SELT: 利用任务分解进行大型语言模型自我评估树搜索

Mengsong Wu[1,2], Di Zhang[2], Yuqiang Li[2], Dongzhan Zhou[2], Wenliang Chen[*1]

[1]Soochow University, [2]Shanghai Artificial Intelligence Laboratory

radi.cat@qq.com, { zhangdi,liyuqiang,zhoudongzhan } @pjlab.org.cn, wlchen@suda.edu.cn

Abstract

虽然大型语言模型（LLM）在广泛的应用中取得了显著的成功，但它们在复杂的推理任务中表现往往会下降。在这项工作中，我们引入了 SELT（自我评估 LLM 树搜索），这是一种利用修改后的蒙特卡罗树搜索（MCTS）来增强 LLM 推理的新框架，无需依赖外部奖励模型。通过重新定义上置信界得分以符合 LLM 的内在自我评估能力，并将推理过程分解为在每个节点上增强语义聚类的原子子任务，SELT 有效地平衡了探索和利用，减少了冗余的推理路径，并缓解了幻觉。我们在具有挑战性的基准测试中验证了我们的方法，包括基于知识的 MMLU 和工具学习数据集 Seal-Tools，其中 SELT 在回答准确性和推理稳健性方面相比基线方法取得了显著提升。值得注意的是，我们的框架无需任务特定的微调即可运行，展示了在不同推理任务中的强大普遍性。相关结果和代码可以在 https://github.com/fairyshine/SELT 找到。

1 引言

大型语言模型（LLMs）在压缩和记忆大量知识方面展示了卓越的能力 (Delétang et al., 2023; Martino et al., 2023) 。然而，LLMs 在复杂推理任务中表现出显著的弱点，经常给出不一致的答案 (Parmar et al., 2024) 。为了增强 LLMs 的推理能力，研究人员探索了各种方法，例如上下文学习（In-Context Learning, ICL, Brown et al., 2020 ）、思维链（Chain-of-Thought, CoT, Wei et al., 2022 ）提示和基于强化学习（RL）的微调。尽管这些方法取得了成功，但它们通常严重依赖于人工设计的推理模板或需要昂贵的奖励模型，这些模型在特定领域的数据上进行了广泛的微调。更好的测试时扩展方法仍然是需要的。最近，蒙特卡罗树搜索（Monte Carlo Tree Search, MCTS, Browne et al., 2012 ）作为一种有前途的技术出现，用于通过系统地调查潜在的推理路径来增强 LLM 推断。然而，目前的方法 (Zhao et al., 2024; Hao et al., 2023) 严重依赖于微调外部奖励模型来评估中间推理步骤，将其应用限制在具有大量任务特定数据的领域，例如数学。它引入了额外的训练开销和来自不完美奖励信号的潜在偏差。

在这项工作中，我们提出了 SELT（自我评估大型语言模型搜索），这是一种为大型语言模型设计的新颖蒙特卡罗树搜索框架，通过两项关键创新来克服这些挑战。首先，我们通过重新定义树的上置信界限（UCT）评分机制的两个部分来修改原始 MCTS 算法，以便符合大型语言模型内在自我评估的能力，消除了外部奖励模型的需求。这种新的评分机制能够更均衡地探索和利用推理路径。其次，我们将推理过程分解为多个原子级的大型语言模型任务，使得大型语言模型可以完成更小、更易管理的子任务。为动态分组语义等价的解决方案，我们在每个树节点引入了一种语义聚类机制，从而减少冗余并使选择更高质量的代表性答案成为可能。通过这些代表性答案，大型语言模型能够更好地自我评估新的答案，如图 **??** 所示，而无需其他奖励模型。该方法不仅提高了搜索效率，而且在保护推理路径多样性的同时缓解了“幻觉陷阱”。我们在两个具有挑战性的基准上评估了 SELT：需要基于领域知识进行多步推理的知识问答数据集 MMLU (Hendrycks et al., 2020) 和涉及与外部工具动态交互的工具学习数据集 Seal-Tools (Wu et al., 2024) 。实验结果表明，SELT 优于基线方法，包括 CoT 和标准 MCTS，在答案准确性和推理稳健性上取得了显著的提升。所提出的两项创新均有效地增强了复杂问题的 LLM 回答能力。值得注意的是，我们的框架不需要进行特定任务的微调，强调了其在各种推理任务中的普遍性。我们的贡献有三方面：

1) 基于自我评估的搜索：我们提出了一种自我评估驱动的 MCTS 适应方法，用于 LLM，消除了对外部奖励模型的依赖，并使 LLM 能够本质上引导树搜索。

2) LLM 推理分解：我们将原始推理任务简化为原子的子任务，从而提高了整体性能。为了简化获得的推理路径，我们应用语义感知的聚

*Corresponding Author.

