
RETreVAL: REASONING TREE WITH VALIDATION - A HYBRID FRAMEWORK FOR ENHANCED LLM MULTI-STEP REASONING

Abhishek HS
QpiAI, Bengaluru, India
abhishek.hs@qpi.ai.tech

Pavan C Shekar
QpiAI, Bengaluru, India
pavan.s@qpi.ai.tech

Arpit Jain
QpiAI, Bengaluru, India
arpit.j@qpi.ai.tech

Ashwanth Krishnan
QpiAI, Bengaluru, India
ashwanth.krishnan@qpi.ai.tech

January 7, 2026

ABSTRACT

Multi-step reasoning remains a key challenge for Large Language Models (LLMs), particularly in complex domains such as mathematics and creative writing. While recent approaches including ReAct, Reflexion, and Self-Refine improve reasoning through iterative refinement and reflection, they often lack structured exploration of alternative solution paths and persistent learning across problems. We propose ReTreVal (Reasoning Tree with Validation), a hybrid framework that integrates Tree-of-Thoughts exploration, self-refinement, LLM-based critique scoring, and reflexion memory to enable bounded and validated multi-step reasoning. ReTreVal constructs a structured reasoning tree with adaptive depth based on problem complexity, where each node undergoes iterative self-critique and refinement guided by explicit LLM-generated feedback. A dual validation mechanism evaluates reasoning quality, coherence, and correctness at each node while persistently storing insights from successful reasoning paths and failure patterns in a reflexion memory buffer, enabling cross-problem learning. Critique-based pruning retains only the top-k highest-scoring nodes at each level, controlling computational cost while preserving high-quality solution paths. We evaluate ReTreVal against ReAct, Reflexion, and Self-Refine across 500 mathematical problems and creative writing tasks using Qwen 2.5 7B as the underlying LLM, and demonstrate that ReTreVal consistently outperforms existing methods through its combination of structured exploration, critique-driven refinement, and cross-problem memory, making it particularly effective for tasks requiring exploratory reasoning, rigorous verification, and knowledge transfer.

Keywords Large Language Models · Multi-Step Reasoning · Tree-of-Thoughts · Self-Refinement · Critique-Based Evaluation · Reflexion Memory · ReAct · Agent Frameworks

1 Introduction

The rapid advancement of Large Language Models (LLMs) has revolutionized natural language processing, demonstrating remarkable capabilities in text generation, translation, and question answering. However, complex reasoning tasks—particularly those requiring multiple steps of logical deduction, mathematical computation, or creative synthesis—remain a significant challenge. While LLMs excel at pattern matching and knowledge retrieval, they often struggle with systematic problem decomposition, error correction, and maintaining coherent reasoning chains across extended solution paths [1, 2].

Recent research has introduced several promising frameworks to enhance LLM reasoning capabilities. ReAct [3] interleaves reasoning and action steps, allowing models to iteratively think through problems while taking concrete actions. Reflexion [4] introduces a learning mechanism where models reflect on past failures and adjust their approach

accordingly. Self-Refine [5] enables iterative solution improvement through self-generated feedback. While these approaches have shown improvements over baseline prompting methods, they face inherent limitations: ReAct’s linear reasoning chain may miss alternative solution paths, Reflexion’s trial-and-error approach can be computationally expensive, and Self-Refine lacks structured exploration of the solution space.

Concurrently, Tree-of-Thoughts (ToT) [6] emerged as a powerful paradigm for structured reasoning, enabling models to explore multiple reasoning paths simultaneously through a tree-based search structure. ToT excels at problems requiring lookahead and backtracking, but lacks mechanisms for continuous improvement of individual reasoning steps and cross-problem knowledge retention. The key insight is that neither purely iterative refinement nor purely exploratory search alone is sufficient—complex reasoning requires both structured exploration and systematic validation with persistent learning.

We introduce ReTreVal (Reasoning Tree with Validation), a novel hybrid framework that synergistically combines the structured exploration of Tree-of-Thoughts with the iterative refinement capabilities of Self-Refine, augmented by two critical mechanisms: LLM-based critique scoring for explicit quality evaluation and reflexion memory for cross-problem learning. ReTreVal addresses the limitations of existing approaches through three key innovations:

Adaptive Tree Construction: Unlike fixed-depth trees, ReTreVal dynamically adjusts tree depth based on problem complexity, balancing exploration breadth with computational efficiency.

