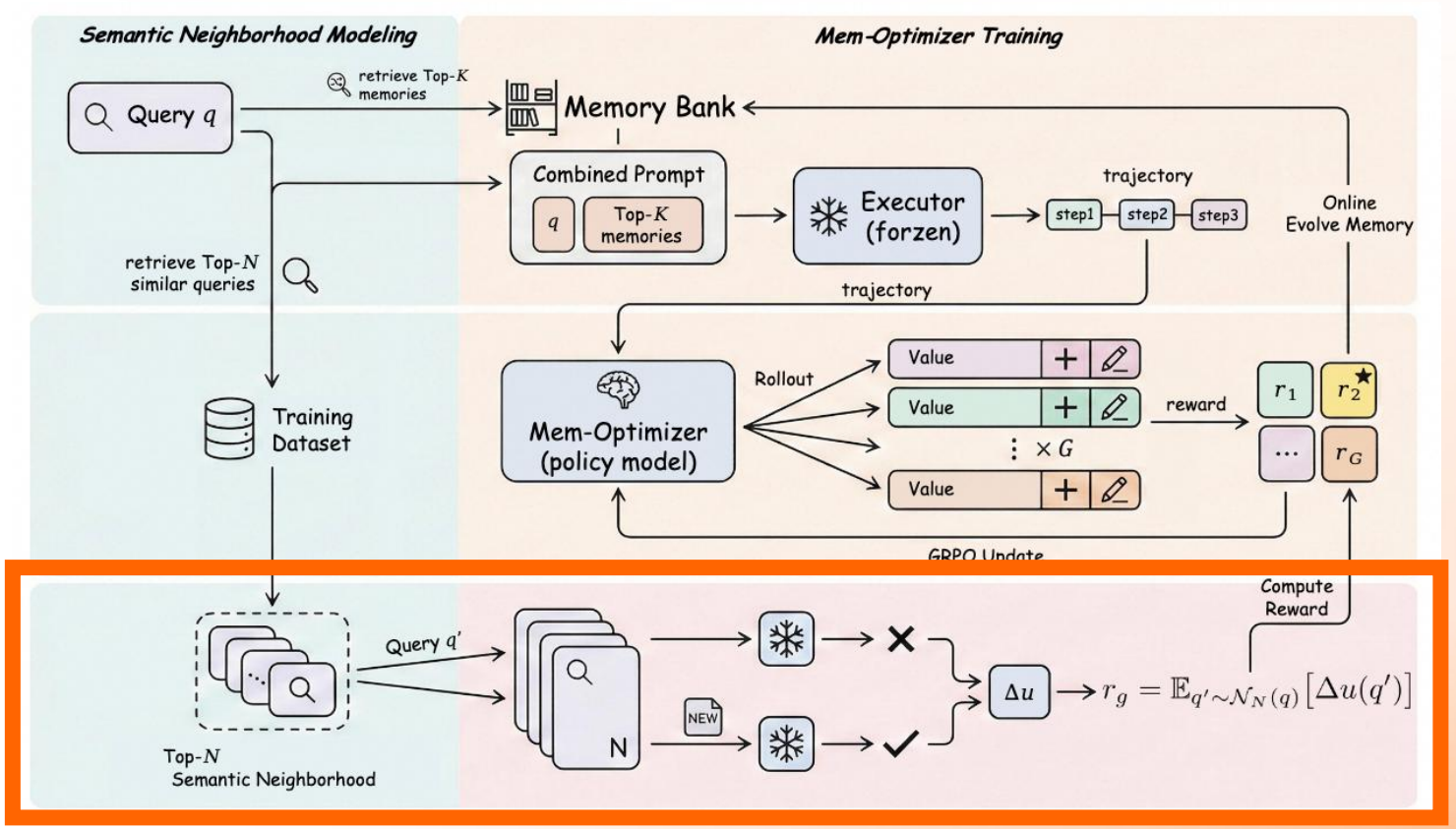
Mem-Optimizer 策略采样

- Mem-Optimizer (UMEM) 从轨迹中提炼记忆及对应操作

$$\{a_q^{(g)}\}_{g=1}^{G} \sim \pi_\phi(\cdot \mid q, \tau_q, \mathcal{B}_t^{topk})$$

UMEM: Unified Memory Extraction and Management Framework for Generalizable Memory

直接优化“未来任务”太难；用相似任务近似“未来任务”



