METAEVO: A META-OPTIMIZATION FRAMEWORK FOR EXPERIENCE-DRIVEN AGENT EVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing methods for experience-driven agent evolution often generate revision principles that lack actionable guidance, as they overlook the inherent limitations of Large Language Models (LLMs) in abstracting knowledge and self-correcting. To address this gap, we introduce MetaEvo, a novel framework that reframes agent evolution as a meta-optimization task. Instead of learning directly from experience, our core idea is to first enhance the agent's intrinsic meta-ability—its capacity to learn how to effectively revise and improve itself. MetaEvo operationalizes this concept through a three-stage pipeline powered by a modular agent system. First, a meta-optimization stage explicitly trains the model on abstracting high-quality principles. These principles are then accumulated in a curated memory and subsequently retrieved by an execution module to guide generation on new tasks. Extensive experiments across a diverse suite of mathematical reasoning and multi-task benchmarks demonstrate that MetaEvo consistently and significantly improves model performance, enabling more robust and generalizable reasoning behaviors.

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks (Brown et al., 2020; Touvron et al., 2023). A growing body of research indicates that LLMs can further self-improve by learning from their own outputs and interactions (Madaan et al., 2023; Shinn et al., 2023; Li et al., 2023; Azov et al., 2024; Gou et al., 2024). This self-improvement process is often structured within agent systems, which augment LLMs with persistent memory and explicit workflows to enhance their problem-solving in complex domains (Madaan et al., 2023; Shinn et al., 2023). A key frontier in this domain is achieving continuous evolution, where an agent not only completes tasks but also progressively refines its underlying strategies. However, designing a robust and scalable framework for such evolution remains a significant challenge.

To enable continuous evolution, many current approaches are experience-driven, leveraging past interactions to guide future behavior. These methods typically involve injecting guidance during inference, such as curated principle rules (Sun et al., 2023), procedural exemplars (Zhao et al., 2024; Li & Qiu, 2023), or textual feedback, often supported by memory modules for reuse (Gao et al., 2024; Chen et al., 2024). While promising, these methods face several critical limitations.

First, a trade-off exists between the quality and scalability of the guiding principles. Manually authored principles ensure high quality but are labor-intensive and difficult to scale (Chen et al., 2023; Sun et al., 2023), while automatically extracted principles offer scalability but are often too generic to be actionable (Yang et al., 2023; Chen et al., 2024). Second, and more fundamentally, most approaches focus on the content of an experience, such as specific rules or answers. In doing so, they largely overlook the agent's latent abstraction capability, which is the ability to derive and internalize correction strategies from examples (Li & Qiu, 2023; Gao et al., 2024). This neglects the crucial opportunity to treat the process of learning from experience not as mere knowledge accumulation, but as a meta-learning problem aimed at improving the model's underlying reasoning and self-correction abilities.

To address these challenges, we propose **MetaEvo**, a novel framework that synergizes **Meta**-optimization with agent **Evo**lution to create more robust and continually improving agents. The core of our framework is a meta optimization stage that fundamentally enhances the agent's ability to learn how to self-correct, rather than merely what to answer. The agent's performance is fur-

ther bolstered by our proposed contrastive driven abstraction method for generating high-quality principles and a principle accumulation mechanism for creating a reusable knowledge base.

Specifically, the MetaEvo framework is structured around the following three-stage pipeline: The first stage is (1) Meta Optimization, where the agent learns how to generate effective revision principles, rather than just what to answer. This is followed by (2) Principle Accumulation, where the high-quality principles generated by the enhanced agent are systematically curated and stored. Finally, in the (3) Principle-Guided Generation stage, these stored principles are retrieved to guide the agent's actions during inference. This entire pipeline is powered by a modular agent system comprising three specialized components: a *plan* module for principle abstraction, a *memory* module for storage and retrieval, and an *execution* module for principle-informed generation.

We conduct extensive experiments across five challenging benchmarks spanning both reasoning and multi-task settings. The results demonstrate that MetaEvo consistently improves the base LLM's performance across all tasks. This improvement holds across different model scales, demonstrating MetaEvo's generality. Notably, MetaEvo is especially effective on tasks requiring multi-step reasoning, suggesting its advantage in handling complex failure patterns. Our main contributions are threefold:

- We propose METAEVO, a novel framework that operationalizes agent evolution as a meta-optimization task. The framework features a three-stage pipeline implemented by a modular agent system with specialized *plan*, *memory*, and *execution* components.
- We introduce a powerful principle learning mechanism featuring two key technical contributions: a meta-optimization stage that employs preference-based learning to teach the agent how to abstract effective principles, and a Contrast-Driven Abstraction (CDA) method that ensures the high quality and actionability of these principles.
- We demonstrate through extensive experiments that MetaEvo achieves state-of-the-art performance on five demanding benchmarks: GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), MATH (Hendrycks et al., 2021), MMLU (Hendrycks et al., 2020), and BBH (Suzgun et al., 2023). Further analysis validates the effectiveness of each component within our framework.