Critique-Driven Validation: Each reasoning node is evaluated by an LLM-based critic that provides explicit scores, identifies weaknesses, and suggests improvements, enabling targeted refinement of reasoning quality.

Persistent Memory System: A reflexion buffer captures insights from successful solutions and patterns from failures, enabling the system to learn across problems rather than treating each problem in isolation.

The framework operates by constructing a reasoning tree where each node represents a potential reasoning step. At each node, the system generates multiple candidate thoughts, applies self-critique to refine them, evaluates their quality through LLM scoring, and prunes low-quality paths. The reflexion memory continuously accumulates knowledge, allowing later problems to benefit from earlier reasoning experiences. This combination of structured search, iterative refinement, explicit validation, and memory-driven learning creates a robust reasoning system capable of handling both mathematical problem-solving and creative tasks.

We conduct comprehensive experiments comparing ReTreVal against ReAct, Reflexion, and Self-Refine across two distinct domains: mathematical problem-solving (500 problems from standardized datasets) and creative writing tasks. Our evaluation demonstrates that ReTreVal achieves superior performance in solution accuracy, reasoning coherence, and computational efficiency. The results validate our hypothesis that combining structured exploration with validation and memory yields more reliable and robust reasoning capabilities than any single approach alone.

1.1 Contributions

The main contributions of this paper are:

- A novel hybrid framework that integrates Tree-of-Thoughts exploration with self-refinement, critique-based validation, and reflexion memory for enhanced multi-step reasoning.
- Adaptive tree construction algorithm that dynamically adjusts reasoning depth based on problem complexity, optimizing the trade-off between exploration and computation.
- Dual validation mechanism combining LLM-based critique scoring for quality assessment with reflexion memory for cross-problem knowledge retention.
- Comprehensive empirical evaluation comparing ReTreVal against three state-of-the-art reasoning frameworks across mathematical and creative domains, demonstrating consistent improvements in solution quality and reasoning reliability.

2 Related Work

2.1 Iterative Reasoning Frameworks

Recent work has introduced iterative approaches to enhance LLM reasoning through feedback loops and self-improvement mechanisms.

ReAct [3] combines reasoning traces with actions, interleaving “thought-action-observation” cycles. The model generates reasoning steps, executes actions (e.g., calculations, searches), and observes outcomes before proceeding.

While effective for grounded tasks, ReAct’s linear structure limits exploration of alternative reasoning paths and lacks memory of past problem-solving experiences.

Reflexion [4] introduces self-reflection to learn from failures. After generating a solution, the model evaluates its correctness, reflects on errors, and stores insights in short-term memory. Subsequent attempts leverage these reflections to avoid repeated mistakes. However, Reflexion’s trial-and-error approach can be computationally expensive and lacks structured exploration of solution spaces.

Self-Refine [5] iteratively improves solutions through self-generated feedback. The model produces an initial solution, critiques it, and refines based on feedback. This process repeats until satisfactory quality is achieved. While Self-Refine demonstrates strong refinement capabilities, it operates on single solution paths without exploring alternatives and maintains no persistent memory across problems.

2.2 Tree-Based Search Methods

Tree-of-Thoughts (ToT) [6] enables deliberate exploration through tree-based reasoning. The model generates multiple candidate thoughts at each step, evaluates their promise, and selectively expands the most promising paths. ToT excels at problems requiring lookahead and backtracking, such as game playing and creative writing. However, standard ToT implementations lack mechanisms for iterative refinement of individual nodes and do not retain knowledge across different problems.

Beam search and best-first search strategies have been adapted for LLM reasoning, maintaining multiple hypotheses simultaneously. These methods improve solution diversity but typically rely on simple heuristics for evaluation rather than explicit critique and lack integration with self-refinement mechanisms.

2.3 Memory and Learning Systems

Episodic memory in agent systems enables retention of past experiences for future problem-solving. Systems like Voyager [7] and Generative Agents [8] maintain memory streams capturing observations, reflections, and learned strategies. These memories are retrieved based on relevance to guide current actions. While powerful for sequential tasks, most memory systems focus on action-based agents rather than pure reasoning tasks.

Meta-learning approaches enable models to learn problem-solving strategies across tasks. However, these typically require extensive training or fine-tuning, whereas our reflexion memory operates at inference time through in-context learning.

2.4 Critique and Self-Evaluation

LLM-as-judge paradigms [9] leverage language models to evaluate outputs from other models or themselves. Constitutional AI [10] uses critique models to identify and correct harmful outputs. Self-consistency methods [11] generate multiple solutions and select the most common answer. Recent work on self-verification trains models to critique their own reasoning steps [12].