类方法来识别和优先处理多样化的高质量解决方案。
3) 实验验证：大量实验证实，在数学、常识和程序推理任务中，SELT 优于基线方法，突显其无需特定任务微调的普遍性。

2 预备知识

2.1 符号列表

为了更好地说明用于 LLM 推理的树搜索算法，可能出现在伪代码中的符号列在表 1 中。

Symbol	Description
v	One node in the reasoning trees which contain its structure information and the state.
v_r	Root node in the reasoning tree.
s	State of the specifc node which contains information about the LLM inference.
Δ	Reward score given by the LLM.
C_β	Constant controlling the bayes averaging.
C_p	The exploration constant of MCTS, usually set to $\sqrt{2}$.
$Q(v)$	Total rewards of the node v .
$N(v)$	Visit times of the node v .
$r(s)$	The reward calculation of the state s .

Table 1: 论文中出现的相关符号。

2.2 蒙特卡洛树搜索

蒙特卡洛树搜索（MCTS，Browne et al., 2012 ）是一种启发式搜索算法，以其在复杂的、随机的环境中进行决策过程中的高效性而闻名。基于蒙特卡洛模拟和树搜索的原理，MCTS 已成为顺序决策问题的基石方法，尤其是在具有巨大状态空间和不完全信息的领域中。

如算法 1 所示，它通过四个递归阶段运作：选择（func tree policy）、扩展（func expand）、模拟（func default policy）和反向传播（func backup）。在选择阶段，算法通过应用一个策略——通常是树的上置信界限（UCT，(Gelly and Wang, 2006) ）——在探索较少访问的节点和优先考虑高奖励估计节点之间进行权衡，从根节点遍历到叶节点。

$$\begin{aligned} \mathrm{S}_{\mathrm{UCT}}(\mathrm{v}') &= \mathrm{S}_{\mathrm{Exploit}}(\mathrm{v}') + \mathrm{S}_{\mathrm{Explore}}(\mathrm{v}') \\ &= \frac{Q(v')}{N(v')} + C_p\sqrt{\frac{2\ln N(v)}{N(v')}} \end{aligned} \tag{1}$$

Algorithm 1: UCT 蒙特卡洛树搜索

Function: MCTS_UCT(s_0 , T)
Input: original state s_0 , search steps T
Output: best leaf state $s*$
new root node v_r of the tree;
v_r .state $\leftarrow s_0$;
$v_0 \leftarrow v_r$;
while current search steps $< T$ **do**
  $v_l \leftarrow$ tree_policy(v_0);
  $\Delta \leftarrow$ default_policy(v_l .state);
  backup(v_l, Δ);
end
$v_{best} \leftarrow v_0$;
while not v_{best} .terminal() **do**
  $v_{best} \leftarrow$ best_child(v_{best} , 0);
end
$s* \leftarrow v_{best}$ .state;
Return: $s*$

当到达叶节点时，通过添加一个或多个子节点来扩展树，从而逐步完善搜索空间。然后从新扩展的节点开始执行蒙特卡罗模拟（或回滚），通常通过随机或轻量级启发式模拟来估计路径的潜在奖励。最后，模拟结果通过遍历的节点进行反向传播，以更新它们的统计度量（例如，访问次数和累计奖励），从而改善未来的搜索迭代。

3 SELT: 自我评估树搜索

3.1 LLM 任务分解

大型语言模型在各种任务上取得了显著进展，包括文本翻译和特定领域的问题解答（QA）。虽然它们能够处理广泛的任务，但其响应的准确性可能有所不同。我们的初步实验表明，问题的措辞可以影响模型对同一查询的响应。为了增强大型语言模型的推理能力，我们将复杂任务分解为更清晰、更易于管理的基本子任务。

我们将任务分为以下类型：

- 对错（真假陈述）：评估和分析信息以形成判断或结论。
- 选择：从一组备选项中选择最合适的选项。
- FITB（填空题）：通过插入缺失的词或短语来完成句子或段落。
- SA（简短回答）：根据可用信息提供特定问题的答案。

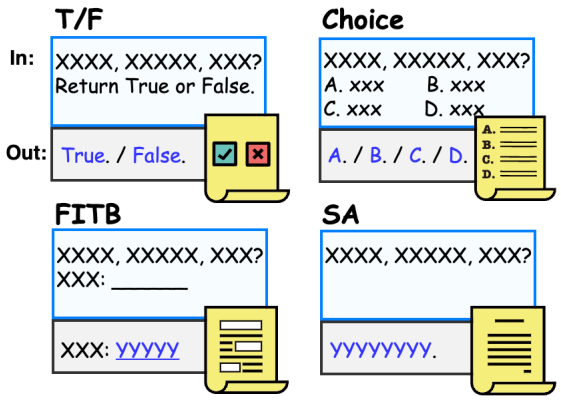

Figure 1: 四种大型语言模型的任务类型。(T/F, 多项选择，填空，简答)