1 RELATED WORK

1.1 EXPERIENCE-DRIVEN EVOLUTION.

Recent work has shifted toward building self-evolving agents. A recent survey (ang Gao et al., 2025) highlights core dimensions of this paradigm and stresses the importance of continual learning from experience. Within this broader context, several studies focus on identifying model failures and generating revision principles that serve as auxiliary training signals or external memory (Sun et al., 2023; Yang et al., 2023; Madaan et al., 2023). Others enhance LLM adaptability through retrieval-augmented architectures and memory-based prompting, allowing models to reference historical corrections during inference (Gao et al., 2024; Zhao et al., 2024; Gong et al., 2024). These approaches enable long-term learning via structured feedback loops and continual exposure to prior mistakes.

However, these experience-driven methods often rely on complex multi-stage pipelines to extract or apply revision signals, while overlooking the model's inherent limitations in abstraction and error correction. As a result, the generated principles may lack actionable guidance or fail to generalize across tasks. Our framework addresses this gap by introducing a meta-optimization mechanism that strengthens the model's ability to abstract, internalize, and reuse correction principles.

1.2 MEMORY-BASED AGENT SYSTEM

A key direction in advancing LLM agents lies in equipping them with external memory systems that support continual learning, behavioral adaptation, and long-horizon planning (Li & Qiu, 2023; Yang et al., 2023). Such memory can take multiple forms, which we broadly categorize into experience replay, structural memory organization, and memory management strategies. Experience replay

methods store intermediate reasoning trajectories or error-feedback pairs to guide subsequent decisions without modifying model parameters (Li & Qiu, 2023; Gao et al., 2024). These approaches improve accuracy but often depend on principle strategies and lack principled abstraction, limiting their ability to generalize across tasks. Structural memory methods organize stored information into functional segments such as episodic and semantic memory, enhancing interpretability and transferability (Zeng et al., 2024; Zhao et al., 2024). Finally, memory management research investigates mechanisms for memory retention, forgetting, and retrieval policies that influence agents' long-term behavior (Xiong et al., 2025; Zeng et al., 2024).

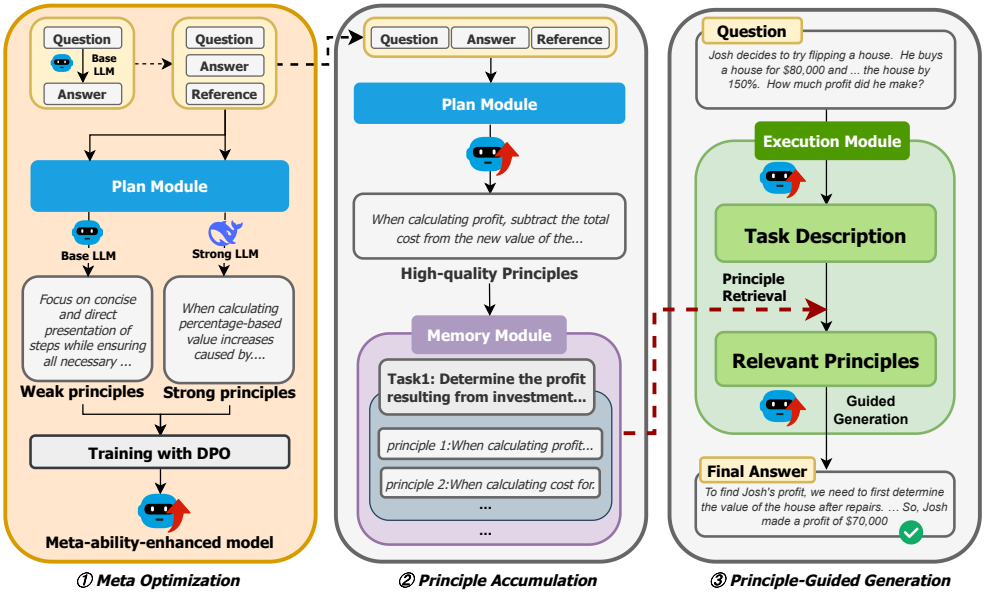

Figure 1: Illustration of the three-stage evolutionary process of the METAEVO framework. 1. **Meta Optimization:** We first train a model to enhance its core meta-ability through preference-based learning on principles. 2. **Principle Accumulation:** The enhanced model then abstracts and accumulates a refined set of principles into a structured *memory* module, which can be iteratively expanded. 3. **Principle-Guided Generation:** At inference time, the agent retrieves the most relevant principles from memory to steer its final response, ensuring strategically sound outputs.

2 METHODOLOGY

This section presents MetaEvo, a meta-optimization framework implemented as a modular agent system, that enables experience-driven and principle-guided evolution through its three core modules: *plan*, *memory*, and *execution*. Accordingly, we first present the overall workflow of the framework, and then delve into the specifics of each constituent module.

2.1 FRAMEWORK WORKFLOW

2.1.1 META OPTIMIZATION

In this stage, we fine-tune the base model to enhance its "meta-ability" to learn how to revise and improve its own outputs. This is achieved by framing the learning process as a meta-optimization task: instead of learning to solve the task directly, the model learns a preference for generating more effective revision principles. The outcome of this stage is a meta-ability-enhanced model trained on a specially constructed preference dataset of these principles.