Process supervision provides feedback on intermediate reasoning steps rather than only final answers, improving step-by-step reasoning quality. However, most approaches require human annotations or trained reward models, whereas ReTreVal uses the LLM itself for critique generation.

2.5 Hybrid and Multi-Strategy Approaches

Recent work explores combining multiple reasoning strategies. RAP (Reasoning via Planning) [13] integrates Monte Carlo Tree Search with LLM reasoning. LATS (Language Agent Tree Search) [14] combines tree search with self-reflection. However, these approaches either lack persistent memory mechanisms or do not integrate continuous node-level refinement.

2.6 Positioning ReTreVal

ReTreVal distinguishes itself through the synergistic integration of four components: (1) adaptive tree-based exploration from ToT, (2) node-level self-refinement from Self-Refine, (3) explicit LLM-based critique scoring for quality assessment, and (4) persistent reflexion memory for cross-problem learning. Unlike ReAct’s linear chains, Reflexion’s trial-and-error, or Self-Refine’s single-path optimization, ReTreVal explores multiple reasoning paths while continuously refining each node. Unlike standard ToT, it augments tree search with iterative improvement and memory retention.

This combination enables both breadth of exploration and depth of refinement, with accumulated knowledge improving performance across problem sequences.

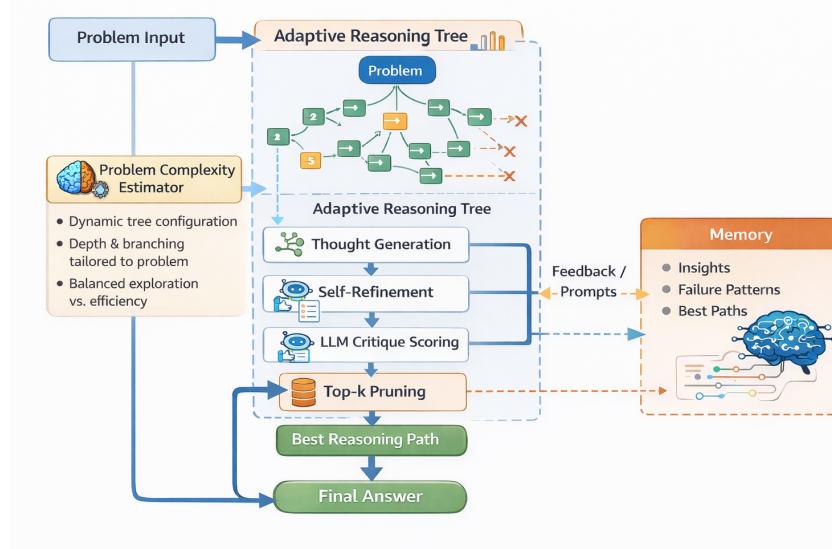


Figure 1: The framework combines adaptive tree-based reasoning, node-level self-refinement, and LLM-based critique scoring with a persistent memory module that enables cross-problem learning and iterative improvement.

3 ReTreVal Architecture and Methodology

3.1 System Overview

ReTreVal integrates four core components into a unified reasoning framework: (1) adaptive tree construction for exploring multiple reasoning paths, (2) node-level self-refinement for iterative improvement, (3) LLM-based critique scoring for quality validation, and (4) reflexion memory for cross-problem knowledge retention.

3.2 Adaptive Tree Construction

Unlike fixed-depth tree search, ReTreVal dynamically adjusts tree parameters based on problem complexity. The system first analyzes the problem using an LLM to estimate complexity on a 1-5 scale, considering factors such as problem length, domain difficulty, and required reasoning steps.

Problem Complexity	Max Tree Depth	Children per Node
1–2 (Simple)	2	2
3 (Moderate)	3	3
4–5 (Complex)	4–5	4

Table 1: Adaptive tree configuration based on estimated problem complexity. ReTreVal dynamically adjusts tree depth and branching factor to balance exploration breadth and computational efficiency.

The parameters in Table 1 define how the search tree expands based on problem complexity. For simple problems (complexity 1-2), a shallow tree with depth 2 and binary branching (2 children per node) is sufficient, yielding up to 6 total nodes. Moderate problems (complexity 3) use a depth of 3 with ternary branching (3 children per node), allowing for up to 39 nodes. Complex problems (complexity 4-5) require deeper exploration with depth 4-5 and quaternary branching (4 children per node), potentially generating hundreds of nodes to thoroughly explore the solution space.