面向可泛化记忆的奖励设计

- 相似任务验证：使用/不使用提炼记忆计算重新推理相似任务
- 计算在相似任务上的收益 Reward
- 成功率收益：

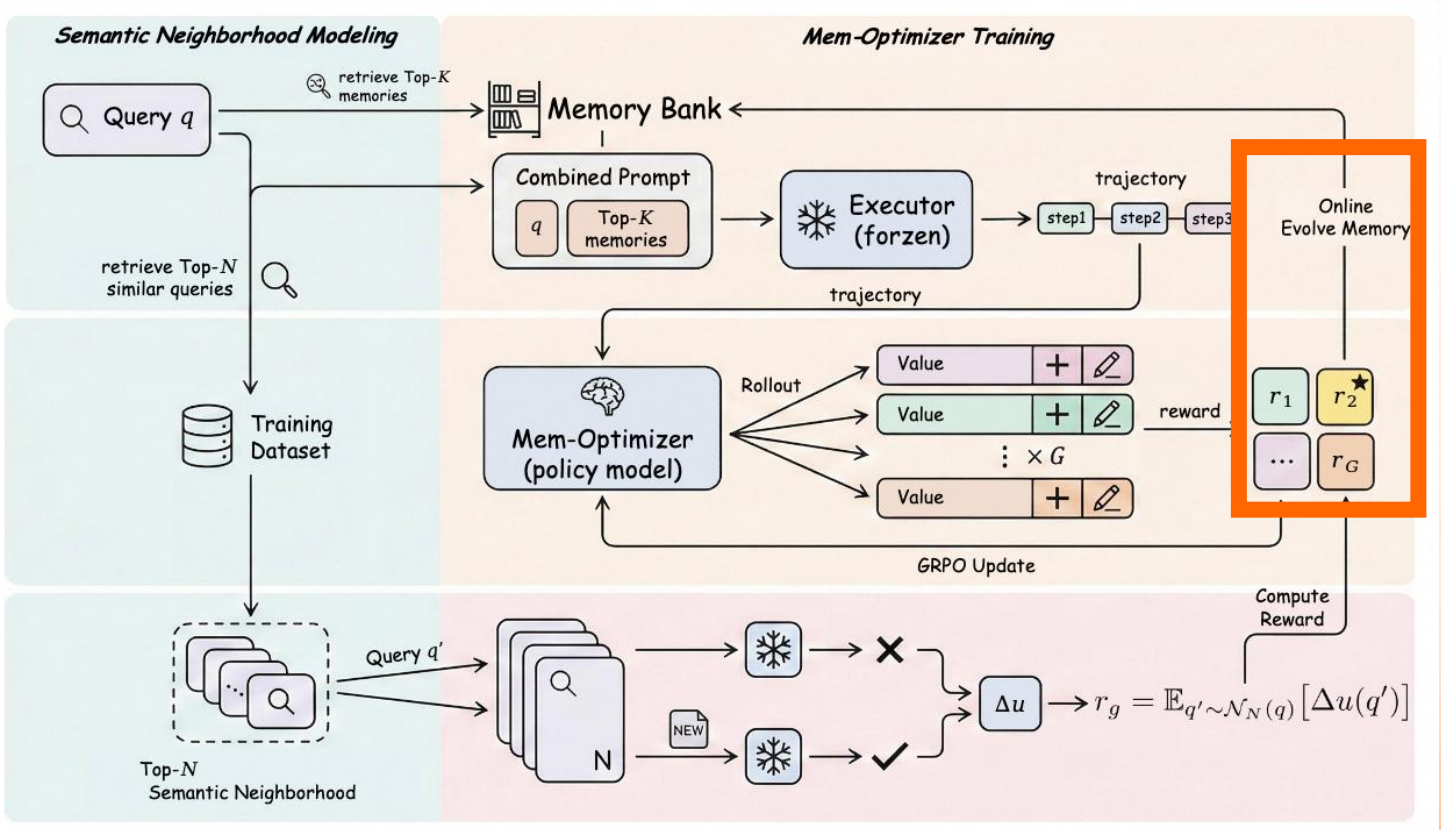
$$\mathcal{G}_{\text{succ}}(q') = c(\tau_{q'}^{\text{mem}}) - c(\tau_{q'}^{\text{ref}})$$

- 执行效率：

$$\mathcal{R}_{\text{eff}}(q') = (c_{\text{mem}} \cdot c_{\text{ref}}) \cdot \left(1 - \frac{\ell_{\text{mem}}^{(g)}}{\ell_{\text{ref}}}\right)$$

UMEM: Unified Memory Extraction and Management Framework for Generalizable Memory

直接优化“未来任务”太难；用相似任务近似“未来任务”



持续记忆演化

- 选择当前 GRPO 组内最佳提炼记忆和策略，更新 memory bank
- 拒绝静态memory上训练，随着训练过程持续演化 memory bank
- 让模型学会适应动态变化的 Memory Bank

03 / UMEM 实验结果分析

UMEM 是提炼可泛化经验记忆高效率模型

实验设置

验证：跨任务泛化能力、对不同 Executor Agent 的适配性、持续演化的稳定性

训练数据

- MMLU train set 2000条样本
- 预处理每条样本的邻域关系

经验记忆提取模型

- Llama-3.2-1B-Instruct
- Qwen3-4B-Instruct
- Qwen3-14B-Instruct

Executor Agent

- Qwen3-8B-Thinking
- GPT-5.1
- Gemini-2.5-Flash

评测任务

- AIME / GPQA / HLE / HotpotQA / ALFWorld
- 流式评估：评测过程中经验持续累积
- In-domain 和 Cross-domain 测试

实验分析：主实验

Models	Parameters	AIME	GPQA	HLE	HotpotQA	ALFWorld		Average
						SR	PR	
Frozen Executor: Qwen3-8B-Thinking								
No Memory	-	51.67	52.53	7.51	62.00	41.04	68.91	47.28
Self-RAG	8B	30.00	43.94	8.56	25.00	30.60	54.23	32.06
ReMem	8B	**61.67**	53.54	6.42	15.00	46.27	66.17	41.51
Memp	8B	46.67	49.49	**11.22**	62.00	44.78	69.78	47.32
Llama-3.2-1B-Instruct	1B	51.67	52.02	6.42	59.00	47.01	74.13	48.38
UMEM-Llama-3.2-1B (Ours)	1B	60.00$_{\uparrow 8.3}$	**54.04**$_{\uparrow 2.0}$	6.95$_{\uparrow 0.5}$	61.00$_{\uparrow 2}$	44.78$_{\downarrow 2.2}$	72.14$_{\downarrow 2.0}$	49.82$_{\uparrow 1.4}$
Qwen3-4B-Instruct	4B	55.00	53.54	6.95	60.00	40.30	65.92	46.95
UMEM-Qwen3-4B (Ours)	4B	58.33$_{\uparrow 3.3}$	52.02$_{\downarrow 1.5}$	8.02$_{\uparrow 1.1}$	**63.00**$_{\uparrow 3}$	**50.75**$_{\uparrow 10.5}$	**73.13**$_{\uparrow 7.2}$	**50.88**$_{\uparrow 3.9}$
Frozen Executor: GPT-5.1								
No Memory	-	40.00	57.57	6.95	39.00	61.94	66.67	45.36
Self-RAG	API	50.00	57.58	7.49	42.00	70.90	83.71	51.95
ReMem	API	30.00	62.63	8.56	43.00	73.13	79.60	49.49
Memp	API	45.00	62.12	**10.16**	52.00	77.61	81.34	54.71
Llama-3.2-1B-Instruct	1B	43.33	61.11	7.49	51.00	61.94	73.63	49.75
UMEM-Llama-3.2-1B (Ours)	1B	45.00$_{\uparrow 1.7}$	62.63$_{\uparrow 1.5}$	8.56$_{\uparrow 1.1}$	**55.00**$_{\uparrow 4}$	64.18$_{\uparrow 2.2}$	75.37$_{\uparrow 1.7}$	51.79$_{\uparrow 2.0}$
Qwen3-4B-Instruct	4B	46.67	62.63	8.02	52.00	70.90	78.86	53.18
UMEM-Qwen3-4B (Ours)	4B	**51.67**$_{\uparrow 5.0}$	**65.15**$_{\uparrow 2.5}$	8.56$_{\uparrow 0.5}$	54.00$_{\uparrow 2.0}$	**82.84**$_{\uparrow 11.9}$	**84.20**$_{\uparrow 5.3}$	**57.74**$_{\uparrow 4.6}$
Frozen Executor: Gemini-2.5-Flash								
No Memory	-	53.33	73.23	10.16	30.00	55.22	72.89	49.14
Self-RAG	API	56.67	71.72	10.16	42.00	59.70	74.50	52.46
ReMem	API	56.67	70.20	10.70	36.00	56.72	75.62	50.99
Memp	API	53.33	74.75	7.49	41.00	60.45	76.74	52.29
Llama-3.2-1B-Instruct	1B	51.67	73.23	10.16	42.00	53.73	71.27	50.34
UMEM-Llama-3.2-1B (Ours)	1B	58.33$_{\uparrow 6.7}$	71.72$_{\downarrow 1.5}$	**13.37**$_{\uparrow 3.2}$	42.00	58.96$_{\uparrow 5.2}$	77.24$_{\uparrow 6.0}$	53.60$_{\uparrow 3.3}$
Qwen3-4B-Instruct	4B	56.67	72.22	9.63	42.00	52.24	72.64	50.90
UMEM-Qwen3-4B (Ours)	4B	**60.00**$_{\uparrow 3.3}$	**76.26**$_{\uparrow 4.0}$	11.76$_{\uparrow 2.1}$	**45.00**$_{\uparrow 3.0}$	**61.19**$_{\uparrow 9.0}$	**78.61**$_{\uparrow 6.0}$	**55.47**$_{\uparrow 4.6}$

