SELF: Self-Evolution with Language Feedback

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have shown impressive adaptability in various fields, yet the optimal pathway of autonomous model evolution remains under-explored. Drawing inspiration from the self-driven learning process of humans, we introduce *SELF* (Self-Evolution with Language Feedback), a novel learning framework that empowers LLMs to continually self-improve their abilities. SELF initiates with a meta-skill learning process that equips the LLMs with capabilities for self-feedback and self-refinement. SELF employs language-based feedback for detailed and nuanced evaluations, pinpointing response flaws and suggesting refinements. Subsequently, the model engages in an iterative process of self-evolution: they autonomously generate responses to unlabeled instructions, refine these responses interactively, and use the refined and filtered data for iterative self-training, thereby progressively boosting their capabilities. Moreover, the SELF framework equips the model with the ability to self-refine during inference, leading to further improved response quality. Our experiments on mathematical and general tasks demonstrate that SELF enables the model to continually self-improve without human intervention. The SELF framework indicates a promising direction for the autonomous evolution of LLMs, transitioning them from passive information receivers to active participants in their development.

1 Introduction

Large Language Models (LLMs), like ChatGPT (OpenAI, 2022) and GPT-4 (OpenAI, 2023) , stand at the forefront of the AI revolution, demonstrating versatility across tasks. Despite their evident capabilities, the way towards achieving autonomous development of LLMs is still under-explored.

The development of automatically improved LLMs can draw inspiration from human self-driven learning mechanisms. When facing new challenges, humans naturally engage in a learning cycle of initial attempts, introspective feedback, and behavior refinement. This leads to a critical question: "Can LLMs mimic the human learning process, utilizing self-refinement to enhance their inherent capabilities?" Fascinatingly, a recent study (Ye et al., 2023) in top-tier LLMs such as GPT-4 has revealed emergent meta-skills for self-refinement, signaling a promising future direction for the self-evolution of LLMs. Despite this, current methods for LLM development typically rely on a single round of instruction fine-tuning (Wei et al., 2021; Zhou et al., 2023) with meticulously human-crafted datasets and reinforcement learning-based methods (Ouyang et al., 2022) that depend on an external reward model. These strategies not only require extensive resources and ongoing human intervention but also treat LLMs as mere passive repositories of information rather than active learners. These limitations hinder LLMs from tapping into their inherent capabilities, obstructing their progress toward a self-driven, autonomous learning paradigm. Thus, we introduce *SELF* (Self-Evolution with Language Feedback) framework, designed to unlock the potential for autonomous self-evolution in LLMs. Fig-

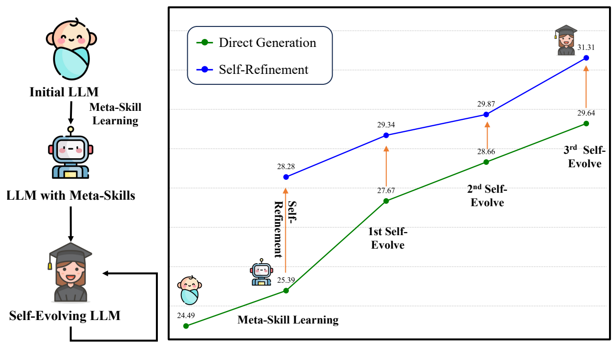


Figure 1: Evolutionary Journey of SELF: An initial LLM undergoes successive self-evolution iterations (1st, 2nd, 3rd), enhancing its capabilities and acquiring a self-refinement meta-skill.

ure 1 depicts how SELF mimics human-like self-driven learning, emphasizing progressive improvement of model capability with self-evolution training. At the core of SELF are the two meta-skills (*self-feedback and self-refinement*), empowering the model to progressively self-evolve by training on its own synthesized data. Additionally, SELF leverages self-generated natural language feedback to offer in-depth analysis and guidance for refining responses, without the need for external rewards or direct human guidance.

Specifically, the SELF framework initiates by teaching LLMs essential meta-skills, namely self-feedback and self-refinement, using a limited set of examples. Once these skills are acquired, the model engages in a cycle of continuous self-evolution, iteratively training with extensive, self-generated data. Given a large-scale unlabeled corpus, this data is compiled from initial responses and refined through self-refinement and filtering, with model itself. During this iterative process, the quality of self-evolution training data and model capability are interactively improved, fostering ongoing self-evolution of LLMs. Crucially, in the inference phase, these learned meta-skills enable LLMs to further enhance response quality via self-refinement. In summary, the SELF framework transforms LLMs from passive recipients of data into active learners in self-evolution and alleviates data scarcity issues by generating large-scale self-curated training datasets. This not only reduces the need for labor-intensive manual intervention but also promotes the continuous self-improvement of LLMs, establishing a more autonomous and efficient training approach.

We evaluate SELF in mathematical and general domains. SELF notably improves the test accuracy on mathematical domains (6.82% on GSM8k (Cobbe et al., 2021) and 4.9% on SVAMP (Patel et al., 2021)), and increases the win rate on general domain (10% on Vicuna testset (Lianmin et al., 2023) and 6.9% on Evol-Instruct testset (Xu et al., 2023)), compared with typical supervised fine-tuning. The main contributions are summarized as follows: (1) SELF empowers LLMs with self-evolving capabilities, allowing for autonomous model evolution, and reducing human intervention. (2) SELF facilitates self-refinement into smaller LLMs, even with challenging math problems. The capability of self-refinement was previously considered an emergent characteristic of top-tier larger LLMs. (3) Experiments demonstrate the effectiveness of SELF in both mathematical and general domains, confirming its advanced capabilities in self-evolution and self-refinement.

2 Related Works

Self-improvement Methods A straightforward and effective method to improve large language models (LLMs) for reasoning tasks is self-consistency (Wang et al., 2022a). This involves sampling various reasoning paths and selecting the most consistent answer. Various research efforts have been undertaken to enhance the output quality of LLMs through online self-improvement (Shinn et al., 2023; Madaan et al., 2023; Ye et al., 2023; Chen et al., 2023; Ling et al., 2023). The main idea is to generate an initial output with an LLM, have the same LLM provide feedback on its output, and then use this feedback to refine the initial output. Some works focus on self-improvement over prompts (Fernando et al., 2023; Zhang et al., 2023). While simple and effective, online self-improvement requires multi-turn inference for refinement, leading to increased computational overhead. Therefore, other methods explore self-improvement during fine-tuning. These methods aim to iteratively enhance the LLM's performance by leveraging both ground truth and synthetic data it generates (Yuan et al., 2024; Chen et al., 2024; Gou et al., 2023; Wang et al., 2023; Li et al., 2023). Our SELF, autonomously enhances its capabilities without reliance on ground-truth data via self-refinement, providing detailed language feedback.