For each input x, we first have the base model generate an initial answer. A *plan* module then uses this input, the generated answer, and a reference answer to abstract a revision principle. We perform

this abstraction process twice: once using the base model itself to yield a "weaker" principle p^-, and once using a more powerful "strong model" to yield a "stronger" principle p^+. This creates a preference pair (p^+, p^-) for the input x. Repeating this yields the full dataset:

$$\mathcal{D}_{\text{meta}} = \left\{ \left(x_i, p_i^+, p_i^- \right) \right\}_{i=1}^{N} \tag{1}$$

We then train the base model on this dataset using Direct Preference Optimization (DPO) Rafailov et al. (2023). The objective is to minimize the expected loss, guiding the model to prefer the stronger revision principles:

$$\min_{\theta} \ \mathbb{E}_{(x,p^+,p^-)\sim\mathcal{D}_{\text{meta}}} \left[\mathcal{L}_{\text{meta}}(f_\theta; x, p^+, p^-) \right] \tag{2}$$

where f_θ is the model parameterized by θ. The loss is to minimize a pairwise preference loss, encouraging the model to assign higher probability to the preferred principle p^+ relative to the dispreferred p^-::

$$\mathcal{L}_{\text{meta}} = -\,\mathbb{E}_{(x,p^+,p^-)} \left[\log \sigma \Big(\beta \big(\log \pi_\theta(p^+ \mid x) - \log \pi_\theta(p^- \mid x) \big) \Big) \right]$$

Here, $\pi_\theta(p \mid x)$ is the model's probability of generating principle p given input x, $\sigma(\cdot)$ is the sigmoid function, and β is a temperature parameter.

Through this process, we obtain a **meta-ability-enhanced** model that demonstrates an improved capacity for abstracting revision principles and correcting failure cases, laying the foundation for subsequent evolution stages.

2.1.2 Principle Accumulation

The goal of this stage is to leverage this enhanced model to build a rich, structured repository of high-quality principles. We achieve this by using the enhanced model to power the *plan* module, processing the same training data as in the previous stage. This generates a new corpus of principles that exhibit greater generalizability and instructional value.

These refined principles are then systematically organized and stored in the *memory* module. They are typically indexed by the semantic representations of their corresponding tasks, creating a task-oriented knowledge base. This memory forms the foundation for the final generation stage.

Crucially, this entire process is designed to be iterative. The model, now augmented with an external memory of principles, can be used as the new base model for another full cycle of meta-optimization and principle accumulation. Through such iterations, the memory continuously expands and the model's capabilities progressively strengthen, facilitating a cycle of continual evolution.

2.1.3 Principle-Guided Generation

This stage describes the inference process, where the structured knowledge accumulated in the *memory* module is actively utilized to guide the generation of responses.

When a new input is presented to the system, the *execution* module is activated. Its first step is to generate a task-level semantic descriptor of the input. This descriptor is then used as a query to retrieve the most relevant principles from the *memory* module based on semantic similarity.

The retrieved principle serves as direct, actionable guidance for the final response generation. It is incorporated into the model's context, for instance, as part of the prompt or as a specific instruction, to steer the output. This ensures that the model's final answer is not only relevant to the input but is also informed by the most effective, high-quality strategies learned throughout the evolution process. This mechanism is the essence of principle-guided generation.

2.1.4 Plan Module

The *plan* module constitutes the central reasoning engine of our agent system. Its primary function is to diagnose deficiencies in a given response and abstract a generalizable principle for improvement. To achieve this in a structured manner, we introduce a method named **Contrast-Driven Abstraction (CDA)**.

The CDA method operates via a two-step, LLM-driven process: (1) Discrepancy Analysis and (2) Principle Abstraction, as illustrated in Figure 2.

In the first step, Discrepancy Analysis, we prompt the LLM for analysis, with the user's question q, the base model's answer x, and an expert reference y. This step aims to perform a fine-grained comparison and output a structured analysis, Δ:

$$\Delta = f_{\text{analyze}}(q, x, y) \tag{3}$$

The resulting structure Δ details each identified discrepancy as a list containing four fields: the `aspect` of comparison (e.g., factual accuracy), the `highquality` excerpt from the reference, the corresponding `lowquality` flaw, and a summary of the `differences`.

In the second step, Principle Abstraction, this structured analysis Δ is fed to the LLM for abstraction. This model's role is to synthesize the detailed findings and distill a single, high-quality revision principle, p:

$$p = f_{\text{abstract}}(\Delta) \tag{4}$$

The principle is a concise, actionable directive expressed in natural language, encapsulating the core lesson from the contrastive analysis. These extracted principles form the fundamental building blocks for the subsequent stages of agent evolution.

2.2 Agent System Modules

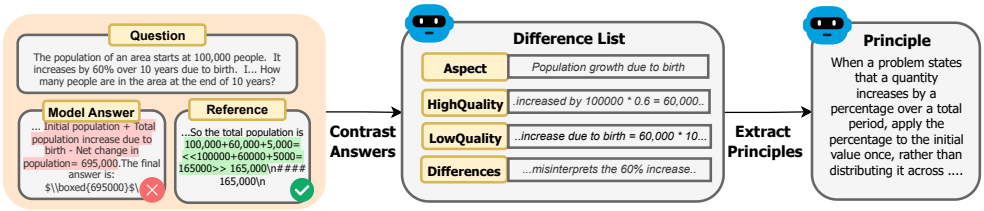

Figure 2: Illustration of the *plan* module:Abstract principles via Contrastive Driven Abstraction.