Qwen3-4B 版本上优化的 UMEM 在三个 executor **上取得了最优平均表现**

UMEM-Qwen3-4B 提炼记忆优于 ReMem和 Memp + GPT-5.1 和 Gemini-2.5-Flash 模型，**平均提升 3.03% 和 3.18%**

实验分析：更多的baseline对比

Method	GPT-5.1	Gemini-2.5-Flash
No Memory	45.36	49.14
ReasoningBank	53.76 ±0.82	54.39 ±0.59
MemRL	45.91 ±0.59	51.67 ±0.87
EvolveR	55.34 ±0.58	52.42 ±0.47
Memp	54.24 ±0.73	51.60 ±0.58
Qwen3-4B-Instruct	53.67 ±0.49	51.00 ±0.91
UMEM-Qwen3-4B (Ours)	**58.10 ±0.72**	**55.46 ±0.82**

在5个benchmark上的平均结果显示：**UMEM-Qwen3-4B 持续稳定优于 SOTA Memory 系统**

实验分析：UMEM 可扩展性分析

Model	Param	AIME	GPQA	HLE	Hotpot	ALF SR	ALF PR	Avg
FROZEN EXECUTOR: QWEN3-8B-THINKING								
No Memory	-	51.67	52.53	7.51	62.00	41.04	68.91	47.28
Llama-3.2-1B	1B	51.67	52.02	6.42	59.00	47.01	74.13	48.38
UMEM-1B (Ours)	1B	60.00 ↑8.3	54.04 ↑2.0	6.95	61.00	44.78	72.14	49.82
Qwen3-4B	4B	55.00	53.54	6.95	60.00	40.30	65.92	46.95
UMEM-4B (Ours)	4B	58.33 ↑3.3	52.02	8.02 ↑1.1	63.00 ↑3.0	50.75 ↑10.5	73.13 ↑7.2	50.88
Qwen3-14B	14B	57.22	53.71	7.84	61.67	47.51	70.69	49.77
UMEM-14B (Ours)	14B	60.56 ↑3.3	55.05 ↑1.3	8.74 ↑0.9	63.67 ↑2.0	52.24 ↑4.7	74.55 ↑3.9	52.47

- UMEM-1B 和 4B 模型在部分任务上出现**性能下降**
- 但是该现象随着 UMEM 尺寸增加逐渐改善；UMEM-14B 模型提炼记忆的能力**持续增强，始终优于基线**