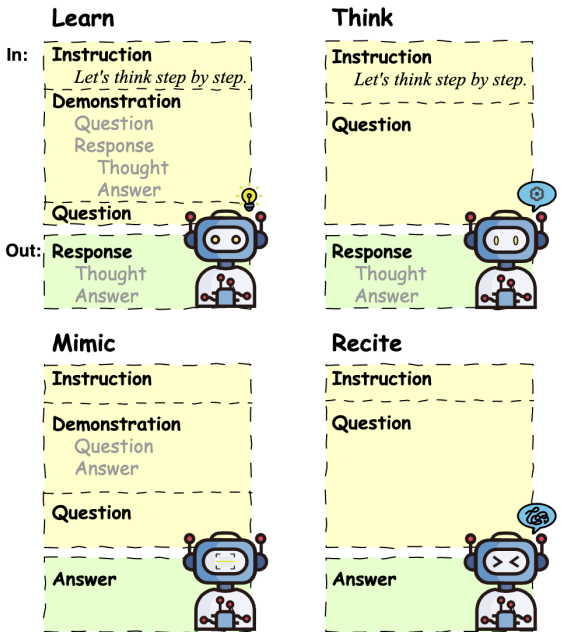

Figure 2: 大语言模型的四种推理模式。(学习、思考、模仿、背诵)

我们定义了以下推理模式：

- 学习：通过思考任务来获取新技能，并模仿范例给出的解决方案。
- 思考：进行推理或解决问题的过程，以得出结论。
- 模仿：模仿或复制给定示例的风格、语调或内容。
- 背诵：逐字逐句地从来源或记忆中复述信息。

通过系统地将任务分解为这些原子组件并定义清晰的推理模式，我们旨在提升 LLM 在各种应用中的性能和可靠性。

3.2 推理树中的节点组成

如图 3 所示，我们在节点内设计了一个分层的信息存储结构。任务相关的信息，例如提示库和推理路径，被存储在节点状态中。当提供新任务时，我们只需修改状态中的提示并观察模拟的响应，而无需关注状态以外的树搜索过程。

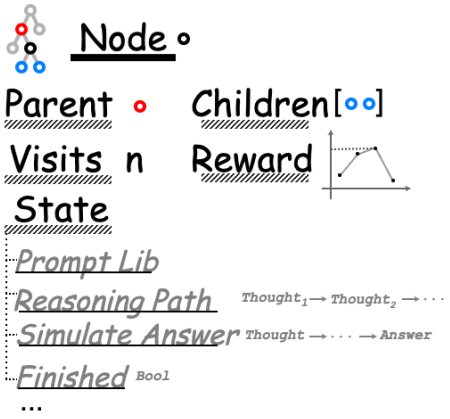

Figure 3: 节点组成。结构信息直接存储在节点中。推理信息存储在其状态中。

3.3 无监督语义感知聚类

在本节中，我们介绍了我们的答案聚类方法。随着推理树的扩展，每个节点上的模拟答案数量增加。为了有效分析大型语言模型（LLM）对特定问题的多样化回答，聚类语义相似的回答是至关重要的。

相似矩阵计算：通过从搜索树中的每个节点收集的模拟答案，我们使用 TF-IDF（词频–逆文档频率，Ramos et al., 2003 ）算法来获得每个答案的表示向量。然后，计算这些表示向量之间的余弦相似度以形成相似性矩阵。

聚类数目确定：为了更好地聚合不同的答案，我们接下来使用相似性矩阵来确定聚类的数量。相似性矩阵 G 作为一个图的度矩阵，可以对应于一个无向图。图 G 的拉普拉斯矩阵 L 通过方程 $L = D - A$ 计算得到，其中 L 是图 G 的拉普拉斯矩阵，D 是相似性矩阵（即，图 G 的度矩阵），A 是图 G 的邻接矩阵。接下来，我们计算矩阵 L 的特征值并按升序排列。识别出连续特征值之间的最大差异，并将对应的特征值跃迁作为聚类的数量。这个跃迁反映了图结构从几个连接的组件到更复杂结构的显著过渡。聚类的数量设置为这个特征值跃迁数，如果聚类的数量过大，则限制在固定的上限 5。

推理节点谱聚类：最后，谱聚类 (Von Luxburg, 2007) 被应用于对所有推理节点进行分类。然后我们获得多个节点集，从中选择每个簇中得分最高的答案作为该簇的代表答案。在评分阶

段，LLM 可以将新的模拟答案与这些代表答案进行比较。

3.4 寻找更好的推理程序

传统的蒙特卡罗树搜索（MCTS）方法倾向于在当前最佳路径的每个节点上详尽地探索所有可能的动作。这种方法对于大型模型的推理过程来说效率较低。此外，将 MCTS 应用于 LLM 推理的现有方法需要微调一个额外的奖励模型。与微调基础模型类似，这一过程缺乏通用性。在本节中，我们提出了一种优化的树搜索策略，旨在提高 LLM 的探索效率，并消除额外训练的需求。该流程如图 4 所示。经过初步评估，我们采用二叉树作为搜索树。随着度的增加，节点数量呈指数增长，在这种设置下提供了足够大的推理空间。