Autonomous Improvements of LLMs "Alignment" (Leike et al., 2018) aims to train agents to act in line with human intentions. Several research efforts (Ouyang et al., 2022; Bai et al., 2022a; Scheurer et al., 2023) leverage Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017). RLHF begins with fitting a reward model to approximate human preferences. Subsequently, an LLM is finetuned through reinforcement learning to maximize the estimated human preference of the reward model. Reward Ranked Fine-tuning (RAFT) utilizes a reward model to rank responses sampled from an LLM. Subsequently, it fine-tunes the LLM using highly-ranked responses (Dong et al., 2023). Recent advancements in LLMs have explored Reinforcement Learning (RL) approaches that do not rely on human feedback. RLAIF (Pang et al., 2023) proposes to employ a LLMs to label the preference data in replace of human supervision. LLMs are updated progressively through online RL in interacting with the environment in Carta et al. (2023). The connection between conventional RL research and RLHF in LLMs is discussed by Sun (2023). However, scalar rewards in Reinforcement Learning (RL) offer limited insights for evaluating complex reasoning tasks (Lightman et al., 2023), as they fail to specify detailed errors and optimization paths. Recognizing this limitation, the SELF framework suggests utilizing natural language feedback, which effectively guides the self-evolution of LLMs. Unlike scalar rewards, which require a retrained model for each evaluation protocol and dimension, natural language feedback is more flexible, addressing multiple aspects simultaneously.

3 Method

As depicted in Fig. 2, the SELF framework enhances model capabilities through a two-stage learning phase: (1) **Meta-skill Learning Phase**: This phase uses an annotated meta-skill training corpus to fine-tune the model, and equips the model with essential meta-skills for self-feedback and self-refinement with limited supervised examples. (2) **Self-Evolution Phase**: With the acquired meta-skills, the model progressively improves through multiple iterations of the self-evolution training process. The whole process is illustrated in Alg. 1 in Appendix I.

3.1 Meta-Skill Learning

Meta-skill learning targets on instill two essential meta-skills into LLMs for self-evolution. (1) **Self-Feedback Ability**: This skill enables LLMs to evaluate their responses using natural language feedback. This provides the suggestion for further refinement, thus laying a solid foundation for subsequent self-refinement. Self-feedback also enables the model to filter out low-quality self-evolution training data if a response is judged as unqualified by the model (section 3.2.1). (2) **Self-Refinement Ability**: Self-refinement enables the model to optimize its responses based on self-feedback. This ability has two applications: (1) enhancing the quality of the self-evolution training corpus (section 3.2.1) and (2) improving model performance by refining the models' outputs during inference (section 3.3).

These meta-skills are acquired by fine-tuning the model using the **Meta-Skill Training Corpus** (section 3.1.1) with designed training objective (section 3.1.2). The resulting model is denoted as M_{meta}.

3.1.1 Meta-Skill Training Corpus

The meta-skill training corpus D_{meta} represents the generation, feedback, and refinement process. It is constructed as follows: (1) For each unlabeled prompt p, the initial model M_{init} generates an initial response r. (2) An annotator L provides evaluation feedback f for the initial response r, then produces a refined answer $\hat{r}$ according to the feedback f. Each instance in D_{meta} takes the form $(p, r, f, \hat{r})$, representing the process of response evaluation and refinement. An example instance of D_{meta} is provided in appendix H.

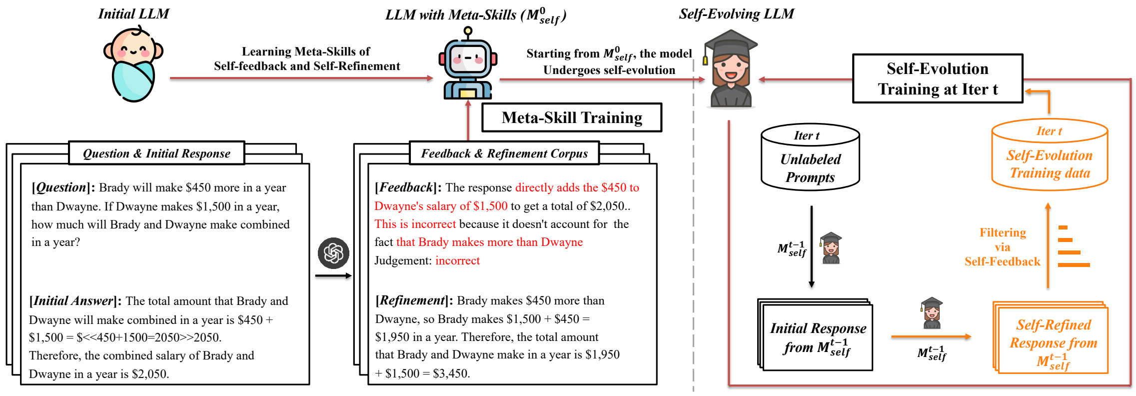
3.1.2 Training Objective

In the meta-skill learning phase, the objective is to empower the initial model M_{init} to develop meta-skills, resulting in an enhanced model M_{meta}. This process is guided by the cross-entropy loss following the maximum likelihood estimation (MLE) paradigm:

$$\mathcal{L}_{\text{meta}}(\phi) = -\mathbb{E}_{(p,r,f,\hat{r})\sim D_{\text{meta}}} \Big[\log \tau_\phi(f|p,r) + \log \tau_\phi(\hat{r}|p,r,f) + \beta \log \tau_\phi(\hat{r}|p)\Big], \tag{1}$$

where p is prompt, r is the initial model response, f is the feedback to the model response r, and $\hat{r}$ is the revised response based on feedback f. $\tau_\phi(y|x)$ denotes the probability distribution given by the auto-regressive language model with parameters ϕ predicting the response y given the input prompt x. The coefficient β in eq. (1) controls a balanced emphasis on direct response generation and the model's capability for self-feedback and self-refinement.

Insight. Training with D_{meta} aims to achieve two goals: (i) To guide the model in generating feedback (f) concerning its initial responses (r) (**self-feedback**) and subsequently employing this feedback to enhance the quality of the final answer ($\hat{r}$) (**self-refinement**). (ii) Training with D_{meta} instructs the model to process problems in a Chain-of-Thought (CoT) manner. This involves evaluating the initial response, integrating the feedback, and then revising the response in a chain process $\Psi(\hat{r}|p) := \sum_{r,f} \tau_\phi(r|p) \cdot \tau_\phi(f|p,r) \cdot \tau_\phi(\hat{r}|p,r,f)$.



(a) Meta-Skill Learning

(b) Iterative Self-Evolution Training

Figure 2: Illustration of SELF. The "Meta-Skill Learning" (left) phase empowers the LLM to acquire meta-skills in self-feedback and self-refinement. The (b)"Self-Evolution" phase (right) utilizes meta-skills for self-evolution training with self-curated data, enabling continuous model enhancement.

3.2 Self-Evolution Training Process

The model M_{meta}, equipped with meta-skills, undergoes progressive improvement through multiple iterations of the self-evolution training process. Each iteration of the self-evolution process begins with the model autonomously creating high-quality training data (section 3.2.1) from an unlabeled corpus. With an unlabeled dataset of prompts, the model generates initial responses and then refines them through self-feedback and self-refinement. These refined responses, superior in quality, are further filtered with self-feedback and utilized as the training data for the model's subsequent self-evolution training (section 3.2.2). This autonomous self-evolving process interactively improves LLMs as the improved model capability leads to better data quality, which in turn boosts model performance. It also alleviates the data scarcity problem by self-generating data.

3.2.1 Self-Evolution Training Data

Let M^t_{evol} denotes the model at t^{th} iteration and initialize M^0_{evol} from M_{meta}. During t^{th} self-evolution iteration , M^{t-1}_{evol} processes each unlabeled prompt p by first generating an initial response r. r is then refined using the model's self-feedback f, resulting in a self-refined response $\hat{r}$. The prompts and their corresponding self-refined responses$(p, \hat{r})$ are then incorporated into the t^{th} round self-evolution datasets, denoted as D^t_{evol}, for subsequent self-evolution processes.