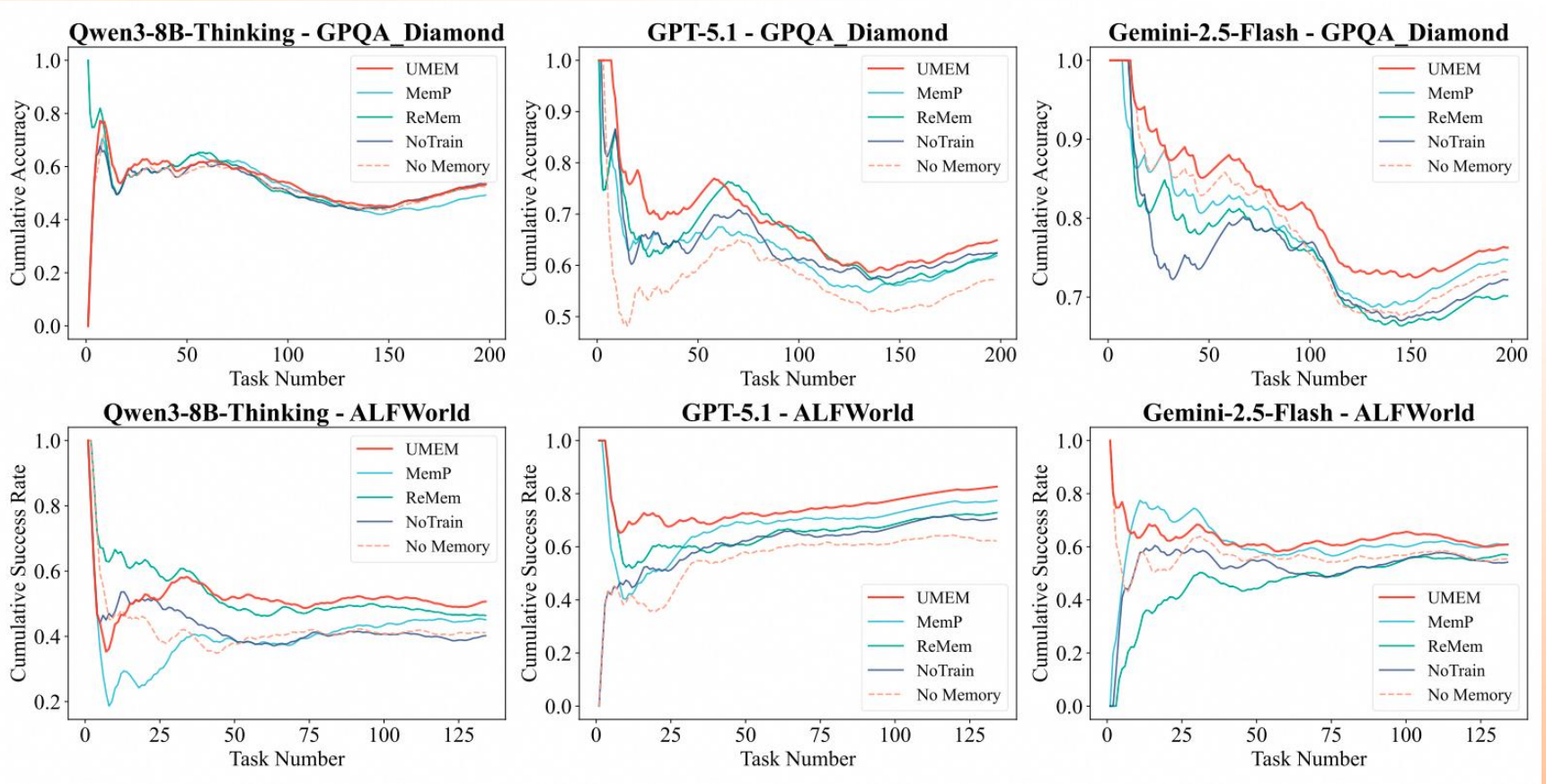
实验分析：UMEM 消融实验分析

Method	GPT-5.1						Qwen3-8B-Thinking					
	AIME (Acc.)	GPQA (Acc.)	HLE (Acc.)	HotpotQA (Acc.)	ALFWorld SR	ALFWorld PR	AIME (Acc.)	GPQA (Acc.)	HLE (Acc.)	HotpotQA (Acc.)	ALFWorld SR	ALFWorld PR
Joint Optimization Components												
UMEM (Full Method)	**51.67**	**65.15**	8.56	54.00	**82.84**	84.20	**58.33**	**53.54**	8.02	**63.00**	50.75	**73.13**
w/o Extraction Opt.	45.00$_{\downarrow 6.7}$	59.60$_{\downarrow 5.6}$	5.88$_{\downarrow 2.7}$	51.00$_{\downarrow 3.0}$	76.12$_{\downarrow 6.7}$	80.72$_{\downarrow 3.5}$	55.00$_{\downarrow 3.3}$	51.01$_{\downarrow 2.5}$	8.02$_{\uparrow 0.0}$	61.00$_{\downarrow 2.0}$	45.53$_{\downarrow 5.2}$	67.79$_{\downarrow 5.3}$
w/o Management Opt.	48.33$_{\downarrow 3.3}$	64.65$_{\downarrow 0.5}$	**9.09**$_{\uparrow 0.5}$	**55.00**$_{\uparrow 1.0}$	80.60$_{\downarrow 2.2}$	**84.33**$_{\uparrow 0.1}$	56.67$_{\downarrow 1.7}$	53.03$_{\downarrow 0.5}$	6.95$_{\downarrow 1.1}$	63.00$_{\uparrow 0.0}$	44.70$_{\downarrow 6.1}$	69.03$_{\downarrow 4.1}$
w/o SNM	41.67$_{\downarrow 10.0}$	64.14$_{\downarrow 1.0}$	6.95$_{\downarrow 1.6}$	52.00$_{\downarrow 2.0}$	79.10$_{\downarrow 3.7}$	81.09$_{\downarrow 3.1}$	55.00$_{\downarrow 3.3}$	50.00$_{\downarrow 3.5}$	7.49$_{\downarrow 0.5}$	60.00$_{\downarrow 3.0}$	**52.99**$_{\uparrow 2.2}$	72.30$_{\downarrow 0.8}$
Sensitivity to Neighborhood Size												
UMEM ($N = 3$, Ours)	**51.67**	**65.15**	**8.56**	**54.00**	**82.84**	**84.20**	**58.33**	**53.54**	8.02	**63.00**	**50.75**	73.13
$N = 1$ (Too Narrow)	48.33$_{\downarrow 3.3}$	63.13$_{\downarrow 2.0}$	7.49$_{\downarrow 1.1}$	50.00$_{\downarrow 4.0}$	78.36$_{\downarrow 4.5}$	81.34$_{\downarrow 2.9}$	51.67$_{\downarrow 6.7}$	51.52$_{\downarrow 2.0}$	8.02$_{\uparrow 0.0}$	59.00$_{\downarrow 4.0}$	41.79$_{\downarrow 9.0}$	67.66$_{\downarrow 5.5}$
$N = 5$ (Too Broad)	46.67$_{\downarrow 5.0}$	64.65$_{\downarrow 0.5}$	7.49$_{\downarrow 1.1}$	52.00$_{\downarrow 2.0}$	81.34$_{\downarrow 1.5}$	84.08$_{\downarrow 0.1}$	58.33$_{\uparrow 0.0}$	52.53$_{\downarrow 1.0}$	**9.63**$_{\uparrow 1.6}$	62.00$_{\downarrow 1.0}$	42.54$_{\downarrow 8.2}$	**73.26**$_{\uparrow 0.1}$