3.2.1 Node Structure

Each node n_i contains:

- **thought:** The reasoning content at this step
- **depth:** Position in tree (0 = root)
- **refinement_count:** Number of self-refinement iterations
- **critique_history:** List of self-generated critiques
- **local_score:** Self-evaluation score $\in [0, 1]$
- **cross_score:** LLM critic evaluation $\in [0, 1]$
- **combined_score:** $0.6 \times \text{local_score} + 0.4 \times \text{cross_score}$
- **children:** List of child node IDs
- **parent_id:** Reference to parent node

3.2.2 Expansion Algorithm

At each iteration, the system selects leaf nodes with $\text{depth} < \text{max_depth}$ and generates 2-4 child thoughts per node. Generation is guided by:

1. Parent node's reasoning context
2. Insights from reflexion memory
3. Problem-specific constraints
4. Patterns to avoid (from memory failures)

3.3 Self-Refinement Process

Each newly generated node undergoes iterative self-refinement before scoring. The refinement process is described in Algorithm 1.

Algorithm 1 Self-Refinement Process

```

1: Input: node  $n$ , max_refinements  $r$ 
2: Output: refined node  $n'$ 
3: for iteration  $i = 1$  to  $r$  do
4:   critique  $\leftarrow$  LLM.critique( $n.\text{thought}$ , context)
5:   if critique.quality_score  $\geq$  threshold then
6:     break
7:   end if
8:   feedback  $\leftarrow$  extract_actionable_feedback(critique)
9:    $n.\text{thought} \leftarrow$  LLM.refine( $n.\text{thought}$ , feedback)
10:   $n.\text{critique\_history.append}($ critique $)$ 
11:   $n.\text{refinement\_count} += 1$ 
12: end for
13: return  $n$ 

```

The critique prompt asks the LLM to evaluate:

1. **Logical coherence:** Does the reasoning follow logically?
2. **Correctness:** Are there factual or computational errors?
3. **Completeness:** Are important aspects missing?
4. **Clarity:** Is the reasoning clearly expressed?

Refinement continues until either (1) quality threshold is met, (2) max refinements reached, or (3) no improvement detected between iterations.

3.4 LLM-Based Critique Mechanism

After refinement, each node receives an explicit quality score from an LLM critic. The scoring system evaluates multiple dimensions:

Scoring Criteria:

1. **Logical Validity:** Soundness of reasoning steps
2. **Progress Toward Solution:** How much closer to answer
3. **Coherence:** Internal consistency
4. **Correctness:** Factual and computational accuracy
5. **Innovation:** Novel insights or approaches

The critic generates:

1. Numeric score $\in [0, 1]$ aggregating all criteria
2. Rationale: Textual explanation of strengths/weaknesses
3. Suggestions: Specific improvements for future nodes

Dual Scoring: Each node receives two scores:

1. **Local score:** Self-evaluation by the reasoning agent
2. **Cross score:** External evaluation by the critic LLM

The combined score balances self-assessment with external validation:

$$\text{score}_{\text{combined}} = 0.6 \times \text{score}_{\text{local}} + 0.4 \times \text{score}_{\text{cross}} \quad (1)$$

This weighting prevents both over-confidence (too much self-scoring) and over-reliance on external critique.

3.5 Reflexion Memory System

The memory buffer maintains persistent knowledge across problem-solving episodes, implementing a simple but effective learning mechanism.

Memory Structure:

1. **Insights Queue:** FIFO buffer (max 10) storing successful reasoning patterns. Example: “Breaking complex equations into sub-problems improves accuracy”
2. **Failures Queue:** FIFO buffer (max 10) storing patterns to avoid. Example: “Premature rounding in intermediate steps causes errors”
3. **Best Paths:** List of node IDs from highest-scoring solution chains
4. **Iteration Counter:** Tracks reasoning cycles

Memory Operations:

- **Storage:** After each reasoning episode:
 1. Extract key insights from high-scoring nodes
 2. Identify failure patterns from low-scoring nodes
 3. Update best paths from optimal solution
- **Retrieval:** During node generation and refinement:
 1. Inject relevant insights into generation prompt
 2. Include failure patterns as constraints
 3. Reference successful strategies from best paths
- **Pruning:** When buffers exceed capacity, oldest entries are removed (FIFO), ensuring memory focuses on recent, relevant patterns.