Algorithm 2: 选择 & 扩展

```
Function: tree_policy( v )
Input: original node v
Output: selected node v
v_initial ← v ;
while not v .terminal() do
    if not v .children then
        Return: expand( v )
    else if random(0,1) < 0.5 then
        v ← best_child( v, C_p );
    else
        if not v .fully_expanded() then
            v ← expand( v );
            if not v .visits then
                Return: v
            else
                Return: tree_policy(
                  v_initial )
            end
        else
            best_child( v, C_p );
        end
end
Return: v
```

选择 & 扩展

算法 2 中的优化 tree_policy 函数优先考虑扩展树的深度以探索后续步骤。当当前节点不是叶节点时，它有 50 % 的几率直接前往最佳子节点（通过方程 2 和 3 计算）以进行下一搜索步骤，而不是尝试在当前节点的其他未尝试的动作。

此外，我们对 best_child 函数中用于计算树的上置信界（UCT，如公式 1 所示）得分以选择最佳子节点的公式进行了改进。如前所述，搜索树是一个二叉树，这意味着在 LLM 树搜索中的动作空间是二元的。设当前节点为 v，需要评估的子节点为 v'。

原始的 UCT 公式（公式 1）由两个部分组成：利用和探索。考虑到大型语言模型（LLM）通常生成没有黄金标准参考的答案，客观评估变得具有挑战性，因此我们旨在通过增加评估次数来增强评估当前答案的信心。评估次数越多，整体分数分布就越可靠。否则，节点的分数应倾向于搜索树的平均分数。

$$\mathrm{S}_{\mathrm{LLM\_Exploit}}(v') = \frac{\mu_{\mathrm{Tree}} \cdot C_\beta + Q(v')}{C_\beta + N(v')} \tag{2}$$

$$\mathrm{S}_{\mathrm{LLM\_Explore}}(v') = C_p \sqrt{\frac{2 \ln N(v')}{N(v')}} \tag{3}$$

因此，我们采用贝叶斯平均法，修改公式的利用部分为公式 2，其中 μ_{Tree} 表示树中所有模拟答案的平均分数，而 C_β 是预期的评估次数（表示置信度，这里设置为 2）。

为了与前面提到的优先深入探索的策略保持一致，我们还优化了 UCT 公式中探索组件，如方程式 3 所示。原始的探索项是 $C_p\sqrt{\frac{2\ln N(v)}{N(v')}}$。设二叉树的左、右子节点的访问次数分别为 a 和 b，其中 $a \le b$ 和 $b = k \cdot a$，且有 $k \ge 1$。使用原始的 UCT 公式，两个节点的探索分数为：$\sqrt{\frac{\ln(a+b)}{a}} \ge \sqrt{\frac{\ln(a+b)}{b}}$。优化后的探索项是 $C_p\sqrt{\frac{2\ln N(v')}{N(v')}}$，，导致探索分数为：$\sqrt{\frac{\ln a}{a}} \ge \sqrt{\frac{\ln b}{b}}$。这种优化保持了原始的单调性，并导致了更渐进的探索分数比率，鼓励优先选择最佳子节点。

这种精炼的策略有效地平衡了探索与开发，减少了冗余的推理路径，并缓解了基于大型语言模型的树搜索算法中的幻觉现象。

Algorithm 3: 模拟

```
Function: default_policy( s )
Input: input state s
Output: reward for state r(s)
while not s .terminal() do
    choose a ∈ s .all_actions() uniformly at
      random;
    s ← next_state (s, a) ;
end
Return: r(s)
```