Data Filtering with Self-feedback: To enhance the quality of D^t_{evol}, we employ the self-feedback capability of M^{t-1}_{evol} to filter out data of lower quality. M^{t-1}_{evol} evaluates the self-refined data, $\hat{r}_{\text{evol}}$, keeping only the responses that meet high-quality standards. The effect is analyzed in appendix Q.

To mitigate the catastrophic forgetting issue of meta-skill, the meta-skill learning data D_{meta} are also included in self-evolution training. At t^{th} iteration, the model undergoes self-evolution training with the updated self-curated data D^t_{evol}, improving its performance and aligning it more closely with human values.

3.2.2 Mathematical Modeling

Main Objective. We denote τ^t_ϕ as the probability distribution generated by M^t_{evol} with parameters ϕ. The self-evolution training loss $\mathcal{L}^t_{\text{evol}}(\phi)$ is defined as:

$$\begin{aligned}
&\mathcal{L}^t_{\text{evol}}(\phi) \\
&= -\mathbb{E}_{p_{\text{evol}}}\mathbb{E}_{\hat{r}_{\text{evol}}\sim\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})}\left[\log \tau^t_\phi(\hat{r}_{\text{evol}}|p_{\text{evol}})\right] \\
&= -\mathbb{E}_{p_{\text{evol}}}\left[\sum_{\hat{r}_{\text{evol}}}\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})\log \tau^t_\phi(\hat{r}_{\text{evol}}|p_{\text{evol}})\right],
\end{aligned} \tag{2}$$

where p_{evol} is sampled from unlabeled prompts corpus (detiled in appendix C.2) for self-evolution t^{th} round. The joint probability distribution is:

$$\begin{aligned}
&\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}}) = \\
&\sum_{r_{\text{evol}},f_{\text{evol}}}\Big(\tau^{t-1}_\phi(r_{\text{evol}}|p_{\text{evol}})\cdot\tau^{t-1}_\phi(f_{\text{evol}}|r_{\text{evol}},p_{\text{evol}}) \\
&\cdot\tau^{t-1}_\phi(\hat{r}_{\text{evol}}|f_{\text{evol}}.r_{\text{evol}},p_{\text{evol}})\Big).
\end{aligned} \tag{3}$$

The rationale behind learning from $\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})$ is discussed in appendix A.1.

Optimizing eq. (2) is equivalent to minimizing the Kullback-Leibler (KL) divergence:

$$
\begin{aligned}
&\mathrm{KL}(\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})||\tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}})) \\
&= \sum_{\hat{r}_{\text{evol}}} \Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}}) \log \frac{\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})}{\tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}})} \\
&= -\underbrace{H(\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}}))}_{\text{constant for fixed } \Psi^{t-1}} - \\
&\underbrace{\sum_{\hat{r}_{\text{evol}}} \Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}}) \log \tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}})}_{\text{Eq. (2)}}.
\end{aligned}
\tag{4}
$$

The optimization of KL divergence is to fine-tune the model parameters ϕ to ensure that the model's predictive probability distribution τ_{ϕ}^{t} aligns with the joint probability of the preceding iteration's chain process (Ψ^{t-1}). The goal is to enhance the model's ability to directly produce refined responses ($\hat{r}_{\text{evol}}$) in the t^{th} iteration, effectively condensing the intricate process of generation, feedback, and refinement from the $(t-1)^{th}$ iteration. This advancement demonstrates the model's evolving capability to streamline the complex steps into a more straightforward inference. The potential plateau is discussed in appendix A.3.

Further Analysis. Assuming that each self-evolution round is effective, implying that as t increases, the quality of responses sampled from Ψ^{t} improves, optimizing the KL divergence as described in eq. (4) is fundamentally a process aimed at enhancing the direct generation of high-quality responses. Before delving deeper, it is crucial to introduce several key concepts. We define a binary variable X to evaluate the quality of responses. A higher probability, $p(X = 1 \mid p_{\text{evol}}, \hat{r}_{\text{evol}})$, indicates a higher quality of the response $\hat{r}_{\text{evol}}$ in relation to the prompt p_{evol}. For the self-evolving model with parameters ϕ at the t^{th} iteration, the model's log-likelihood of producing high-quality responses to a specified prompt is defined as follows:

$$
\begin{aligned}
&\log p^{t}(X = 1 \mid p_{\text{evol}}) \\
&:= \log \sum_{\hat{r}} p(X = 1 \mid p_{\text{evol}}, \hat{r}_{\text{evol}}) \tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}}).
\end{aligned}
$$

By minimizing the KL divergence in eq. (4), we can increase $\log p^{t}(X = 1 \mid p_{\text{evol}})$ by progressively improving its Evidence Lower Bound (ELBO):

$$
\begin{aligned}
&\log p^{t}(X = 1 \mid p_{\text{evol}}) \\
&= \log \sum_{\hat{r}_{\text{evol}}} p(X = 1 \mid p_{\text{evol}}, \hat{r}_{\text{evol}}) \tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}}). \\
&= \log \mathbb{E}_{\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})} \left[\frac{p(X = 1 \mid p_{\text{evol}}, \hat{r}_{\text{evol}}) \tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}})}{\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})} \right] \\
&\geq \mathbb{E}_{\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})} \left[\log \frac{p(X = 1 \mid p_{\text{evol}}, \hat{r}_{\text{evol}}) \tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}})}{\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})} \right] \\
&= \mathbb{E}_{\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})} \left[\log p(X = 1 \mid p_{\text{evol}}, \hat{r}_{\text{evol}}) \right] \\
&\quad - \underbrace{\mathrm{KL}(\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})||\tau_{\phi}^{t}(\hat{r}_{\text{evol}}|p_{\text{evol}}))}_{\text{Eq. (4)}}.
\end{aligned}
$$

The entire self-evolution training process can be viewed as a continuous exploration of inherent model capabilities in generation, self-feedback, and self-refinement, ultimately enhancing the model's ability to generate high-quality responses directly.

Overall Objective. In the iterative self-evolution process, meta-skills, i.e., the ability to self-feedback and self-refinement, is crucial for guiding the evolution process. We incorporate D_{meta} into self-evolution training to mitigate the potential catastrophic forgetting of meta-skills:

$$
\begin{aligned}
&\mathcal{L}_{\text{meta}}^{t}(\phi) = -\mathbb{E}_{(p,r,f,\hat{r}) \sim D_{\text{meta}}} \\
&\left[\log \tau_{\phi}^{t}(f|p, r) + \log \tau_{\phi}^{t}(\hat{r}|p, r, f) \right].
\end{aligned}
$$

The total objective for the t^{th} round of self-evolution is:

$$
\mathcal{L}_{\text{self}}^{t}(\phi) = \mathcal{L}_{\text{evol}}^{t}(\phi) + \mathcal{L}_{\text{meta}}^{t}(\phi).
$$

3.3 Response Refinement during Inference

Equipped with the meta-skills for self-feedback and self-refinement, the model can conduct self-refinement during inference. Specifically, the model generates an initial response and then refines it using self-refinement, akin to the method described in section 3.1. Response refinement during inference consistently improves the model's performance as shown in section 4.2.