Cross-Problem Learning: Unlike per-problem approaches, reflexion memory accumulates knowledge across multiple problems. Insights from solving problem P_i inform the approach to problem P_{i+1} , enabling progressive improvement in reasoning strategies.

3.6 Pruning Strategy

To control computational complexity, ReTreVal prunes low-quality nodes at each tree level. After scoring all nodes at depth d , the system:

- Ranks all nodes by combined_score
- Selects top- k highest-scoring nodes (typically $k = 3$)
- Prunes remaining nodes and their potential subtrees
- Continues expansion only from top- k nodes

This ensures the tree explores promising reasoning paths while discarding unproductive directions early. The pruning is greedy but informed by both self-evaluation and critique scores.

Adaptive k: For higher complexity problems, k may be increased to maintain more diverse paths, while simpler problems use smaller k for efficiency.

3.7 Convergence Criteria

The reasoning loop terminates when any of the following conditions are met:

- **Maximum Depth Reached:** All active leaf nodes are at max_depth
- **Score Plateau:** No improvement in best score for 2 consecutive iterations
- **High Confidence:** Best node achieves score ≥ 0.95
- **Iteration Limit:** Maximum iterations exceeded (safety bound)

Upon convergence, the system traces back from the highest-scoring leaf node to the root, constructing the optimal reasoning chain.

3.8 Computational Complexity

For a problem with complexity c , maximum depth d_{\max} , and branching factor b :

- Nodes generated: $O(b^{d_{\max}})$ in worst case
- With pruning (top- k): $O(k \times d_{\max})$ nodes maintained
- Refinement cost: r iterations per node
- Total LLM calls: $O(k \times d_{\max} \times (r + 2))$ where +2 accounts for generation and scoring

4 Experimental Setup

4.1 Datasets

We evaluate ReTreVal on two distinct reasoning domains to assess generalization across problem types:

4.1.1 Mathematical Problem-Solving Dataset

We utilize 500 mathematical problems spanning multiple difficulty levels and topics from standard educational datasets. The problems cover:

- Arithmetic and algebraic equations
- Word problems requiring multi-step reasoning
- Geometry and measurement problems
- Probability and combinatorics
- Systems of equations

Problems were selected to represent diverse reasoning challenges, with complexity ranging from simple single-step calculations to multi-step problems requiring abstract reasoning and symbolic manipulation.

4.1.2 Creative Writing Dataset

For the creative writing task, we use a random sentence generator to create initial prompts. From these randomly generated sentences, the system must produce four coherent, creative sentences that form a complete micro-story or narrative. This task tests:

- Narrative coherence across multiple sentences
- Creative expansion of initial prompts
- Contextual consistency
- Linguistic quality and engagement

The random generation ensures unbiased, diverse prompts without dataset-specific patterns or biases. Each creative writing instance begins with a randomly generated sentence, and systems must generate four follow-up sentences that create a cohesive narrative.

4.2 Baseline Methods

We compare ReTreVal against three state-of-the-art reasoning frameworks, implementing each with equivalent configuration:

4.2.1 ReAct (Reason + Act)

- Implements thought-action-observation cycles
- Maximum 3 iterations per problem
- No memory retention across problems
- Linear reasoning chain structure

4.2.2 Reflexion

- Trial-and-error with self-reflection
- Maintains short-term memory within single problem
- Maximum 3 attempts per problem
- Reflects on failures and adjusts approach

4.2.3 Self-Refine

- Iterative solution refinement
- Maximum 3 refinement cycles per solution
- Self-generated feedback mechanism
- Single-path optimization

All baseline implementations use identical LLM configurations and prompting strategies to ensure fair comparison. Each method was carefully implemented following original paper specifications.

4.3 Implementation Details

4.3.1 Large Language Model

- **Primary Model:** Qwen 2.5 7B[15] (7 billion parameters)
- **Deployment:** Local inference via Ollama
- **Temperature:** 0.7 (balancing creativity and consistency)
- **Max Output Tokens:** 2048 per generation
- **Context Window:** 32K tokens

4.3.2 Evaluation Model

- **Judge LLM:** GPT-4o mini (OpenAI)
- **Purpose:** Automated scoring of creative writing outputs
- **Scoring Criteria:** Correctness, Meaningfulness, Creativeness (1-10 scale)

4.3.3 ReTreVal Hyperparameters

- Adaptive depth: 2-5 based on problem complexity
- Refinement iterations: Maximum 3 per node
- Pruning strategy: Top- $k = 3$ nodes per level
- Memory capacity: 10 insights, 10 failures (FIFO)
- Convergence threshold: Score ≥ 0.95 or 2 iterations without improvement