2.2.1 Memory Module

The *memory* module functions as the agent's long-term knowledge repository, designed to systematically store and manage the principles generated by the *plan* module. We define the memory $\mathcal{M}$ as a key-value structure, where each key represents a task descriptor t and the value is a set of principles $\mathcal{P}_t$ relevant to that task:

$$\mathcal{M} = \{t \mapsto \mathcal{P}_t \mid t \in \mathcal{T}\} \tag{5}$$

Here, $\mathcal{T}$ is the set of all task descriptors. Each descriptor t is itself a concise, natural-language summary of an input query q, produced by an LLM to capture the query's core intent.

To ensure the quality and efficiency of the repository, the memory is not merely appended but is actively curated. When a new principle p_{new} is generated for a task t, it undergoes a validation process against the existing principles $\mathcal{P}_t$ for that task. This process, mediated by an LLM-based evaluator, checks for semantic redundancy and logical conflicts. For each existing principle $p \in \mathcal{P}_t$, the evaluator assesses the pair (p_{new}, p) and follows a set of update rules:

- **Semantic Equivalence:** If p_{new} is judged to be a rephrase or a minor variant of an existing principle, p_{new} replaces the old one. This ensures the memory reflects the agent's most up-to-date reasoning.
- **Logical Conflict:** If p_{new} contradicts an existing principle, an LLM-based arbiter is invoked to select the one with superior generalizability or correctness. The winner is retained, and the other is discarded.
- **No Significant Overlap:** If p_{new} is determined to be novel and non-conflicting, it is directly added to the memory set for task t. This update can be expressed as:

$$\mathcal{P}_t \leftarrow \mathcal{P}_t \cup \{p_{\text{new}}\} \tag{6}$$

This curation mechanism prevents the accumulation of redundant or flawed information. It allows the memory to evolve into a compact, coherent, and non-redundant knowledge base, providing the agent with a refined set of high-quality principles to draw upon during the generation stage.

2.2.2 EXECUTION MODULE

The *execution* module is the agent's action-taking component, responsible for generating the final, principle-guided response during inference. It leverages the curated knowledge within the *memory* module to inform its generation process, which unfolds in two phases: Principle Retrieval and Guided Generation.

The retrieval phase begins with a new query q. The module first generates a corresponding task descriptor, t_q, which serves as a semantic key for searching the memory. This is typically done by prompting an LLM: $t_q = f_{\text{describe}}(q)$. This key is then compared against all task descriptors $\mathcal{T}$ stored in the memory $\mathcal{M}$ to find the best match, t^*, based on semantic similarity:

$$t^* = \arg\max_{t \in \mathcal{T}} \text{sim}(t_q, t) \tag{7}$$

where $\text{sim}(\cdot, \cdot)$ is a semantic similarity function, such as cosine similarity over text embeddings. If the similarity score of the best match exceeds a predefined threshold τ_r, the entire set of principles associated with t^* is retrieved. The retrieved set, $\mathcal{P}_{\text{retrieved}}$, is therefore defined as:

$$\mathcal{P}_{\text{retrieved}} = \begin{cases} \mathcal{P}_{t^*} & \text{if } \text{sim}(t_q, t^*) > \tau_r \\ \emptyset & \text{otherwise} \end{cases} \tag{8}$$

In the Guided Generation phase, the retrieved principles $\mathcal{P}_{\text{retrieved}}$ (if any) are incorporated into the context for a generation LLM, f_{gen}. These principles act as explicit, context-aware instructions or constraints, steering the model's output. The final response is thus generated with the benefit of proven, task-relevant strategies:

$$\text{response} = f_{\text{gen}}(q, \mathcal{P}_{\text{retrieved}}) \tag{9}$$

By dynamically retrieving and applying relevant knowledge, the *execution* module allows the agent to generalize from past experiences to new, unseen problems, ensuring its responses are not only accurate but also strategically sound.

3 EXPERIMENTS

3.1 SETTINGS

Benchmarks We evaluate the performance of MetaEvo on five benchmarks. GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), and MATH (Hendrycks et al., 2021) are employed to assess arithmetic and advanced mathematical reasoning. To evaluate factual knowledge, logical deduction, and generalization ability, we select representative subsets from MMLU (Hendrycks et al., 2020) and BBH (Suzgun et al., 2023). For MMLU, we include 10 subject areas such as High School Mathematics, Physics, Philosophy, and Computer Science. For BBH, we focus on 8 reasoning-intensive tasks, including Dyck language recognition, logical deduction, and causal judgment, covering symbolic, analogical, and commonsense reasoning types. This subset was chosen to better reflect our goal of evaluating principle abstraction and transfer, while keeping experiments computationally feasible and consistent with prior work for fair comparison.