4 Experiments

This section begins with an introduction to the experimental settings (section 4.1), encompassing the evaluation data, baseline model, and model variations. The following experiments are exhibited: (1) We demonstrate the efficacy of SELF compared to baselines in the main experiment (Section 4.2). (2) We show progressive performance enhancements observed during the self-evolution processes in the ablation study (Section 4.3). (3) Comparison with other self-improvement methods (Section 4.4)

The additional experiments in the Appendix provide comprehensive insights into our SELF framework. (3) appendix Q shows the impact of data filtering with self-feedback in self-evolution training data construction. (4) appendix K evaluates different meta-skill training data organization methods, highlighting the effectiveness of single-response refinement over multiple-response. (5) appendix L analyzes various self-evolution training strategies, emphasizing the superiority of "Restart Training". (6) appendix M demonstrates that SELF outperforms supervised fine-tuning (SFT) with human-annotated data. (7) appendix N assesses the scalability of SELF across varying base models, indicating its increased effectiveness with more advanced models. (8) appendix O exhibits that the quality of the meta-skill data influences the self-evolution process, with improvements observed when using higher-quality data. (9) appendix P conducts the comparison between single-round and iterative self-evolution training and reveals the advantages of the iterative approach in improving LLMs' capabilities over successive rounds.

4.1 Experiment Settings

4.1.1 Evaluation

Inference Setting. We adopt two inference settings: (1) **Direct Response** (default): the model directly answers the question with a Zero-shot Chain of Thought (CoT) methodology (Kojima et al., 2022), which is the default setting to evaluate the model capability directly; (2) **Self-Refinement**: during inference, the model self-refines its original answer for once, as described in section 3.3.

Benchmarks. We evaluate two mathematical benchmarks to show the efficacy of SELF on complex reasoning tasks and further verify the generalizability of SELF on seven general benchmarks. Please refer to Appendix F for more details about these benchmarks.

4.1.2 Setup and Baselines

The **complete SELF framework** includes meta-skill training with D_{meta}, three iterations of self-evolution training, and optional self-refinement during inference. Our evaluation primarily focuses on assessing how self-evolution training can progressively enhance the capabilities of LLMs. For building the meta-skill training corpus D_{meta}, we employ GPT-4 as the language model labeler L due to its proven proficiency in refining responses (An et al., 2023) via the prompt described in appendix B[1]. The data statistic of D_{meta} is shown in appendix C.1 and further details of unlabeled corpus construction is described in appendix C.2. We note that all model training utilized the same training hyperparameters, as shown in appendix D. In this study, we experiment with Vicuna-7b (**Vicuna**) (Chiang et al., 2023). All other compared baselines are outlined. For more details about these baselines, please refer to Appendix G:

(1) Vicuna + D_{QA}: we construct D_{QA}, which includes all the $(p, \hat{r})$ pairs from D_{meta}, and fine-tune the model as:

$$\mathcal{L}_{\text{QA}}(\phi) = -\mathbb{E}_{(p,\hat{r})\sim D_{\text{QA}}}\left[\log \tau_\phi(\hat{r}|p)\right].$$

(2) RLHF: we utilize the RLHF implementation from trlx[2].

(3) Self-Consistency: we compare the self-refinement inference strategy in SELF with the Self-Consistency (Wang et al., 2022a).

4.2 Main Result

4.2.1 Math Test

Model	SE	SC	SR	GSM8K(%)	SVAMP(%)
Vicuna				16.43	36.40
		✓		19.56	40.20
			✓	15.63	36.80
Vicuna + D_{QA}				24.49	44.90
		✓		25.70	46.00
			✓	24.44	45.30
Vicuna + SELF (Ours)	✓			29.64	49.40
	✓	✓		29.87	50.20
	✓		✓	31.31	49.80
	✓	✓	✓	**32.22**	**51.20**

Table 1: Experiment results on GSM8K and SVAMP compare SELF with other baseline methods. We evaluate the impact of Self-Evolution (SE), Self-Consistency (SC), and Self-Refinement (SR) strategies on model performance.

In section 4.2.1, we compare SELF against baseline models, as detailed in section 4.1.2. This comparison elucidates SELF's effectiveness in enhancing LLM performance through self-evolution and offers several key insights:

(1) Self-Evolution Enhances LLM: Vicuna + SELF significantly outperforms its baseline Vicuna + D_{QA} ($24.49\% \xrightarrow{+5.15\%} 29.64\%$ on GSM8K and

[1]Separate prompts have been designed for the math domain appendix B.1 and general domain appendix B.2.

[2]https://github.com/CarperAI/trlx

44.90% $\xrightarrow{+4.5\%}$ 49.40% on SVAMP) in direct response setting, showcasing self-evolution is effective in optimizing LLMs.

(2) SELF Instills Self-Refine Capability in LLMs: The integration of self-refinement inference strategy with Vicuna + SELF further boosts performance (29.64% $\xrightarrow{+1.67\%}$ 31.31%), while baseline models show marginal or negative effect via self-refinement. We also provide a case analysis for the limited self-refinement ability of baseline models, as shown in fig. 4.

(3) SELF can work with Self-Consistency: SELF works effectively with self-consistency, improving accuracy across models. The base Vicuna model, which may have uncertainties in its outputs, shows notable improvement with self-consistency, achieving a +3.13% increase. Combining self-refinement with self-consistency further elevates performance (e.g., 29.64% $\xrightarrow{+2.58\%}$ 32.22% on GSM8K), indicating that these two strategies can complement each other effectively.

4.2.2 Comparison with RLHF

Method	Feedback Acc.(%)	GSM8K Acc.(%)
Vicuna + D_{QA}	-	24.49
RLHF	24	25.55
SELF	**72**	**27.67**

Table 2: Comparison of SELF and RLHF on GSM8K. "Feedback Acc." measures how accurately feedback identifies correct and incorrect answers, while "GSM8K Acc." shows the model performance on GSM8K testset.

In section 4.2.2, we compare the performance of SELF with RLHF. To alleviate the effect led by different amounts of training data and make a fair comparison, for SELF, we adopt data solely from the initial round of self-evolution training. This ensures the same training data quantity with RLHF and leads to sub-optimal results compared with the one in section 4.2.1.

As section 4.2.2 shows, RLHF achieves a 25.55% accuracy on GSM8K, which is lower than the 27.67% performed by SELF. We observe that the simple scalar reward of RLHF often fails to identify the correctness of the reasoning process, which limits performance improvements. On the GSM8K test set, for incorrect answers produced by the SFT model (Vicuna + D_{QA}), the reward model only identifies 24% of them as incorrect, i.e., the reward model assigns lower scalar rewards to incorrect answers compared to correct answers. In contrast, SELF utilizes informative natural language feedback to provide a more accurate assessment. It correctly identifies 72% of incorrect answers.

4.2.3 General Test

To demonstrate the generalizability of the SELF framework across a wider range of datasets and tasks, we conducted following experiments for comparing three configurations of the Vicuna model, i.e., Vicuna, Vicuna + D_{QA}, and Vicuna + SELF with details in Appendix R.