- 不优化记忆提炼/管理：**训练性能大幅下降**
- 去除语义邻域奖励建模导致性能下降：**证明针对可泛化记忆提取的必要性**

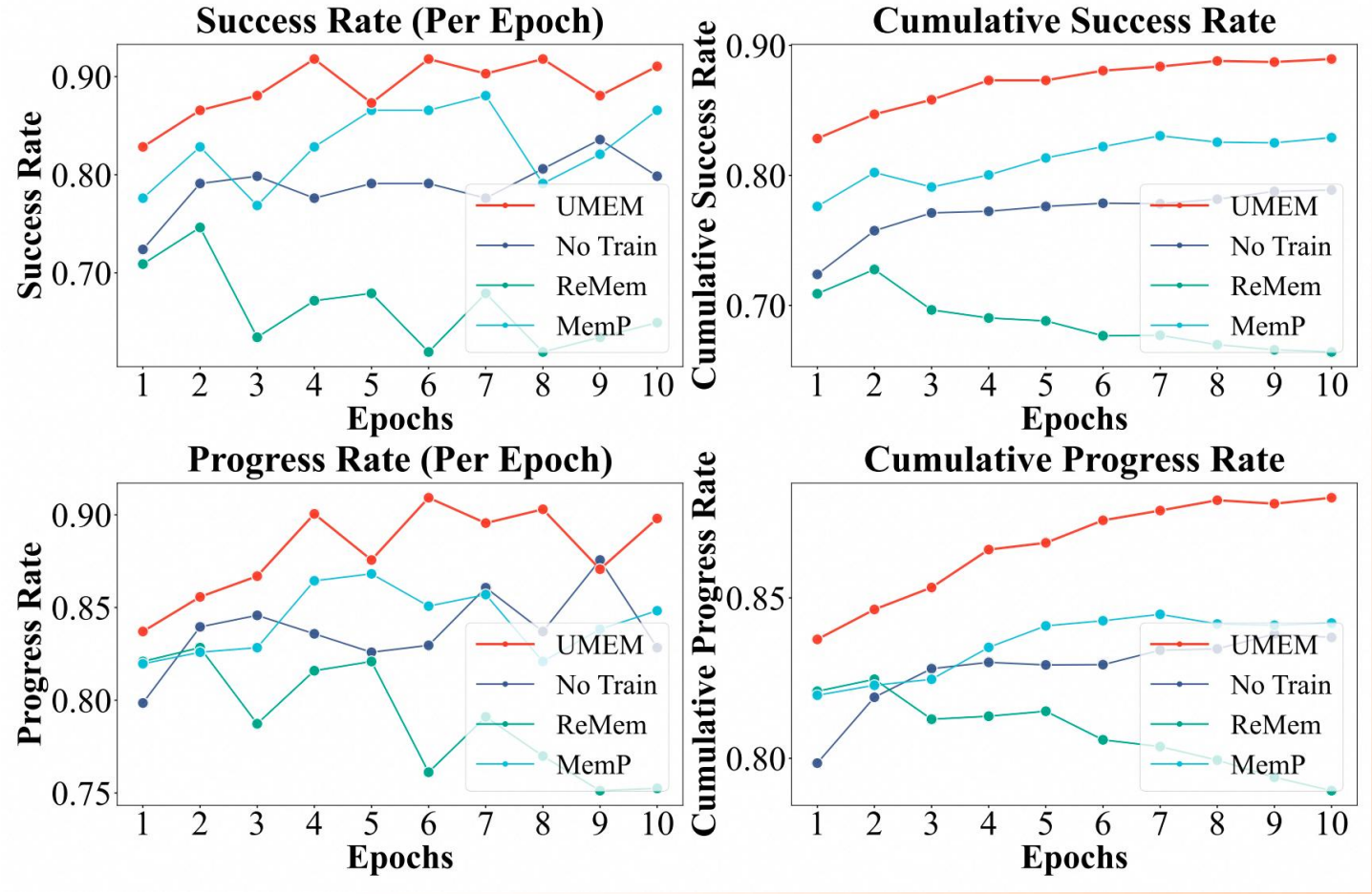
- 过小的邻域**难以捕捉和近似任务的迁移和泛化**
- 过大的领域导致**记忆噪音影响奖励计算和优化**

实验分析：UMEM 在长期 Self-Evolve 过程中更稳定



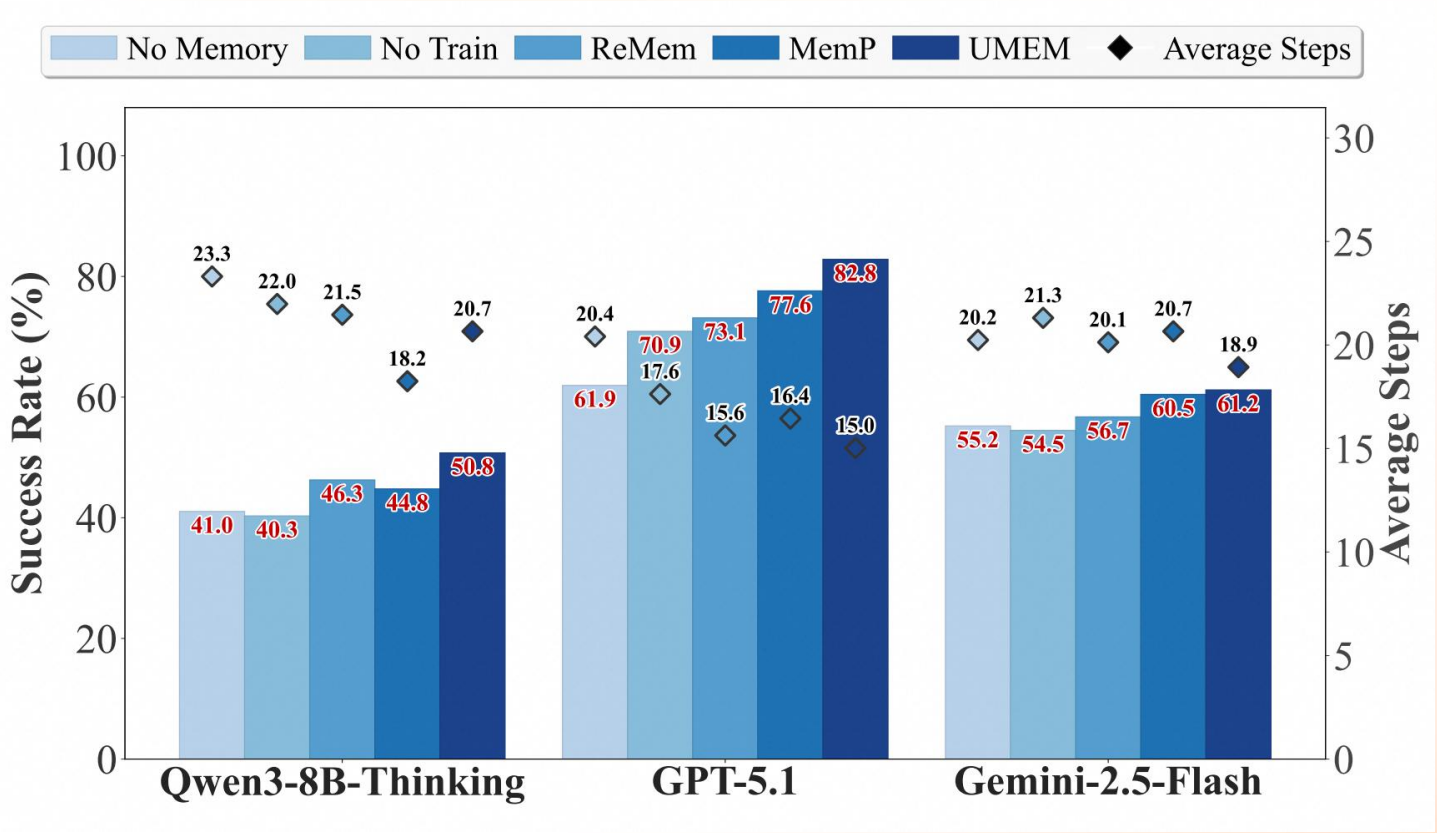
- **流式评估的挑战性：memory会不断累积，错误也会被持续放大**
- UMEM的累积曲线退化更慢，后期优势更加明显；证明提炼的记忆错误更少
- MemP 和 ReMem出现了明显的局部优化，说明记忆泛化性较差