Models We conduct experiments using two backbone language models: LLaMA 3.1-8B-Instruct (Dubey et al., 2024) and Qwen 2.5-14B-Instruct (Yang et al., 2024). These models are used as the initial agent foundation for generation, principle abstraction, and memory interaction. We use DeepSeek-R1 (DeepSeek-AI et al., 2025) as the strong language model to generate high-quality reference answers and better principles. It serves as an external oracle for constructing preference pairs used in meta optimization.

Experimental Setup We conduct all experiments using low-rank adaptation (LoRA) with a rank of 8 and a scaling factor of 32, applied to all linear modules in the transformer architecture. The model is trained for one epoch with a maximum of 400 steps. We adopt a learning rate of 1×10^{-4}, a per-device batch size of 1 for both training and evaluation, and apply gradient accumulation with one step. The warmup ratio is set to 0.05. All training is performed using bfloat16 precision. Training and inference are run on NVIDIA A6000 GPUs with 48GB memory.

3.2 BASELINE

We employ the following baseline methods. (1) **Base Model**: The original model without additional optimization; (2) **Self-Refine**: A self-evolution method based on feedback and self-correction; (3) **Self-Evolving GPT**: A self-evolution framework that induces and reuses task-specific experience to improve reasoning performance. (4) **MetaEvo w/o MO**: Our framework without applying meta-optimization; (5) **MetaEvo w/o CDA**: Our framework without applying Contrast Driven abstraction(CDA); (6) **MetaEvo**: The full framework combining meta optimization with CDA to enable principle-guided self-evolution.

Method	**LLaMA3.1-8B-Instrcut**						**Qwen2.5-14B-Instrcut**					
	GSM8K	SVAMP	MATH	MMLU	BBH	Avg.	GSM8K	SVAMP	MATH	MMLU	BBH	Avg.
Base Model	84.5	88.2	60.2	66.7	61.1	72.1	92.2	91.5	71.1	76.5	76.1	81.5
Self-Refine	77.9	79.8	57.5	50.3	59.7	65.0	89.5	87.8	67.5	67.1	70.9	76.6
Self-Evolving	84.7	86.1	61.2	<u>68.5</u>	<u>63.9</u>	72.9	93.7	93.3	<u>72.8</u>	**80.2**	<u>78.3</u>	<u>83.7</u>
MetaEvo w/o CDA	<u>88.7</u>	81.3	59.3	67.9	61.6	71.8	91.9	87.8	70.2	76.2	75.3	80.3
MetaEvo w/o MO	88.3	<u>90.5</u>	<u>63.4</u>	67.3	62.4	<u>74.4</u>	<u>95.1</u>	<u>93.7</u>	72.2	78.1	76.4	83.1
MetaEvo	**94.1**	**93.8**	**66.3**	**69.1**	**64.7**	**77.6**	**97.1**	**95.2**	**73.2**	<u>78.7</u>	**77.9**	**84.4**

Table 1: Performance (%) of reasoning and self-improvement methods across benchmarks.**Bold numbers** represent the best performance on each dataset, while <u>underlined numbers</u> denote the second-best results.

4 ANALYSIS

4.1 VALIDATING METAEVO EVOLUTION FRAMEWORK

Table 1 presents the evaluation results of our method on five benchmark datasets. Compared to the base models, our approach consistently yields performance improvements across all datasets.

On GSM8K and SVAMP, which focus on numerical reasoning, our method yields substantial accuracy gains, reflecting enhanced capabilities in error correction and generalization. On MATH, the framework demonstrates robustness in handling complex multi-step reasoning. Further improvements on MMLU and BBH confirm the generalizability of MetaEvo across diverse knowledge-intensive tasks.

These results underscore the effectiveness of MetaEvo in enhancing both task-specific accuracy and cross-task generalization.

4.2 META ABILITY ENHANCES THE MODEL'S CAPACITY FOR SELF-IMPROVEMENT

Meta optimization plays a pivotal role in enabling effective principle abstraction and guided generation. To assess its impact, we compare the full MetaEvo framework with a variant without meta optimization (denoted as w/o MO in Table 1). While the w/o MO variant achieves competitive results on GSM8K and SVAMP, it consistently underperforms the full framework across all tasks. The performance gap underscores the importance of meta-level optimization in refining correction principles and enhancing generalization across reasoning scenarios.

Further analysis reveals that such principles are often overly generic, lacking task-specific utility and causing redundant or inconsistent outputs. In contrast, our method consistently surpasses baselines. The optimized principles are more targeted and actionable, enabling precise, failure-aligned

revisions. These findings underscore the necessity of meta optimization in transforming abstract principles into actionable tools for self-correction and improvement.

Figure 3 illustrates the sample efficiency advantage of meta-optimization. We compare two training paradigms. In result-based training, the model is optimized to predict the correct answer directly from the input, with no intermediate supervision. In contrast, principle-based training (MetaEvo) first learns to generate and refine intermediate principles, which are stored and later retrieved to guide answer generation. This additional layer of structured supervision allows MetaEvo to achieve stronger performance with fewer labeled examples.