Five Open LLM Leaderboard datasets This experiment evaluates the SELF model, trained for general domains on five datasets. The results of these experiments are summarized in Table 3:

Datasets	Vicuna	Vicuna + D_{QA}	Vicuna + D_{QA} + SELF
Arc	71.34	72.54	73.71
TruthfulQA	50.36	51.17	52.36
Winogrande	69.38	68.12	67.17
HellaSwag	73.80	75.01	76.24
MMLU	48.60	48.71	49.17
Average	62.70	63.11	63.73

Table 3: Results on five open LLM leaderboard datasets.

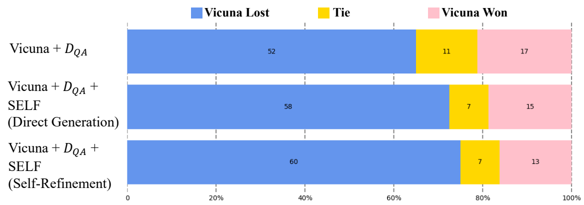
The overall average performance of the SELF framework showed improvement over its baseline.

Vicuna and Evol-instruct Test Evaluations We also test the efficacy and generalizability of SELF on two general domain benchmarks, explicitly using the Vicuna and Evol-Instruct test sets.

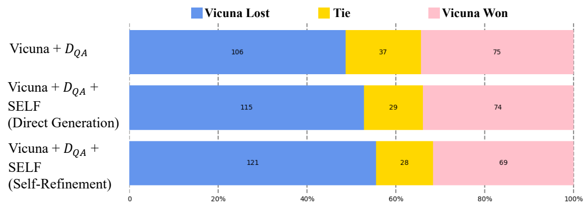
The results are depicted in Figure 3. In the figure, blue represents the number of test cases where the model being evaluated is preferred over the baseline model (Vicuna), as assessed by GPT-4. Yellow denotes test cases where both models perform equally, and pink indicates the number of test cases where the baseline model is favored over the model being evaluated.

In the Vicuna testset, SELF increases direct response win rate from 65.0% to 72.5% compared with Vicuna + D_{QA}. After self-refinement, the win rate is further improved to 75.0%. In the Evol-Instruct testset, the win rate of Vicuna + D_{QA} is 48.6%. SELF increases the win rate to approximately 52.8%. Applying self-refinement during inference further improves the win rate to 55.5%.

These findings in the general domain highlight the SELF framework's adaptability and robustness,



(a) Results on Vicuna testset.



(b) Results on Evol-Instruct testset.

Figure 3: Results on Vicuna testset and Evol-Instruct testset

particularly when self-refinement is employed, showcasing its efficacy across varied test domains.

4.3 Ablation Study

SVAMP (%)		GSM8K (%)		D_{QA}	D_{meta}	Self Evol.		
DR	SR	DR	SR			1st	2nd	3rd
36.4	36.8	16.43	15.63					
44.9	45.3	24.49	24.44	✓				
46.8	47.0	25.39	28.28		✓			
47.8	48.0	27.67	29.34		✓	✓		
48.9	49.0	28.66	29.87		✓	✓	✓	
49.4	**50.2**	**29.64**	**31.31**		✓	✓	✓	✓

Table 4: Performance under various training settings of SELF. A checkmark ✓ in a column denotes the additive adoption of the corresponding setting in that training scenario. We present two kinds of inference results: **Direct Response** (DR) and **Self-Refinement** (SR), the latter conducts self-refinement to DR.

We conduct ablation experiments on SVAMP and GSM8K datasets to assess the incremental effect of each stage. While baseline models exhibit slight or even adverse effects via self-refinement, the SELF framework endows LLMs with an inherent capability through meta-skill learning and multi-iterations of self-evolution training. As depicted in table 4, our framework facilitates gradual performance improvements through successive SELF stages. More detailed observations are highlighted in Appendix T:

4.4 Comparison with self-improvement methods

We provide additional experiments comparing our SELF method with two self-improvement works, i.e., SPIN (Chen et al., 2024) and Self-rewarding (Self-RW) (Yuan et al., 2024). We compared fairly by reimplementing each method based on the Mistral-7B (Jiang et al., 2023) post-meta-skill learning (Base). We report the results in the GSM8K dataset.

Model	SELF	Self-RW	SPIN
Base	51.10	51.10	51.10
Iter 1	52.23 ± 0.15	52.15 ± 0.10	52.24 ± 0.18
Iter 2	52.41 ± 0.10	52.45 ± 0.12	52.44 ± 0.20
Iter 3	53.51 ± 0.18	52.37 ± 0.17	52.44 ± 0.14

Table 5: Comparison of accuracy on the GSM8K dataset over 3 self-improvement iterations.

Unlike SPIN and Self-RW, which use Direct Preference Optimization loss, our SELF framework, utilizing standard supervised fine-tuning loss, achieves higher accuracy on the GSM8K dataset after three self-improvement iterations. As demonstrated in Table 5, our SELF framework is efficient and effective during iterative self-improvement training. The small standard deviation further highlights the reliability of our results.

5 Conclusion

We present SELF (Self-Evolution with Language Feedback), an innovative framework that enables LLMs to undergo self-evolution via self-feedback and self-refinement. SELF transforms LLMs from passive information recipients to active participants in their evolution. It utilizes natural language feedback for detailed and informative evaluations Through meta-skill learning, SELF equips LLMs with the capability for self-feedback and self-refinement. This allows models to autonomously enhance their abilities through self-evolution training and online refinement. Experiments conducted on benchmarks underscore SELF's capacity to progressively enhance model capabilities while reducing the need for human intervention. SELF represents a leading step in the autonomous development of LLMs, showcasing their potential for continuous learning and self-evolution.

6 Limitations

As the self-evolution process progresses through multiple iterations, there is a possibility that performance improvements may plateau. This phenomenon could occur due to several factors, such as the model reaching its inherent capacity limits or the diminishing returns from additional rounds of self-evolution. We add a discussion in Appendix A.3. Moreover, although the use of natural language feedback in the SELF framework enhances the evaluation and refinement process, it introduces a dependency on the accuracy and relevance of the feedback provided. Ensuring that the language feedback precisely pinpoints true information is critical.

References

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. 2023. Learning from mistakes makes llm better reasoner. *arXiv preprint arXiv:2310.20689*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosiute, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemí Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022b. Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. 2023. Grounding large language models in interactive environments with online reinforcement learning. *arXiv preprint arXiv:2302.02662*.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. *CoRR*, abs/2401.01335.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

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. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *CoRR*, abs/2305.14325.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. 2023. Promptbreeder: Self-referential self-improvement via prompt evolution. *CoRR*, abs/2309.16797.

Xinyang Geng and Hao Liu. 2023. Openllama: An open reproduction of llama.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. CRITIC: large language models can self-correct with tool-interactive critiquing. *CoRR*, abs/2305.11738.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *CoRR*, abs/2205.11916.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. 2018. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023. Self-alignment with instruction backtranslation. *arXiv preprint arXiv:2308.06259*.

Zheng Lianmin, Chiang Wei-Lin, and Zhuang Siyuan (Ryans). 2023. Vicuna-blog-eval. `https://github.com/lm-sys/vicuna-blog-eval`.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3214–3252. Association for Computational Linguistics.

Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. 2023. Deductive verification of chain-of-thought reasoning. *arXiv preprint arXiv:2306.03872*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984.