模拟

在这一步骤中，大型语言模型按照算法 3 中建立的推理路径完成响应。在计算最终答案的

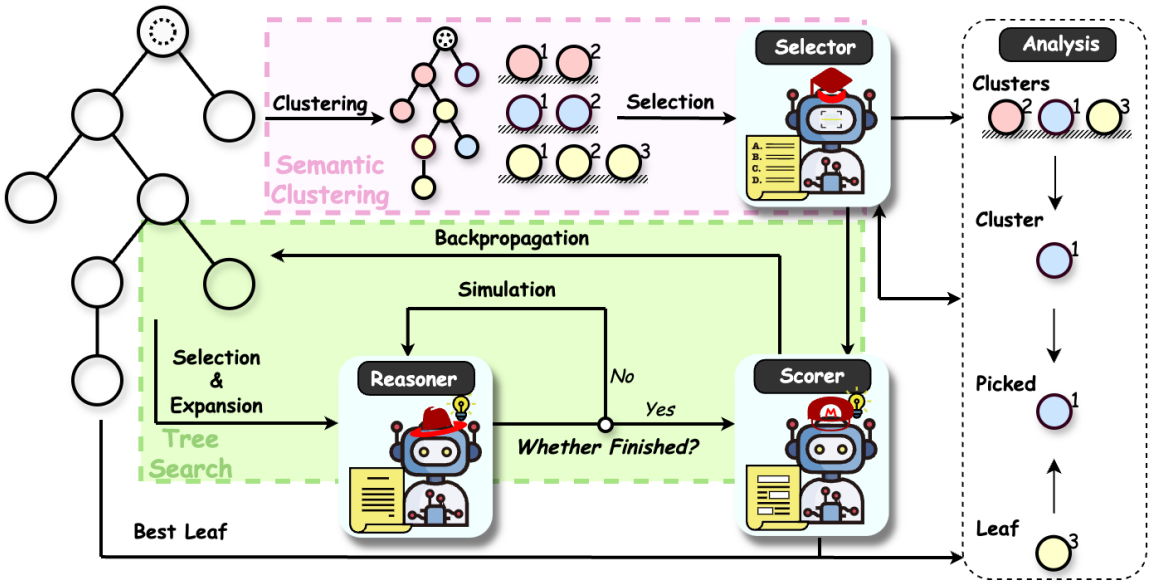

Figure 4: SELT 的完整程序。

Algorithm 4: 反向传播

Function: backup(v , Δ)
Input: input node v , reward score Δ
while v *is no null* **do**
  v .visits $\leftarrow$ v .visits $+1$;
  v .reward $\leftarrow$ v .reward $+\Delta$;
  v .reward_list.append(Δ);
  $v \leftarrow v$ .parent;
end

奖励之前，我们对推理树中的所有节点基于它们的模拟答案进行语义聚类，详见第 3.3 节。通过使用每个聚类的代表性响应作为参考标准，大型语言模型可以更准确地评估新的模拟答案。

反向传播

在获得当前节点的奖励分数后，我们进行反向传播以更新推理路径上至根节点的奖励，如同算法 4 中所描述的。每个子节点的奖励被记录在节点信息中。奖励列表可以用于进一步的评分稳定性分析。

分析

这一步是关于如何在搜索循环之后从推理树中获得最佳答案。我们尝试通过两种方法获取答案，并在其中找到更好的方法。

叶子: 通过 UCT（将 C_p 设为 0，如算法 1 中所示），来自最佳叶子节点的答案标记为“叶子”。

聚类: 如第 3.3 节所述，我们对树中的所有节点进行聚类，并选择每个聚类的代表节点。最终，LLM 在这些代表答案中选择的最佳答案标记为“聚类”。

选择: 在“叶子”和“聚类”之间选择 1 个，以获得选择的答案。

4 实验

4.1 实验设置

为了更好地展示实验结果，本节概述了实验设置。我们以 1-shot、1-shot CoT 和标准 MCTS 作为基线。结果表中将标准 MCTS 表示为 SELT 与 Leaf 和 S_{raw} 设置。相关的基准、模型和其他参数如下所示。

对于基础模型，我们使用 Llama-3.1-8B-Instruct [1] (Dubey et al., 2024)，这是较小的 LLM 中相对较强的模型。为了加速推理，我们部署了 vLLM 框架 [2] 。虽然不需要微调，但树搜索过程耗时较长，因为它基于搜索步骤 T（在我们的实验中设置为 100）。经过验证，100 个搜索步骤足以探索大多数问题的动作空间。

对于基准测试，我们使用基于知识的问答数据集 MMLU (Hendrycks et al., 2020) 和工具学习数据集 Seal-Tools (Wu et al., 2024) 。由于速度限制，我们仅使用了其中的一部分。对于 MMLU，我们选择了抽象代数、大学物理、大学化学的分支作为数学、物理和化学领域。

在使用 LLM 进行逐步推理时，确定每一步的细节层次是至关重要的。在句子级推理中，LLM 在每一步生成下一句思考。在回应级推理中 (Zhang et al., 2024a,b)，LLM 会对前一步生成的整个回应进行改进。我们在这两个细节层次上进行了实验。

[1] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct
[2] https://github.com/vllm-project/vllm

Method	Mathematics				Physics				Chemistry			
1-shot	32.00				40.20				52.00			
1-shot CoT	49.00				71.57				61.00			
SELT	S_{raw}	S_{α}	S_{β}	$S_{\alpha+\beta}$	S_{raw}	S_{α}	S_{β}	$S_{\alpha+\beta}$	S_{raw}	S_{α}	S_{β}	$S_{\alpha+\beta}$
Sentence-Level												
Leaf (Raw)	39.00	53.00	50.00	49.00	71.57	78.43	67.65	77.45	56.00	54.00	52.00	63.00
Cluster	48.00	42.00	45.00	50.00	60.78	70.59	64.71	74.51	52.00	52.00	41.00	59.00
Picked	51.00	51.00	49.00	50.00	67.65	77.45	67.65	80.39	55.00	58.00	45.00	64.00
Both	62.00	57.00	60.00	53.00	81.37	81.37	80.39	84.31	66.00	61.00	59.00	74.00
Clusters	85.00	84.00	85.00	76.00	100.00	98.04	98.04	97.06	95.00	92.00	95.00	90.00
Response-Level												
Leaf (Raw)	47.00	60.00	46.00	44.00	72.55	78.43	76.47	75.49	62.00	62.00	59.00	58.00
Cluster	38.00	56.00	41.00	50.00	59.80	73.53	65.69	66.67	48.00	63.00	50.00	54.00
Picked	53.00	63.00	49.00	46.00	72.55	77.45	78.43	76.47	60.00	62.00	59.00	57.00
Both	56.00	63.00	53.00	54.00	74.51	79.41	81.37	80.39	67.00	65.00	67.00	64.00
Clusters	88.00	78.00	86.00	75.00	98.04	93.14	100.00	90.20	90.00	82.00	94.00	81.00