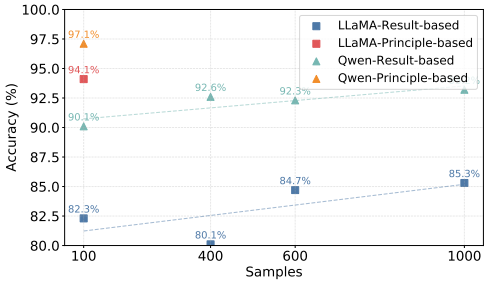

Figure 3: The figure compares two training methods on GSM8K. Our method achieves higher performance with fewer training samples.

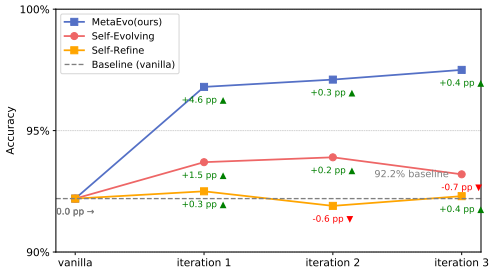

Figure 4: Accuracy comparison across iterations for MetaEvo, Self-Evolving, and Self-Refine. MetaEvo on GSM8K achieves consistent improvements and outperforms baselines at all iterations.

Method	LLaMA3-8B-Instruct		Qwen2.5-14B-Instruct	
	GSM8K	SVAMP	GSM8K	SVAMP
Base Model	84.5	88.2	92.2	91.5
Iteration 1	92.4	91.9	96.8	94.8
Iteration 2	93.7	92.6	**97.1**	94.6
Iteration 3	**94.1**	**93.8**	96.7	**95.2**

Table 2: Performance (%) on GSM8K and SVAMP for LLaMA3-8B-Instruct and Qwen2.5-14B-Instruct across iterative optimization rounds.

Enhanced meta-ability of principle abstraction Raises the performance ceiling in iterative improvement.

We further investigate the impact of iterative evolution within our framework. Specifically, we conduct up to three full iterations of the MetaEvo pipeline on GSM8K and SVAMP, using `LLaMA3.1-8B-Instruct` and `Qwen2.5-14B-Instruct` as base models. In each iteration, we execute principle accumulation followed by principle-guided generation, where the model's output from the previous round serves as the input for the next. After three rounds, both the principle memory and the quality of model responses demonstrate progressive enhancement.

As shown in Figure 4, MetaEvo exhibits consistent performance improvements across iterations and consistently outperforms other self-evolving baselines, demonstrating its enhanced capacity to support continual self-improvement through effective principle accumulation. Table 2 provides detailed results. The model's performance increases consistently over three iterations, indicating that the progressive refinement of principles directly contributes to improved reasoning quality. After three rounds, the model achieves an absolute improvement of 9.6% in GSM8K and 5.6% in SVAMP relative to the baseline, with notable gains even in the final iteration (0.4% and 1.2%, respectively).

These findings underscore the distinctive advantage of meta-ability optimization: by emphasizing the acquisition of improvement strategies over direct answer generation, the model can iteratively refine its behavior through feedback, progressively achieving higher-quality generation.

Meta-Optimization enhances the intrinsic reasoning ability of the model, even without explicit reuse of previous experience. Our results demonstrate that directly strengthening the model's capacity for principle abstraction via meta-optimization leads to consistent performance gains. As shown in Table 3, models trained with preference-based supervision outperform their base counterparts across multiple reasoning tasks, confirming that abstract principle alignment can improve general reasoning behavior independent of memory retrieval or prior instance reuse.

We argue that this abstraction process represents a meta-level capability that operates above specific instances, enabling the model to generalize from error patterns and revise its own behavior accordingly. Rather than memorizing task-specific corrections, the model learns to organize and apply high-level strategies that support more coherent, self-aware reasoning. This improvement feeds back into downstream performance, as the model internalizes a transferable scaffold for decision-making. In essence, principle abstraction serves not just as a mechanism for fixing past mistakes, but as a foundation for building more adaptive and generalizable reasoning behavior.

Method	GSM8K	SVAMP
LLaMA3.1-8B		
Base Model	84.5	88.2
w/ MO	**86.7**	**90.5**
Qwen2.5-14B		
Base Model	92.2	91.5
w/ MO	**95.2**	**93.7**

Table 3: Comparison of base models with and without Meta Optimization (MO) on GSM8K and SVAMP.

Method	GSM8K	SVAMP
Base model	84.5	88.2
Random principles	69.4	77.9
Meta-evo w/o CDA	82.7	81.5
Meta-evo w/ CDA	92.4	91.9

Table 4: Another experimental setting with Meta Optimization (MO) results.

4.3 Contrastive Analysis Drives Precise and Actionable Principle Abstraction

Effective evolution relies on high-quality principles, and contrastive analysis is key to extracting effective principles. To assess the impact of different principle generation strategies, we compare three settings: (1) *Random principles*, which introduce task-irrelevant noise; (2) *Direct abstraction* (MetaEvo w/o CDA), in which principles are generated without contrastive analysis; and (3) *Contrastive analysis* (MetaEvo w/ CDA), which employs contrastive-driven abstraction to derive principles.