OpenAI. 2022. Chatgpt. `https://chat.openai.com/chat`.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Jing-Cheng Pang, Pengyuan Wang, Kaiyuan Li, Xiong-Hui Chen, Jiacheng Xu, Zongzhang Zhang, and Yang Yu. 2023. Language model self-improvement by reinforcement learning contemplation. *arXiv preprint arXiv:2305.14483*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press.

Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. 2023. Training language models with language feedback at scale. *arXiv preprint arXiv:2303.16755*.

Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Preprint*, arXiv:2303.11366.

Hao Sun. 2023. Reinforcement learning in the era of llms: What is essential? what is needed? an rl perspective on rlhf, prompting, and beyond. *arXiv preprint arXiv:2310.06147*.

Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michaël Mathieu, Nat McAleese, Nathalie Bradley-Schmieg, Nathaniel Wong, Nicolas Porcel, Roberta Raileanu, Steph Hughes-Fitt, Valentin Dalibard, and Wojciech Marian Czarnecki. 2021. Open-ended learning leads to generally capable agents. *CoRR*, abs/2107.12808.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022a. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022b. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13484–13508. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Seonghyeon Ye, Yongrae Jo, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, and Minjoon Seo. 2023. Selfee: Iterative self-revising llm empowered by self-feedback generation. Blog post.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *CoRR*, abs/2401.10020.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4791–4800. Association for Computational Linguistics.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

A Discussion

A.1 Why Refinement is Better

We discuss why it's necessary to optimize $\tau_\phi^t(\hat{r}_{\text{evol}}|p_{\text{evol}})$ in the t^{th} round self-evolution by learning from $\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})$, and why we believe samples from $\Psi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}})$ are typically of higher quality than those from $\tau_\phi^{t-1}(r_{\text{evol}}|p_{\text{evol}})$ directly.

Firstly, similar to the insights analyzed in section 3.1.2, we believe that a process akin to CoT, involving feedback followed by refinement before providing an answer, helps in generating high-quality responses. Secondly, r_{evol} is already a reasonably good output after meta-skill learning and previously $(t-1)$ rounds of self-evolution. We can assume that the self-feedback f_{evol} is informative, hence $\hat{r}_{\text{evol}} \sim \tau_\phi^{t-1}(\hat{r}_{\text{evol}}|p_{\text{evol}}, r_{\text{evol}}, f_{\text{evol}})$ is of higher quality than $r_{\text{evol}} \sim \tau_\phi^{t-1}(r_{\text{evol}}|p_{\text{evol}})$ because it incorporates useful feedback information. If f_{evol} suggests that the initial response r_{evol} does not require refinement, we still proceed through the process of revising from r_{evol} to $\hat{r}_{\text{evol}}$ using f_{evol}, but set $\hat{r}_{\text{evol}} = r_{\text{evol}}$. By doing so, we ensure that the quality of $\hat{r}_{\text{evol}}$ is at least as good as that of r_{evol}.

Moreover, as described in section 3.2.2, we utilize **Data Filtering with Self-feedback**. In other words, we only keep $\hat{r}_{\text{evol}}$ evaluated as *qualified*, allowing us to emphasize high-quality outputs and further improve τ_ϕ^t.

A.2 Why Integration of Meta-skill Training Data D_{meta} Elevates Direct QA

The D_{meta} dataset trains the model to not only modify answers but also to fully grasp a prompt, create feedback, and then develop a revised answer. This approach resembles training the model to think through a problem in a chain-of-thought methodically (CoT) manner, before responding. The training encompasses a thorough examination of the entire process, which not only betters the model's direct response capability but also enriches its understanding of the logic behind those answers, thereby enhancing its generalization ability.

A.3 Potentially Limited Plateau of Self-evolution Training

Based on eq. (2) and eq. (3), the model in the t^{th} round is updated to improve direct response quality by incorporating the generate-feedback-refinement process from the $(t-1)^{th}$ round. This is based on the assumption that the refined response is superior to the initial one generated by M_{evol}^{t-1}. As illustrated in Fig. 1, the direct generation performance of M_{evol}^{t} (green curve) consistently falls below the self-refinement of M_{evol}^{t-1} (blue curve). The self-refinement gains in the $(t-1)^{th}$ round indicate the potential benefit that the t^{th} round self-evolution could bring to direct generation. This also helps determine when to halt the self-evolution process, i.e., the process can be stopped when self-refinement brings no benefit to the direct response.

B Prompt of Generating Feedback and Refinement for D_{meta}

We introduce the prompt for generating feedback and refinement in two domains: Math and General. We outline specific prompts designed to guide the evaluation and improvement of responses to questions for building D_{meta} in each domain.

B.1 Math Domain

For the Math Domain, the prompt instructs evaluators to assess the quality of a response to a math question, provide a step-by-step analysis, and determine its correctness. If the response is incorrect, the evaluator is asked to refine and provide a correct answer.

> **Prompt for feedback and refinement:**
> **(Feedback)** Please assess the quality of the response to the given question.
> Here is the question: p.
> Here is the response: r.
> Firstly, provide a step-by-step analysis and verification for response starting with "Response Analysis:".
> Next, judge whether the response correctly answers the question in the format of "judgment: correct/incorrect".
> **(Refinement)** If the answer is correct, output it. Otherwise, output a refined answer based on the given response and your assessment.

B.2 General Domain

For the general test, aligned with the methodology described in section 3, we deploy the following prompt to guide an LLM-based annotator in generating response feedback and refinement. This prompt serves as the foundation for the meta-skill learning corpus and assists in producing self-evolution training data in the general test setting.

> **Prompt for feedback and refinement:**
> **(Feedback)** Please assess the quality of response to the given question.
> Here is the question: p.
> Here is the response: r.
> Firstly provide an analysis and verification for response starting with "Response Analysis:".
> Next, then rate the response on a scale of 1 to 10 (1 is worst, 10 is best) in the format of "Rating:"
> **(Refinement)** Finally output an improved answer based on your analysis if no response is rated 10.

C Data Generation

C.1 D_{meta} Data Quantity

The D_{meta} dataset was generated using 3.5k unlabeled prompts from GSM8K and 2K from SVAMP[3].

For general tests, 6K conversations were selected from 90K ShareGPT dialogues to form the general D_{meta} data.

C.2 Unlabeled Prompts for Self-Evolution Training

Math Domain: For math tests, unlabeled prompts in self-evolution training were sourced as follows:

(1) First round self-evolving phase: 4K leftover prompts from GSM8k and 1K from SVAMP, excluding those used in D_{meta}.

(2) Second/Third rounds: 10K/15K prompts were generated using Self-Instruct method (Wang et al., 2022b), based on a template shown in appendix C.2 with initial 4 to 6 seed examples.

General Domain: 15K unlabeled prompts from ShareGPT dialogues were used for self-evolution training data construction.

[3] Adhering to the official recommendation https://github.com/arkilpatel/SVAMP/tree/main, training prompts consist of MAWPS (Koncel-Kedziorski et al., 2016) and ASDiv-A (Miao et al., 2020)

You are an experienced instruction creator. You are asked to develop 3 diverse instructions according to the given examples.
Here are the requirements:
1. The generated instructions should follow the task type in the given examples.
2. The language used for the generated instructions should be diverse.
Given examples: {examples}
The generated instructions should be:
A. ...
B. ...
C. ...