Table 2: 基准 MMLU 上的实验结果。带有 S_{raw} 的 SELT-Leaf 可以视为原始的 MCTS。

Method	Single							Multiple						
	Format	Tool			Param			Format	Tool			Param		
		P	R	F1	P	R	F1		P	R	F1	P	R	F1
1-shot	100.00	76.03	92.00	83.26	55.87	82.14	66.51	99.00	92.33	90.60	91.46	74.81	86.58	80.27
1-shot CoT	99.00	63.33	95.00	76.00	48.87	77.38	59.91	100.00	94.36	94.36	94.36	82.26	87.12	84.62
MCTS	95.00	75.63	90.00	82.19	59.42	73.21	65.60	86.00	94.89	81.50	87.69	82.16	73.35	77.50
SELT														
S_{α}														
Leaf (Raw)	95.00	76.23	93.00	83.78	57.48	73.21	64.40	87.00	96.28	81.19	88.10	82.23	71.20	76.32
Cluster	99.00	78.05	96.00	86.10	58.18	76.19	65.98	99.00	95.15	92.16	93.63	82.78	85.15	83.95
Picked	99.00	80.67	96.00	87.67	57.87	74.40	65.10	100.00	95.51	93.42	94.45	83.13	86.40	84.74
$S_{\alpha+\beta}$														
Leaf (Raw)	96.00	79.13	91.00	84.65	64.50	76.79	70.11	86.00	94.57	81.82	87.73	81.98	74.06	77.82
Cluster	95.00	75.42	89.00	81.65	59.49	69.05	63.91	98.00	95.22	93.73	94.47	83.87	88.37	86.06
Picked	96.00	77.97	92.00	84.40	62.56	72.62	67.22	96.00	95.08	90.91	92.95	83.84	84.44	84.14

Table 3: 基准 Seal-Tools 的实验结果。

除了在第 3.4 节中提到的分析结果外，我们采用“Both”记录金答案是否在“Leaf”和“Cluster”之间。我们还使用“Clusters”判断金答案是否在每个簇的所有代表性答案中。$S_{\alpha+\beta}$ 是在第 2 节中提到的修改后的 UCT。S_{raw} 是原始版本的 UCT，S_{α} 是修改探索的 UCT（如方程 2 所示），而 S_{β} 是修改开发的 UCT（如方程 3 所示）。

4.2 主要结果

4.2.1 知识问答

MMLU 评估 LLMs 根据其内部知识回答领域特定问题的能力。如表 2 所示，通过我们新修改的 UCT 和答案聚类策略，句子级 SELT 超过了原始 MCTS（叶子带有 S_{raw} 设置）(Browne et al., 2012) 。它也优于响应级 SELT，表明句子级树搜索对于 LLMs 更为有效，因为它提供了足够的推理空间以供探索。另一方面，响应级 SELT 倾向于优先考虑利用，而修改后的探索策略没有很好的表现。这可能表明它需要更积极地探索推理树的更深层。

4.2.2 工具学习

Seal-Tools 评估大型语言模型在推理过程中适当地调用外部工具的能力，这对于自治代理的基础模型至关重要。如表 3 所示，SELT 在单一工具调用分项中取得了更大的改善，而多工具调用的结果则受输出格式的影响。一个有趣的观察是，在单一调用分项中，1-shot CoT 的基线表现比 1-shot 更差。这可能是因为单一调用数据不需要逐步推理。

4.3 实验分析

实验表明，SELT 在句子和回应层面上均优于原始的 MCTS。在多种设置下，对 UCT 开发和语义聚类的改进都是有益的。与回应层面相比，句子层面的树搜索可以更彻底地探索动作空间并具有更高的上限。然而，LLM 遵循格式的能力仍需注意，因为这可能影响长链推理的有效性。

5 相关工作

研究人员尝试通过优化训练过程来提升大语言模型（LLM）的能力。在这个阶段，增大模型规模和使用更多的数据被证明是有效的。模型推理的测试时间缩放定律 (Snell et al., 2024) 同样表现良好。许多研究已经确认，探索 LLM 的推理方法可以显著提高其在众多下游任务中的表现。相比于微调方法，它所需的计算资源更少，并且可以随时使用。