As shown in Table 4, random principles significantly degrade performance, confirming that irrelevant guidance can mislead the model's reasoning process. Direct abstraction produces unguided principles, which may introduce noise or even conflict with the model's original reasoning trajectory. In contrast, principles derived through contrastive analysis yield the highest and most consistent performance, achieving 92.4% on GSM8K and 91.9% on SVAMP, demonstrating their effectiveness in guiding generation. These results validate the crucial role of contrastive analysis in extracting reliable, high-quality principles that serve as effective guidance for principle-guided generation

5 Conclusion

In this paper, We introduce MetaEvo, a meta-optimization framework that facilitates principle-guided evolution in large language models. By enhancing the model's meta-ability, MetaEvo shifts the objective from direct answer optimization to learning how to revise. The framework integrates meta optimization with an agent system that extracts, stores, and reuses high-quality revision principles. Experimental results across multiple reasoning benchmarks demonstrate that MetaEvo consistently improves performance, supports iterative self-improvement, and enhances generalization. Notably, contrastive analysis proves essential for generating precise and actionable principles.

REFERENCES

Huan ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Qihan Ren, Cheng Qian, Zhenghailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A survey of self-evolving agents: On path to artificial super intelligence, 2025. URL https://arxiv.org/abs/2507.21046.

Guy Azov, Tatiana Pelc, Adi Fledel Alon, and Gila Kamhi. Self-improving customer review response generation based on llms, 2024. URL https://arxiv.org/abs/2405.03845.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL https://arxiv.org/abs/2005.14165.

Ding Chen, Shichao Song, Qingchen Yu, Zhiyu Li, Wenjin Wang, Feiyu Xiong, and Bo Tang. Grimoire is all you need for enhancing large language models. *CoRR*, abs/2401.03385, 2024. doi: 10.48550/ARXIV.2401.03385. URL https://doi.org/10.48550/arXiv.2401.03385.

Liting Chen, Lu Wang, Hang Dong, Yali Du, Jie Yan, Fangkai Yang, Shuang Li, Pu Zhao, Si Qin, Saravan Rajmohan, Qingwei Lin, and Dongmei Zhang. Introspective tips: Large language model for in-context decision making. *CoRR*, abs/2305.11598, 2023. doi: 10.48550/ARXIV.2305.11598. URL https://doi.org/10.48550/arXiv.2305.11598.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu,

Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

Jinglong Gao, Xiao Ding, Yiming Cui, Jianbai Zhao, Hepeng Wang, Ting Liu, and Bing Qin. Self-evolving GPT: A lifelong autonomous experiential learner. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 6385–6432. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.346. URL https://doi.org/10.18653/v1/2024.acl-long.346.

Ran Gong, Qiuyuan Huang, Xiaojian Ma, Yusuke Noda, Zane Durante, Zilong Zheng, Demetri Terzopoulos, Li Fei-Fei, Jianfeng Gao, and Hoi Vo. Mindagent: Emergent gaming interaction. In Kevin Duh, Helena Gómez-Adorno, and Steven Bethard (eds.), *Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pp. 3154–3183. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-NAACL.200. URL https://doi.org/10.18653/v1/2024.findings-naacl.200.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Sx038qxjek.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *CoRR*, abs/2009.03300, 2020. URL https://arxiv.org/abs/2009.03300.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. *CoRR*, abs/2103.03874, 2021. URL https://arxiv.org/abs/2103.03874.

Cheng Li, Jindong Wang, Yixuan Zhang, Kaijie Zhu, Wenxin Hou, Jianxun Lian, Fang Luo, Qiang Yang, and Xing Xie. Large language models understand and can be enhanced by emotional stimuli, 2023. URL https://arxiv.org/abs/2307.11760.

Xiaonan Li and Xipeng Qiu. Mot: Memory-of-thought enables chatgpt to self-improve. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 6354–6374. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.

EMNLP-MAIN.392. URL https://doi.org/10.18653/v1/2023.emnlp-main.392.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pp. 2080–2094. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.NAACL-MAIN.168. URL https://doi.org/10.18653/v1/2021.naacl-main.168.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David D. Cox, Yiming Yang, and Chuang Gan. Principle-driven self-alignment of language models from scratch with minimal human supervision. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/0764db1151b936aca59249e2c1386101-Abstract-Conference.html.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 13003–13051. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.824. URL https://doi.org/10.18653/v1/2023.findings-acl.824.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023. doi: 10.48550/ARXIV.2302.13971. URL https://doi.org/10.48550/arXiv.2302.13971.

Zidi Xiong, Yuping Lin, Wenya Xie, Pengfei He, Jiliang Tang, Himabindu Lakkaraju, and Zhen Xiang. How memory management impacts llm agents: An empirical study of experience-following behavior, 2025. URL https://arxiv.org/abs/2505.16067.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024. doi: 10.48550/ARXIV.2412.15115. URL https://doi.org/10.48550/arXiv.2412.15115.

Zeyuan Yang, Peng Li, and Yang Liu. Failures pave the way: Enhancing large language models through tuning-free rule accumulation. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 1751–1777. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.109. URL https://doi.org/10.18653/v1/2023.emnlp-main.109.