D Training Hyperparameters

Our experiments were conducted in a computing environment with 8 NVIDIA V100 GPUs, each having 32GB of memory. All models were fine-tuned in a full-parameter setting. We utilized the AdamW optimizer for model training over 3 epochs, with a batch size of 128. The learning rate was set at 2e-5, including a 3% learning rate warmup period. Below we provide a comprehensive overview of the training hyperparameters employed in table 6. These parameters were uniformly applied across all training methods in our experiments.

Parameter	Value
Global Batch Size	128
LR	2×10^{-5}
Epochs	3
Max Length	2048
Weight Decay	0
Warmup Ratio	0.03

Table 6: Training hyperparameters.

We note that the SELF framework is compatible with versatile LLMs. In this study, we perform the experiment with **Vicuna-7b** (Chiang et al., 2023), a capable open-source instruction-following model fine-tuned from LLaMA-7b (Touvron et al., 2023), will be referred to simply as "Vicuna" in subsequent sections. To verify the generalizability of SELF, we also experiment with OpenLLaMA (Geng and Liu, 2023) and Vicuna-1.5 (Chiang et al., 2023) in Appendix N.

E Case Study Analysis

This subsection provides an in-depth case study that contrasts the performance of the original Vicuna and Vicuna + SELF models. Illustrated in fig. 4, both models perform initial predictions, followed by self-feedback and refinement steps. Notably, Vicuna's refinement fails to correct its initial errors, while Vicuna + SELF effectively utilizes self-feedback and refinement to derive an accurate and logically coherent answer.

F Benchmark Details

GSM8K (Cobbe et al., 2021) contains high-quality, linguistically diverse grade school math word problems crafted by expert human writers, which incorporates approximately 7.5K training problems and 1K test problems. The performance is measured by accuracy (%). **SVAMP** (Patel et al., 2021) is a challenge set for elementary Math Word Problems (MWP). It is composed of 1K test samples. The evaluation metric is accuracy (%). **Vicuna testset** (Lianmin et al., 2023) is a benchmark for assessing instruction-following models, containing 80 examples across nine skills in mathematics, reasoning, and coding. **Evol-Instruct testset** (Xu et al., 2023) includes 218 real-world human instructions from various sources, offering greater size and complexity than the Vicuna testset. **Arc** (Accuracy Normalized) (Clark et al., 2018) assesses the model's performance on answering multiple-choice questions. **TruthfulQA** (Multiple Choice 2) (Lin et al., 2022) evaluates the model's ability to discern truthful answers from deceptive ones. **Winogrande** (Accuracy) (Sakaguchi et al., 2020) tests the model's competency in completing fill-in-the-blank tasks with binary options for commonsense reasoning. **HellaSwag** (Accuracy Normalized) (Zellers et al., 2019) evaluates the model's understanding of daily situations and commonsense reasoning. **MMLU** (Accuracy) (Hendrycks et al., 2021) assesses the model's proficiency in generating language responses comprehensively.

G Baseline Details

(1) Vicuna + $D_{\mathbf{QA}}$: To demonstrate the improvement brought by SELF and exclude the impact of standard domain-specific supervised fine-tuning (SFT), we set a direct baseline that trained solely on pseudo-labeled question-answer pairs in the meta-skill training corpus. Specifically, we construct

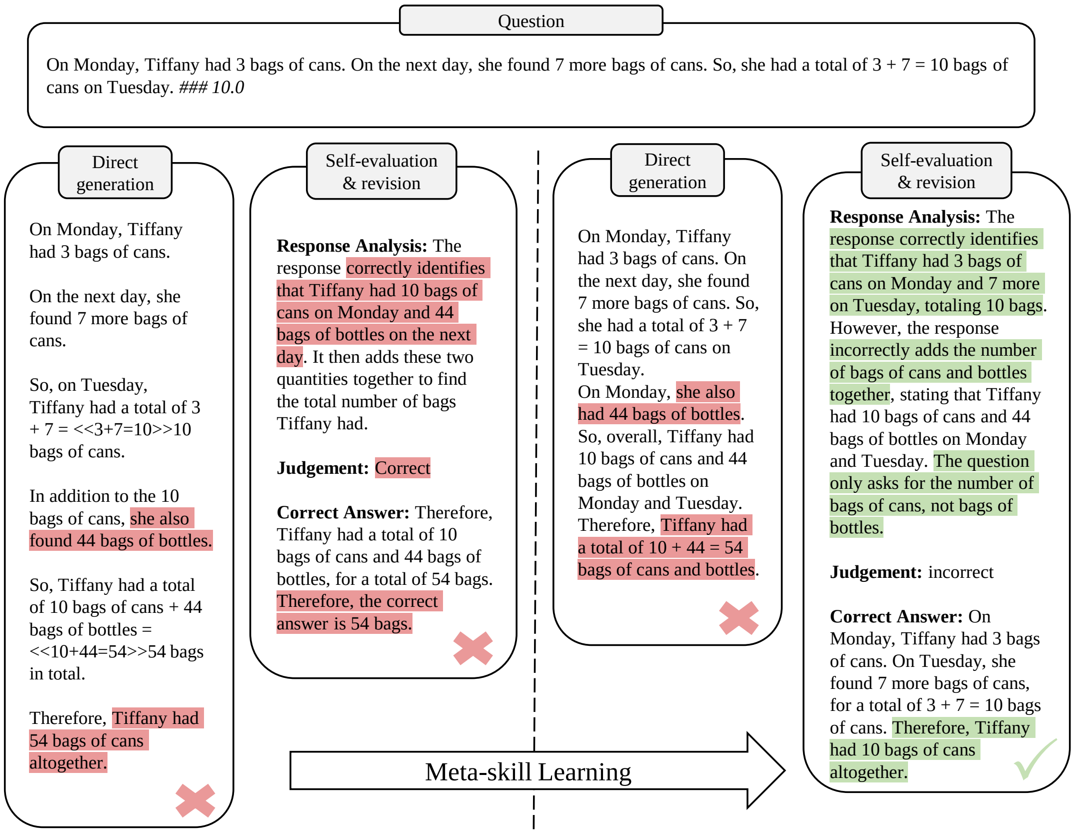


Figure 4: Case study comparing the original Vicuna (left) and Vicuna+SELF (right) on a SVAMP problem. Both models generate direct predictions and undergo self-feedback and self-refinement. Both models initially produce answers, followed by self-feedback and self-refinement. Vicuna maintains the incorrect response after refinement, whereas Vicuna+SELF demonstrates enhanced self-refinement, leading to a correct and logically consistent solution.

D_{QA}, which includes all the $(p, \hat{r})$ pairs from D_{meta}, and fine-tune the model as:

$$\mathcal{L}_{\text{QA}}(\phi) = -\mathbb{E}_{(p,\hat{r})\sim D_{\text{QA}}}\left[\log \tau_\phi(\hat{r}|p)\right].$$

We refer to this approach as Vicuna + D_{QA}, the most straightforward baseline. The performance gap between Vicuna + D_{QA} and SELF verify the efficacy of the proposed SELF framework, excluding the effect of training on domain-specific QA data.

(2) RLHF: we utilize the RLHF implementation from trlx[4]. We apply the same SFT model as the policy model for RLHF, **Vicuna +** $\boldsymbol{D_{\text{QA}}}$ as described above, which is consistent with SELF. The reward model is initialized from **Vicuna-7b** and is fine-tuned using pair-wise comparison data derived from the meta-skill training corpus D_{meta} (section 3.1.1), where the refined response $\hat{r}$ is presumed to be better than the original one r.