实验分析：UMEM 的 Test-time Scaling



- **拉长演进尺度（ALFWorld 10 轮持续自演进）**
  - 部分 Baseline 出现性能退化
  - UMEM 在 per-epoch 与 cumulative success 指标上都保持大幅领先，即使出现波动也能快速稳定回复

实验分析：UMEM 更高的执行效率



- ALFWorld 任务完成成功率和执行效率的对比分
  - UMEM 的 success rate 一致更好，且完成任务的平均步数显著更低
  - 任务的成功率高并非来源于test-time scaling，而是学会使用记忆经验提高决策效率和质量

实验分析：Cross-Task 泛化分析

Method	single-bench avg	cross-bench avg
UMEM-4B	57.74	58.33
No-Train	53.18	52.92
No-Memory	45.36	45.36

CTU 和 ERR 对比分析

Method	Cross-Task Utility (CTU)	Error Recovery Rate (ERR)
UMEM-4B	41.47%	23.17%
No-Train	39.29%	17.80%

- **积累的经验记忆是否可以跨任务泛化**
  - UMEM在跨任务经验积累场景下性能进一步提升，GPT-5.1 Executor 下从 57.74% → 58.33%； Baseline 表现出性能损失
  - Cross-Task Utility (CTU) 和 Error Recovery Rate (ERR) 证明 UMEM记忆可用性更高、纠正更多的错误

实验分析：UMEM 灾难性遗忘评估

训练后的 UMEM 模型是否只是特化到 Memory 提取和管理任务上？

灾难性遗忘评估 (Catastrophic Forgetting Evaluation)

Model	IFEval (Strict Avg)	HumanEval (Pass@1)
Qwen3-4B	85.53%	63.53%
UMEM-4B	87.33%	65.85%

- UMEM-4B 模型训练后在部分通用benchmark，如IFEval和HumanEval表现出性能提升
- **在实际使用 OpenClaw 等个人助手的经验积累过程中可以直接替换传统的Memory提取管理workflow为UMEM；实现用较低的开销获取更高的记忆库质量**

实验分析：Case Study

Task & Retrieved Experiences

[Task Goal] Put a clean cloth in countertop.

[Retrieved Experiences]

`[0]` *key*: "clean some plate and put it in countertop."
value: To clean a plate and place it on a countertop, first locate the plate (e.g., in a cabinet), pick it up, go to the sinkbasin, clean it using the command "clean [plate] with sinkbasin", and **only after cleaning** move it to a countertop. Skipping the cleaning step or placing a dirty plate on the countertop results in failure.

`[1]` *key*: "put a clean knife in countertop."
value: To put a clean knife on a countertop, first locate the knife (e.g., on a countertop or in storage), pick it up, go to the sinkbasin, clean it using "clean [knife] with sinkbasin", and **only after cleaning** move it to a countertop. Skipping the cleaning step results in failure.

`[2]` *key*: "clean some pan and put it in countertop."
value: To clean a pan and place it on a countertop, first locate the pan, verify its identity by examining it, pick it up, go to the sinkbasin, clean it using "clean [pan] with sinkbasin", and **only after cleaning** move it to a countertop.

UMEM Enhanced Agent: w/ Memory (Success)

[Environment] You are in the middle of a room. You see a cabinet 4, cabinet 3, cabinet 2, cabinet 1, countertop 1, garbagecan 1, handtowelholder 2, handtowelholder 1, sinkbasin 2, sinkbasin 1, toilet 1, toiletpaperhanger 1, and towelholder 1.

[Execution Trajectory]

Step 1: `go to handtowelholder 1` → You see a handtowel 1. — *Exploring...*
Step 2: `take handtowel 1 from handtowelholder 1` → You pick up the handtowel 1. — *Wrong object*
Step 3: `go to sinkbasin 2` → You arrive at sinkbasin 1.
Step 4: `move handtowel 1 to sinkbasin 1` → You move the handtowel 1 to the sinkbasin 1.
Step 5: `take handtowel 1 from sinkbasin 1` → You pick up the handtowel 1.
Step 6: `go to countertop 1` → You see a candle 1, a cloth 3, and a spraybottle 1. — *Found cloth!*
Step 7: `examine countertop 1` → On the countertop 1, you see a candle 1, a cloth 3, and a spraybottle 1.
Step 8: `move handtowel 1 to countertop 1` → You move the handtowel 1 to the countertop 1.
Step 9: `take cloth 3 from countertop 1` → You pick up the cloth 3. — *Correct object*
Step 10: `go to sinkbasin 1` → You arrive at sinkbasin 1. — *Key Step*
Step 11: `clean cloth 3 with sinkbasin 1` → You clean the cloth 3 using the sinkbasin 1. — *Key Step*
Step 12: `go to countertop 1` → You arrive at countertop 1.
Step 13: `move cloth 3 to countertop 1` → You move the cloth 3 to the countertop 1.