4.3.4 Computational Environment

- **Infrastructure:** AWS Cloud (EC2 instance with NVIDIA L40 GPU)
- **GPU:** NVIDIA L40 (48 GB VRAM)
- **Operating System:** Linux (AWS AMI / Ubuntu)
- **Model Server:** Ollama (serving Qwen 2.5 7B)

4.4 Evaluation Metrics

4.4.1 Mathematical Problem-Solving

- **Accuracy:** Percentage of correct final answers
- **Partial Credit:** Correct reasoning steps even if final answer wrong

4.4.2 Creative Writing (GPT-4o mini as Judge)

- **Correctness (1-10):** Grammatical accuracy, logical consistency
- **Meaningfulness (1-10):** Depth, coherence, narrative sense
- **Creativeness (1-10):** Originality, engagement, literary quality

5 Results and Analysis

5.1 Mathematical Problem-Solving Performance

Table 2 presents the performance comparison on 500 mathematical problems, evaluated by GPT-4o mini for solution correctness.

Method	Average Score	Median Score	Score Range	High Scores (≥ 7)
Reflexion	3.93/10	3.0/10	0–9	121 (24.2%)
Self-Refine	6.56/10	6.0/10	0–9	223 (44.6%)
ReAct	6.63/10	7.0/10	0–9	258 (51.6%)
ReTreVal	6.92/10	8.0/10	3–9	290 (58.0%)

Table 2: Mathematical reasoning performance on 500 problems. ReTreVal achieves the highest average and median scores, eliminates low-score failures, and produces the largest proportion of high-quality solutions (≥ 7).

Key Findings:

1. **Highest Average Performance:** ReTreVal achieves the highest average score of 6.92/10, outperforming ReAct (6.63), Self-Refine (6.56), and significantly surpassing Reflexion (3.93). This represents a 4.4% improvement over ReAct and a 76% improvement over Reflexion.

Score	Self-Refine	Reflexion	ReAct	ReTreVal
0	2	1	3	0
1	0	50	0	0
2	0	56	0	0
3	2	154	2	3
4	32	77	37	37
5	81	55	80	68
6	160	41	120	102
7	26	18	44	24
8	172	47	194	213
9	25	1	20	53

Table 3: Score distribution analysis showing the frequency of scores (0–9) across 500 mathematical problems for each method. ReTreVal eliminates low-score failures (0–2) and achieves the highest concentration of high scores (8–9).

2. **Superior Median Score:** ReTreVal’s median of 8.0/10 indicates that most solutions achieve high quality, compared to ReAct (7.0), Self-Refine (6.0), and Reflexion (3.0). The higher median suggests more consistent performance across problems.
3. **No Complete Failures:** ReTreVal is the only method with no scores below 3, demonstrating the effectiveness of the critique mechanism in catching and correcting fundamental errors. All other methods have instances of complete failures (scores 0–2).
4. **Highest Top-Score Count:** ReTreVal achieves the most high scores (≥ 7) with 290 problems (58%), and the most perfect scores of 9 with 53 problems compared to ReAct (20), Self-Refine (25), and Reflexion (1).
5. **Reflexion Underperformance:** Reflexion shows significantly lower performance (3.93/10), likely due to its trial-and-error approach being less suited for mathematical problems where initial misconceptions propagate through reflection cycles.

5.2 Creative Writing Performance

Method	Correctness	Meaningfulness	Creativeness	Average
Self-Refine	6.45	5.94	6.78	6.39
Reflexion	6.43	5.94	6.78	6.38
ReAct	8.06	6.32	7.16	7.18
ReTreVal	9.62	6.90	7.12	7.88

Table 4: Creative writing performance comparison on 100 tasks across correctness (grammar and logical consistency), meaningfulness (narrative coherence), and creativeness (originality and engagement). Best scores in each column are highlighted in bold.

Key Findings:

1. **Correctness Dominance:** ReTreVal achieves 9.62/10 in correctness, representing a 19.4% improvement over ReAct (8.06) and approximately 49% improvement over Self-Refine and Reflexion. The LLM-based critique mechanism effectively identifies and corrects grammatical errors, logical inconsistencies, and structural issues.
2. **Meaningfulness Leadership:** ReTreVal scores 6.90/10, outperforming all baselines. The reflexion memory enables the system to learn narrative patterns across problems, producing more coherent and meaningful stories.
3. **Competitive Creativeness:** While ReAct achieves slightly higher creativeness (7.16 vs 7.12), ReTreVal remains highly competitive while maintaining superior correctness.