自从生成性语言模型 ChatGPT 进入公众视野以来，许多基于提示的方法已被发现可以指导 LLMs 输出得更好。上下文学习（ICL, Brown et al., 2020 ）在提示中添加类似的示例作为演示；然后模型可以模仿它们的思维方式以获得更好的答案。思维链（CoT, Wei et al., 2022 ）要求模型逐步思考问题，并更加关注问题的细节。思维树（ToT, Yao et al., 2024 ）使用 DFS 和 BFS 的搜索策略来在推理树中获得更好的思维路径。ReAct (Yao et al., 2023) 在生成过程中重复推理和观察反思。虽然这些方法在鼓励 LLM 更多思考方面取得了一定成功，但它在解决复杂问题时可能仍然会遇到困难，特别是当这些问题需要深入推理或探索广泛可能性解决方案的能力时。

为了进一步激发大型语言模型 (LLM) 的潜力，在推理过程中正在研究蒙特卡洛树搜索（MCTS）。MCTS 算法在围棋游戏中取得了巨大成功，这证明了该方法的有效性。它用于寻找巨大的多步骤行动空间 (Browne et al., 2012) 中最佳可能路径。LLM-MCTS (Zhao et al., 2024) 将常识性先验信念与 MCTS 结合起来，实现有效推理。RAP (Hao et al., 2023) 将 LLM 重新考虑为世界模型，通过 MCTS 在广阔的推理空间中进行战略探索。有些工作使用 MCTS 构造模型微调的训练数据（(Tian et al., 2024; Zhang et al., 2025) ）。LATS (Zhou et al., 2024) 结合 MCTS 和 ReAct 一起观察并反思推理路径中可能存在的问题。

6 结论

在本文中，我们介绍了 SELT，这是一种新颖的 MCTS 框架，它利用大语言模型的内在自我评估能力来增强复杂推理，而无需依赖外部奖励模型。通过重新定义 UCT 评分机制并将推理过程分解为原子子任务与语义聚类，SELT 有效地平衡了探索和利用，同时减轻了冗余和幻觉问题。我们在 MMLU 和 Seal-Tools 基准上的广泛实验表明，SELT 不仅在回答准确性和鲁棒性方面显著优于链式思维提示和标准 MCTS 等传统方法，而且在各种推理任务中保持强大的泛化能力，无需特定任务的微调。这些令人鼓舞的结果为未来进一步整合更具适应性的自我评估技术并在更多推理领域中探索更广泛的应用铺平了道路。总的来说，SELT 代表了一种更高效、更可靠和更具可扩展性的大型语言模型推理过程的有希望的进展。

7

局限性

虽然 SELT 在增强 LLM 推理能力方面表现出了可喜的改进，但也应注意一些局限性。

自我评估：该框架依赖于 LLM 的内在自我评估能力来修改 UCT 评分，这可能并不总能对中间推理质量做出准确的评估。自我评估中的错误或偏见可能会在树搜索中传播，影响最终结果。

计算成本：尽管采取了提高搜索效率的措施，与传统的提示方法相比，树搜索引入的额外计算开销仍然较高，这可能会限制实时应用。

基准测试：由于计算资源有限，我们目前的实验仅限于特定的基准测试（例如，基于知识的问答和工具学习任务），需要进一步研究来评估该框架在更广泛和开放式推理挑战中的可扩展性和性能。

解决这些限制将在未来的研究中至关重要，以进一步优化 LLM 的推理性能，确保更广泛的适用性，并推动大型语言模型在复杂推理方面的进展。

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, et al. 2023. Language modeling is compression. *arXiv preprint arXiv:2309.10668*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela

Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Sylvain Gelly and Yizao Wang. 2006. Exploration exploitation in go: Uct for monte-carlo go. In *NIPS: Neural Information Processing Systems Conference On-line trading of Exploration and Exploitation Workshop*.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Ariana Martino, Michael Iannelli, and Coleen Truong. 2023. Knowledge injection to counter large language model (llm) hallucination. In *European Semantic Web Conference*, pages 182–185. Springer.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13679–13707.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Haitao Mi, and Dong Yu. 2024. Toward self-improvement of llms via imagination, searching, and criticizing. *arXiv preprint arXiv:2404.12253*.

Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Mengsong Wu, Tong Zhu, Han Han, Chuanyuan Tan, Xiang Zhang, and Wenliang Chen. 2024. Seal-tools: Self-instruct tool learning dataset for agent tuning and detailed benchmark. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 372–384. Springer.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2025. Rest-mcts*: Llm self-training via process reward guided tree search. *Advances in Neural Information Processing Systems*, 37:64735–64772.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. 2024a. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b. *arXiv preprint arXiv:2406.07394*.