[Result] SUCCESS **Total Steps: 13** The agent explores, finds the correct object, and applies the cleaning pattern from memory.

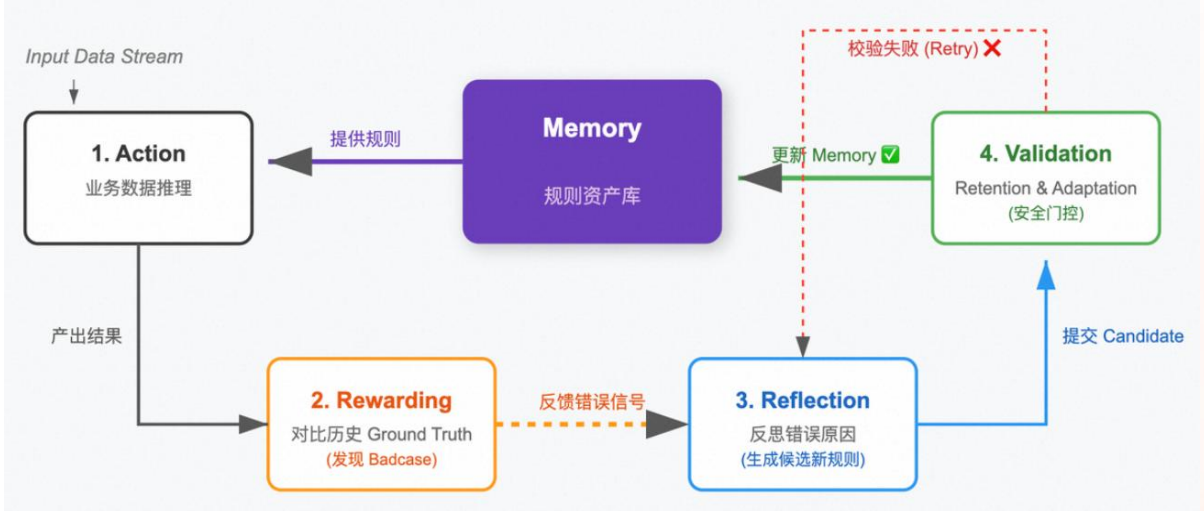
- 任务：把一块干净的布放在台面上
- 检索 UMEM 的经验记忆：
  - 清洗某个盘子并把它放在台面上
  - 把一把干净的刀放在台面上
  - 清洗某个平底锅并把它放在台面上。
- UMEM 学到了可迁移的程序性经验
  - 能从类似经验中抽出通用 pattern
  - 找到物体 → 去 sink → clean → place

04 / 业务应用与成果

Marco DeepResearch 项目

UMEM 在跨境电商业务场景的应用

大规模电商场景的商品审核场景下，规则往往细微且持续动态变化
人工审核开销大，Self-Evolve Agent Memory 是重要技术解决手段



应用场景 (Application Scenarios)

场景类型	具体任务
透明图检测	白底图审核 (White-background Image Auditing)
情感标题审核	短标题审核 (Short Title Reviews)
品牌商品审核	正品/仿品识别 (Authentic/Counterfeit Detection)

业务效果 (Business Impact)

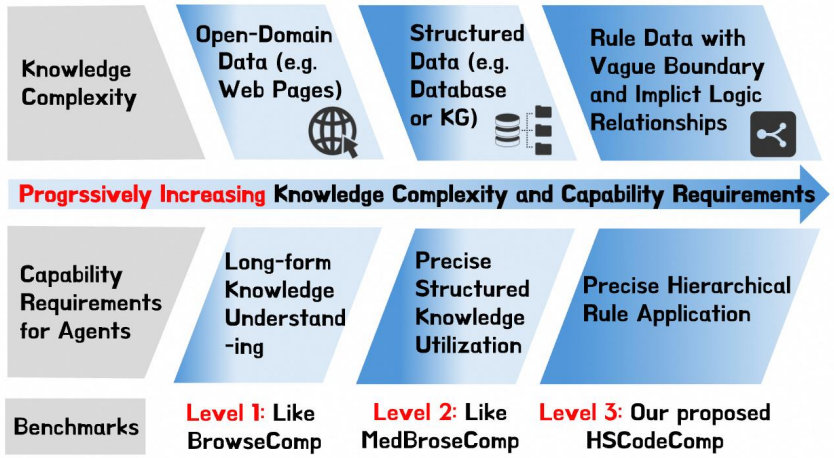
指标维度	对比基准	UMEM 效果
效率提升	人工调优 3-5 天	10 分钟 自主循环
白底图审核	人工优化基线	+11% 准确率提升
短标题审核	人工优化基线	+2% 准确率提升

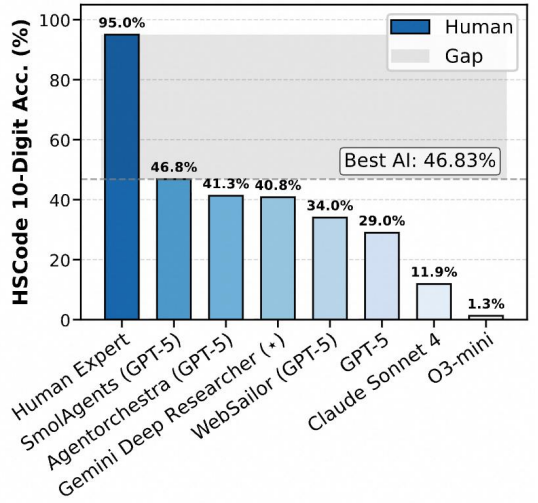
UMEM 属于 Marco DeepResearch 项目

- 跨境电商业务提炼的关于 Self-Evolve Agent、Agentic InfoSeeking 等部分成果已被 ACL 2026 接受
- https://github.com/AIDC-AI/Marco-DeepResearch