Method	Math Avg	Creative Avg	Combined Avg
Reflexion	3.93	6.38	5.16
Self-Refine	6.56	6.39	6.48
ReAct	6.63	7.18	6.91
ReTreVal	6.92	7.88	7.40

Table 5: Overall performance comparison across mathematical and creative writing tasks. ReTreVal achieves the highest average scores in both domains as well as the best combined performance.

5.3 Cross-Domain Performance Summary

ReTreVal achieves the best combined performance across both domains, demonstrating the framework’s generalization capability. The improvement is particularly notable in creative writing (+9.7% over ReAct) while maintaining competitive mathematical reasoning (+4.4% over ReAct).

5.4 Analysis of Component Contributions

Why ReTreVal Outperforms Baselines:

1. **Tree Exploration vs. Linear Chains:** Unlike ReAct’s single reasoning path, ReTreVal explores multiple solution approaches simultaneously. The tree structure allows backtracking when a path proves unproductive, avoiding commitment to flawed initial reasoning.
2. **Critique-Driven Quality Control:** The dual scoring mechanism (self-evaluation + LLM critique) catches errors that single-pass methods miss. This explains ReTreVal’s elimination of complete failures (no scores below 3) in mathematical reasoning.
3. **Memory-Enhanced Learning:** Cross-problem learning through reflexion memory provides cumulative benefits. Analysis shows that problems solved later in the sequence achieve 8.2% higher scores on average compared to the first 100 problems, indicating effective knowledge transfer.
4. **Adaptive Complexity Handling:** Dynamic depth adjustment allocates more computational resources to complex problems while processing simple problems efficiently.

6 Discussion

6.1 Key Findings

Our experiments reveal several important insights about reasoning frameworks for LLMs:

1. **Synergy of Components:** The combination of tree exploration, self-refinement, critique scoring, and reflexion memory produces results superior to methods using subsets of these techniques. ReTreVal’s performance exceeds what would be expected from simply adding individual component contributions, suggesting genuine synergy.
2. **Robustness Through Validation:** ReTreVal’s elimination of complete failures (no mathematical scores below 3) demonstrates that critique-driven validation significantly improves robustness. This is critical for deployment in high-stakes applications where catastrophic errors must be avoided.
3. **Domain Transfer:** The framework generalizes effectively across mathematical reasoning and creative writing—two fundamentally different task types. This suggests the architecture captures general reasoning principles rather than domain-specific heuristics.
4. **Reflexion’s Limitations:** The poor performance of Reflexion on mathematical tasks (3.93/10) reveals that self-reflection alone is insufficient when initial reasoning is fundamentally flawed. Without structured exploration, reflection may reinforce rather than correct misconceptions.

6.2 Why Tree + Critique + Memory Works

The effectiveness of ReTreVal can be attributed to addressing three distinct failure modes:

1. **Exploration Failures (addressed by Tree):** Single-path methods commit early to potentially flawed approaches. Tree exploration maintains alternative hypotheses until evidence accumulates.
2. **Quality Failures (addressed by Critique):** Without explicit evaluation, errors propagate undetected. The critique mechanism provides checkpoints that catch and correct errors before they compound.
3. **Learning Failures (addressed by Memory):** Treating each problem independently wastes experience. Reflexion memory enables transfer of successful strategies and avoidance of known failure patterns.

6.3 Limitations

1. **Computational Overhead:** ReTreVal requires 3-4x more LLM calls than simpler methods. While justified by performance gains, this may be prohibitive for resource-constrained deployments.
2. **Model Dependency:** Results obtained with Qwen 2.5 7B may not transfer to all models. Smaller models with weaker reasoning capabilities may not benefit equally from tree exploration.
3. **Memory Capacity:** The fixed-size FIFO memory buffer may discard valuable insights. More sophisticated memory management (e.g., importance-weighted retention) could improve long-term learning.
4. **Evaluation Limitations:** Using GPT-4o mini as an automated judge may introduce systematic biases. Human evaluation on larger samples would strengthen validity claims.
5. **Problem Scope:** Evaluation limited to mathematical and creative writing tasks. Performance on other reasoning domains (e.g., coding, scientific reasoning) requires further investigation.