Di Zhang, Jianbo Wu, Jingdi Lei, Tong Che, Jiatong Li, Tong Xie, Xiaoshui Huang, Shufei Zhang, Marco Pavone, Yuqiang Li, et al. 2024b. Llama-berry: Pairwise optimization for o1-like olympiad-level mathematical reasoning. *arXiv preprint arXiv:2410.02884*.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2024. Large language models as commonsense knowledge for large-scale task planning. *Advances in Neural Information Processing Systems*, 36.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 62138–62160. PMLR.

A 更多实验结果

Method	Single							Multiple						
	Format	Tool			Param			Format	Tool			Param		
		P	R	F1	P	R	F1		P	R	F1	P	R	F1
1-shot	100.00	76.03	92.00	83.26	55.87	82.14	66.51	99.00	92.33	90.60	91.46	74.81	86.58	80.27
1-shot CoT	99.00	63.33	95.00	76.00	48.87	77.38	59.91	100.00	94.36	94.36	94.36	82.26	87.12	84.62
Sent SELT														
S_{raw}														
Picked	95.00	80.36	90.00	84.91	63.35	72.02	67.41	93.00	94.54	86.83	90.52	82.27	80.50	81.37
Leaf	95.00	75.63	90.00	82.19	59.42	73.21	65.60	86.00	94.89	81.50	87.69	82.16	73.35	77.50
Cluster	92.00	83.02	88.00	85.44	62.03	69.05	65.35	98.00	94.52	91.85	93.16	82.38	85.33	83.83
S_{α}														
Picked	99.00	80.67	96.00	87.67	57.87	74.40	65.10	100.00	95.51	93.42	94.45	83.13	86.40	84.74
Leaf	95.00	76.23	93.00	83.78	57.48	73.21	64.40	87.00	96.28	81.19	88.10	82.23	71.20	76.32
cluster	99.00	78.05	96.00	86.10	58.18	76.19	65.98	99.00	95.15	92.16	93.63	82.78	85.15	83.95
S_{β}														
Picked	91.00	80.91	89.00	84.76	62.31	73.81	67.57	92.00	94.52	86.52	90.34	80.66	79.07	79.86
Leaf	92.00	78.26	90.00	83.72	59.33	73.81	65.78	79.00	92.34	71.79	80.78	80.73	63.69	71.20
Cluster	94.00	83.78	93.00	88.15	65.10	74.40	69.44	98.00	94.25	92.48	93.35	80.75	84.79	82.72
$S_{\alpha+\beta}$														
Picked	96.00	77.97	92.00	84.40	62.56	72.62	67.22	96.00	95.08	90.91	92.95	83.84	84.44	84.14
Leaf	96.00	79.13	91.00	84.65	64.50	76.79	70.11	86.00	94.57	81.82	87.73	81.98	74.06	77.82
Cluster	95.00	75.42	89.00	81.65	59.49	69.05	63.91	98.00	95.22	93.73	94.47	83.87	88.37	86.06
Resp SELT														
S_{raw}														
Picked	98.00	69.34	95.00	80.17	56.22	77.98	65.34	98.00	92.21	89.03	90.59	82.17	82.47	82.32
Leaf	98.00	69.06	96.00	80.33	53.97	76.79	63.39	94.00	91.95	85.89	88.82	80.93	77.46	79.16
Cluster	97.00	68.89	93.00	79.15	58.48	77.98	66.84	99.00	93.97	92.79	93.38	85.14	87.12	86.12
S_{α}														
Picked	99.00	62.18	97.00	75.78	50.57	79.17	61.72	97.00	92.95	90.91	91.92	82.86	83.01	82.93
Leaf	99.00	61.69	95.00	74.80	50.39	77.38	61.03	95.00	93.14	89.34	91.20	82.57	80.50	81.52
Cluster	100.00	68.06	98.00	80.33	53.60	79.76	64.11	99.00	90.55	93.10	91.81	80.94	85.87	83.33
S_{β}														
Picked	96.00	74.19	92.00	82.14	60.95	76.19	67.72	99.00	93.71	93.42	93.56	81.23	87.48	84.24
Leaf	96.00	71.54	93.00	80.87	56.19	75.60	64.47	87.00	92.83	81.19	86.62	81.00	72.45	76.49
Cluster	97.00	76.42	94.00	84.30	64.90	80.36	71.81	100.00	90.68	91.54	91.11	79.83	85.69	82.66
$S_{\alpha+\beta}$														
Picked	98.00	64.79	92.00	76.03	52.07	75.00	61.46	97.00	94.48	91.22	92.82	81.74	84.08	82.89
Leaf	98.00	65.96	93.00	77.18	53.25	77.98	63.29	97.00	93.85	90.91	92.36	80.83	83.72	82.25
Cluster	98.00	65.28	94.00	77.05	54.07	79.17	64.25	100.00	94.06	94.36	94.21	82.03	88.19	85.00

Table 4: 基准 Seal-Tools 的完整实验结果。