HSCodeComp (ACL 2026 Award Candidate)

边界模糊且逻辑交织的层次化规则指令对Agent是挑战





商品 HSCode 编码预测任务：最佳Agent和关务人类专家差距极大（95% vs 46.8%）

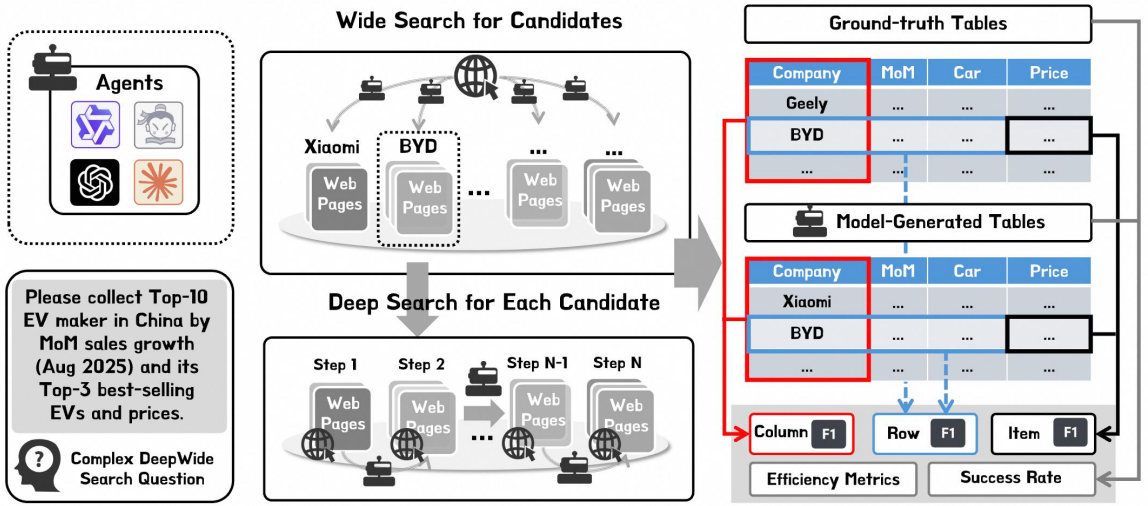
Agent 不擅长应对模糊且逻辑交织的层次化规则

UMEM 属于 Marco DeepResearch 项目

- 跨境电商业务提炼的关于 Self-Evolve Agent、Agentic InfoSeeking 等部分成果已被 ACL 2026 接受
- https://github.com/AIDC-AI/Marco-DeepResearch

DeepWide Search

兼具深度和宽度挑战的真实场景复杂信息搜索



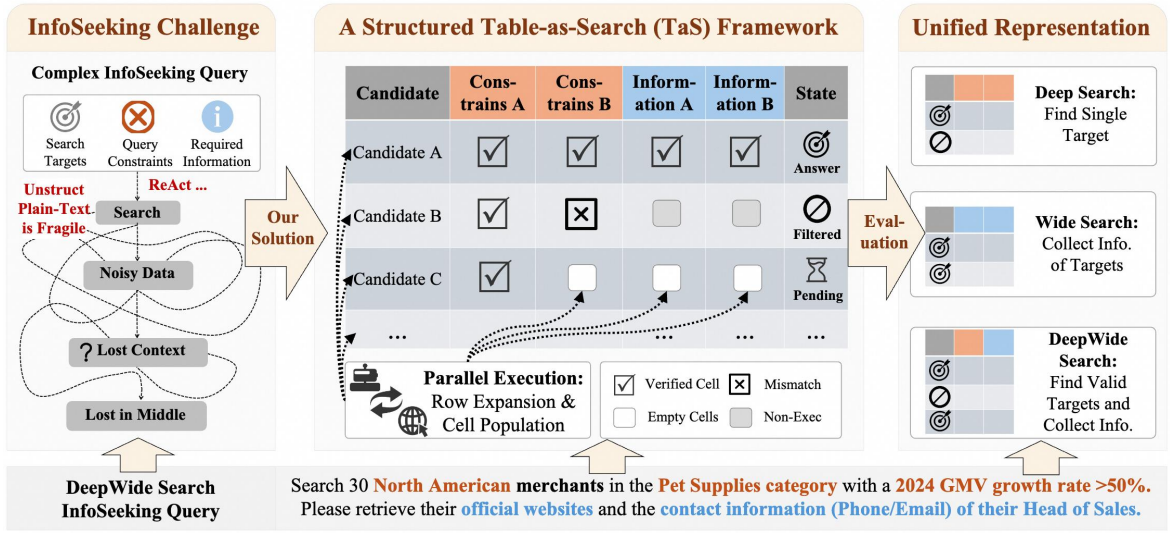
真实业务场景的信息搜素任务往往都是既深又宽的
而不单纯是Deep Search or Wide Search

UMEM 属于 Marco DeepResearch 项目

- 跨境电商业务提炼的关于 Self-Evolve Agent、Agentic InfoSeeking 等部分成果已被 ACL 2026 接受
- https://github.com/AIDC-AI/Marco-DeepResearch

Table-as-Search (ACL 2026)

针对Agentic InfoSeeking设计的Harness系统



围绕复杂Agentic InfoSeeking任务，尤其是深宽搜索任务
设计一套基于表格的结构化 Harness 系统，在Deep/Wide/DeepWide 任务上SOTA

UMEM 属于 Marco DeepResearch 项目

- 跨境电商业务提炼的关于 Self-Evolve Agent、Agentic InfoSeeking 等部分成果已被 ACL 2026 接受
- https://github.com/AIDC-AI/Marco-DeepResearch

Marco-DeepResearch

验证驱动的设计使 8B 模型 追上 30B 模型效果

在数据合成、轨迹收集、Test-time Scaling 阶段显式引入Verification机制
有效榨取 8B-scale 模型的 Deep Search 潜力，追上部分 30B 模型效果

Thanks