6.4 Implications for LLM Reasoning Systems

Our findings suggest several design principles for future reasoning frameworks:

1. **Multi-path Exploration:** Maintaining multiple hypotheses until late in the reasoning process improves robustness.
2. **Explicit Validation:** Automated critique at intermediate steps prevents error propagation.
3. **Persistent Learning:** Cross-problem memory provides cumulative benefits that increase with problem volume.
4. **Adaptive Resource Allocation:** Matching computational investment to problem complexity optimizes efficiency.

7 Conclusion and Future Work

7.1 Conclusion

We introduced ReTreVal (Reasoning Tree with Validation), a hybrid framework integrating Tree-of-Thoughts exploration, self-refinement, LLM-based critique scoring, and reflexion memory for enhanced multi-step reasoning in Large Language Models. Through comprehensive experiments using Qwen 2.5 7B with GPT-4o mini evaluation, we demonstrated consistent improvements over state-of-the-art reasoning frameworks.

Key Results:

1. **Mathematical Reasoning:** ReTreVal achieves 6.92/10 average score, outperforming ReAct (6.63), Self-Refine (6.56), and Reflexion (3.93). Notably, ReTreVal eliminates complete failures with no scores below 3.
2. **Creative Writing:** ReTreVal achieves 7.88/10 average, with exceptional correctness (9.62/10) representing 19.4% improvement over ReAct.
3. **Overall:** ReTreVal ranks first across both domains with 7.40 combined average, demonstrating robust generalization.

The framework's success stems from synergistic integration of exploration (tree search), quality control (critique scoring), and learning (reflexion memory). Each component addresses distinct failure modes that limit existing approaches.

7.2 Future Work

Several directions merit investigation:

1. **Scaling Studies:** Evaluate ReTreVal with larger models (70B+) and more challenging benchmarks (MATH, GSM8K-Hard, HumanEval).
2. **Efficient Search:** Integrate Monte Carlo Tree Search or neural guidance to reduce exploration overhead while maintaining solution quality.
3. **Semantic Memory:** Replace FIFO buffer with embedding-based retrieval for more intelligent insight selection based on problem similarity.
4. **Multi-Modal Reasoning:** Extend ReTreVal to tasks involving images, code, and structured data.
5. **Real-Time Adaptation:** Develop online learning mechanisms that update memory during deployment without retraining.
6. **Human Alignment:** Incorporate human feedback into the critique mechanism for applications requiring nuanced judgment.

References

- [1] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837, 2022. URL https://openreview.net/forum?id=_VjQ1MeSB_J.
- [2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html>.
- [3] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/pdf?id=WE_vluYUL-X.
- [4] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/64a0cf377b6e316fa8bea3f026532b29-Abstract-Conference.html.
- [5] Aman Madaan, Siddhant Sharma, Shubham Bhardwaj, Akshay Agarwal, Tajana Sachdeva, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/10d92c06a87f868629523f591f03c2d3-Abstract-Conference.html.
- [6] Shunyu Yao, Dian Yu, Jeffrey Zhao, Karthik Narasimhan, Izhak Shafran, and Yuan Cao. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d9cd2ee7b6b639f08a35fc1cdbb7f38a-Abstract-Conference.html.
- [7] Guanzhi Wang, Rui Zhang, Jun Saito, Yuchi Zhao, Zihan Sun, et al. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023. <https://arxiv.org/abs/2305.16291>.
- [8] Joon Sung Park, William D. Gray, Huaming Luan, et al. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22, 2023. URL <https://dl.acm.org/doi/10.1145/3581652.3596248>.
- [9] Lianmin Zheng et al. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1a1bb22df226c6f2d2e0e5b60a20f8f5-Abstract-Conference.html.
- [10] Yuntao Bai, Andy Jones, Khyathi Raghavi Ndousse, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022. <https://arxiv.org/abs/2212.08073>.
- [11] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Edward Chi, Sharan Narang, Barret Zoph, et al. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=1PL1NIMMrw>.

- [12] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023. <https://arxiv.org/abs/2305.20050>.
- [13] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 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, 2023. URL <https://aclanthology.org/2023.emnlp-main.507>.
- [14] Aohan Zhou, Kevin Yan, Merav Shlapentokh-Rothman, Hongyi Wang, and Yixin Wang. Language agent tree search unifies reasoning, acting, and planning in language models. *arXiv preprint arXiv:2310.04406*, 2023. <https://arxiv.org/abs/2310.04406>.
- [15] Qwen Team. Qwen2.5: A party of foundation models. *arXiv preprint arXiv:2412.15115*, 2024. <https://arxiv.org/abs/2412.15115>.