Ruihong Zeng, Jinyuan Fang, Siwei Liu, and Zaiqiao Meng. On the structural memory of llm agents, 2024. URL https://arxiv.org/abs/2412.15266.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: LLM agents are experiential learners. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 19632–19642. AAAI Press, 2024. doi: 10.1609/AAAI.V38I17.29936. URL https://doi.org/10.1609/aaai.v38i17.29936.

6 Appendix

LLM Usage Statement

In preparing this manuscript, a large language model (LLM) was employed solely to aid in polishing and improving the clarity, grammar, and readability of the text. No LLM was used to generate original scientific content, results, or analyses. All research ideas, experiments, data collection, and interpretations reported in this work were performed solely by the authors.

Prompt Design

Basic Prompt

```
<system>
You are an expert in solving problems. Please answer the question below. Let's think step by step.
</system>

{question}
```

Prompt for Guided Generation

```
<system>
You are an expert in solving problems. Please answer the question below. Let's think step by step.
</system>


Please ensure that your input adheres to the specified principles. Carefully follow the rules provided to complete the task accurately and efficiently.
```

<input>
question: {question}

principle: {rules}
</input>

Generate Task Description

<system>
Your task is to identify and extract the main task description from a given question.
</system>

<task description>
First, analyze the domain of the question, categorize it into a relevant subcategory, and then generate a concise, clear, and abstract task description that reflects the core objective.

Steps to Perform Structured Analysis:

- **Analyze the domain of the question**: Determine the field or category the question belongs to.
- **Categorize the task**: Identify the specific type of problem within that domain.
- **Generate the task description**: Based on the identified domain and subcategory, create a task description that is concise, clear, abstract, and focuses on the core objective. Avoid including unnecessary details or background information, and aim for a general formulation that reflects the essence of the task.

Output Format:
Show your analysis and provide your final response in the following JSON format:
"Task Description": "description": "Clear, abstract, and specific description of the task, focusing on the core action or objective." </task description>

<input>
Question: {question}
</input>

Memory Maintenance

Your goal is to compare the new_principle against each of the existing_principles, and decide one of the following for each:

1. **Redundant**: if the new and old principle express essentially the same idea. Prefer the newer one.
2. **Conflicting**: if the two principles provide contradictory guidance. Keep the one that is more general or correct.
3. **Complementary**: if the two principles provide distinct but compatible guidance. Keep both.
4. **Irrelevant**: if the existing principle is not applicable to the current task anymore. Suggest deletion.

Please return your evaluation in the following JSON format:
"comparisons": ["relation": "Redundant — Conflicting — Irrelevant"]
<input>
New_principle: {new_principle}
Existing_principle: {existing_principle}

Contrast Driven Extraction

<system>
You are an expert in analyzing and comparing task responses to identify *fine-grained*, *task-relevant*, and *impactful* differences between answers that affect quality.

<task description>
Given a high-quality and a low-quality answer to the same task, identify detailed differences that reflect meaningful changes in correctness, reasoning, completeness, or clarity.

Follow these steps:

- **Step 1: Understand the task type**
  Identify whether the task involves reasoning, generation, factual recall, explanation, etc. This will guide how you compare the answers.
- **Step 2: Perform a targeted comparison**
  Compare the answers component by component, such as sentence by sentence, step by step, or idea by idea—depending on the task structure.
- **Step 3: Identify key differences**
  For each meaningful difference:
  - Quote or paraphrase the *specific content* from both answers.
  - Indicate the **aspect** being affected.
  - Explain *why this difference matters*—how it affects the task's success, clarity, or correctness.

Important guidelines:

- Avoid vague language like "clearer" or "more logical" unless supported by concrete details.
- Specify missing steps, incorrect reasoning, unsupported claims, or structural flaws.
- Use task-specific language.

<output format="json">
"differences": ["Aspect": "Aspect being evaluated", "HighQuality": "Quoted or paraphrased content from the HQ answer that shows good performance", "LowQuality": "Quoted or paraphrased content from the LQ answer that shows the issue", "Differences": "Detailed explanation of why this difference affects answer quality, referencing task goals or logical consequences"]
</output format>

<input>
Question: {input}
Low-quality Answer: {predict}
High-quality Answer: {reference}

Principle Generation

<system>
You are a prompt engineering expert skilled in deriving precise and generalizable principles that improve language model outputs. Your task is to formulate principles based on observed differences between high- and low-quality answers, ensuring each principle reflects a specific failure pattern and offers guidance for correction.

<task description>
Your task is to generate reusable and insightful improvement principles based on observed differences between two answers.

Follow these steps:

- **Step 1:** Carefully examine each identified difference and explain how it impacts the answer quality.
- **Step 2:** For each difference, derive a principle that captures the core insight and helps guide future answer generation.
- **Step 3:** Ensure each principle is general enough to be reused across similar tasks, yet clearly grounded in the specific difference observed.
- **Step 4:** Respond strictly in the following JSON format, where each principle includes a concise description and a short explanation of how to apply it.

<input>
Input:
Question: {input}
Difference: {difference}

<output format="json"> "'json "output": ["Principle": "State a clear and generalizable insight derived from the difference."]