(3) Self-Consistency: we compare the self-refinement inference strategy in SELF with the Self-Consistency (Wang et al., 2022a) (i.e., multiple sampling and selecting an answer with majority voting) and examine their combined efficacy.

[4]https://github.com/CarperAI/trlx

H Meta-Skill Training Corpus

The example shown below exemplifies a standard training example from our meta-skill corpus. It illustrates the model's initial response, followed by its self-feedback, and the ensuing refinement. This process demonstrates how the model is trained for self-feedback and self-refinement capabilities.

Question: Greg has an alarm set to ring three times a day as a reminder. When the alarm goes off, it continues to ring until Greg turns it off. The first time it went off today, it rang four times. The second time it went off, it rang for three times as long as the first time. The third time, it rang for half as long as the second time. How many times did the alarm ring in all?
Original Response: The alarm rang four times the first time, three times as long as the first time the second time, and half as long as the second time the third time. So, the total number of times the alarm rang is 4 + 3*3 + 2*2 = 24.
Feedback: The response correctly interprets the problem. The first time the alarm rang, it rang for 4 times. The second time it rang, it rang for 3 times as long as the first time, which is 3*4 = 12 times. The third time it rang, it rang for half as long as the second time, which is 12/2 = 6 times. However, the final calculation is incorrect. The total number of times the alarm rang is not 4 + 3*3 + 2*2 = 24, but 4 + 12 + 6 = 22. *Judgement: incorrect*
Refined Response: The alarm rang four times the first time, three times as long as the first time the second time, and half as long as the second time the third time. So, the total number of times the alarm rang is 4 + 12 + 6 = 22.

Algorithm 1 Two-Phase SELF Process

Data: (1) Meta-Skill training data (D_{meta}) and (2) unlabeled prompts
Input: An initial Language Model M_{init}
Result: A stronger Language Model M_{evol}^T after self-evolving

```
// Meta-Skill Learning Phase
```
Data: Meta-Skill learning corpus (D_{meta})
M_{meta} = Supervised_fine_tuning(M_{init}, D_{meta})

```
// Self-Evolving Phase
```
Initialize M_{evol}^0 with M_{meta}
foreach *iteration t in 1 to Number of self-evolving iterations T* **do**
  `// Data-Augmentation`
  Initialize D_{evol}^t as an empty set
  foreach *prompt p_{evol}^i in t^{th} unlabeled prompts* **do**
    Generate direct output r_{evol}^i using M_{evol}^{t-1}
    Generate self-refined output $\hat{r}_{\text{evol}}^i$ from r_{evol}^i using M_{evol}^{t-1}
    Use M_{evol}^{t-1} to filter the self-refined output
    Add $(p_{\text{evol}}^i, \hat{r}_{\text{evol}}^i)$ to D_{evol}^t, where r_i is the refined response
  end
  `// Self-Evolution Training`
  M_{evol}^t = Supervised_fine_tuning(M_{evol}^{t-1}, D_{evol}^t)
end

```
// Training Complete
```
return Improved Language Model M_{evol}^T

I Algorithm

The "Two-Phase SELF Process" algorithm outlines a method for developing a base language model through a two-staged approach: Meta-Skill Learning and Self-Evolving. The process starts with training on a "Meta-Skill Learning corpus", which consists of data representing the generation, feedback and refinement process. Following this, the model enters the "Self-Evolving Phase", where it undergoes iterative refinements, employing data augmentation in each iteration to produce self-refined outputs from its previously refined versions. This iterative self-evolution aims to leverage accumulated knowledge and further enhance the model with newly generated data. The final outcome is an advanced Language Model that has significantly evolved from its original state through multiple self-evolution stages. More details are delineated in Alg. 1.

J Data Filtering Standards

We design a boolean function, *qualified*(f), to evaluate feedback f across different domains, determining if a response to a specific prompt satisfies essential quality criteria.

In the **Math Domain**,the function assesses feedback based on the explicit statement of "correctness" in the evaluator's judgment, aligned with the prompt structure in appendix B.1. It checks if the word "correct" immediately follows the phrase "judgment:" in the feedback. A presence of "correct" results in *qualified*(f) returning 1, meeting the qualification criteria. Absence leads to a return of 0.

For the **General Domain**, following the structure in appendix B.2, *qualified*(f) extracts and evaluates a numerical rating from the feedback. If the rating, found after "Rating:", is 7 or higher, the function returns 1, indicating qualification. Ratings below 7 return 0, failing to meet the threshold. A rating of 7 balances quality and training data quantity.

qualified(f) is key in both domains for filtering and assessing feedback quality, ensuring only high-quality responses are used for refined answer generation in self-evolution training. Post data filtering, Ψ^{t-1} in eq. (3) requires an update to $\Psi'^{t-1} = \Psi^{t-1} \times$ *qualified*(f), adding a quality filter through self-feedback. For clarity, we continue using original formulation as stated in eq. (3) in the main text.

K Multiple v.s. Single Self-Refinement

This study explores the effects of two meta-skill training data organization strategies on model performance: (1) Multiple Self-Refinement ($D_{\text{meta-multi}}$), involving the sampling of three responses for the model to choose the best for refinement, and (2) Single Self-Refinement (D_{meta}), where the model generates and refines a single response.

table 7 compares these methods' performances. Both strategies show performance gains with increased training data volume. However, as data volume expands, the multiple-response refinement shows a smaller improvement in direct generation performance ($+4.02\%$) than the single-response method ($+5.84\%$). Considering the simplicity and computational efficiency of the single-response method, which only samples one response during inference, and its better performance than the multiple-response approach, we have opted for the single-response refinement strategy in our experiments.

Data Size	Vicuna + D_{meta}	Vicuna + $D_{\text{meta-multi}}$
3.5k	25.39 $\rightarrow$ 28.28	25.92 $\rightarrow$ 27.29
7.5k	31.23 $\rightarrow$ 32.98	29.94 $\rightarrow$ 32.14

Table 7: Performance comparison of single and multiple response refinement with varying volumes of meta-skill training data. The arrow indicates improvement from direct generation to self-refinement: "direct generation $\rightarrow$ self-refinement".

L Self-Evolution Training: Continual Training v.s. Restart Training

Training Approach	DR (%)	SR (%)
Base Model	24.49	24.49
Restart Training	**27.67**	**29.34**
Continual Training (Mixed Data)	27.22	28.43
Continual Training (D^t_{evol} Only)	24.87	25.85

Table 8: Analysis about varied self-evolution training methodologies on GSM8K.

"Restart Training", which combines meta-skill learning corpus with all rounds of self-evolution training data, significantly improves direct generation (+3.18%) and self-refinement (+3.85%).

"Continual Training (Mixed Data)", where the model is trained simultaneously with all rounds of self-evolution data, also shows notable enhancements in direct generation (+2.73%) and self-refinement (+3.94%). In contrast, "Continual Training (D^t_{evol} Only)", which trains the model sequentially with self-evolution data from each round, demonstrates more modest gains (+0.38% in direct generation, +0.98% in self-refinement). The relatively lower performance of the latter approach highlights the importance of a mixed data strategy for effective self-evolution training.

Throughout our main text, we have consistently employed the "Restart Training" method. This approach was selected for its superior performance, as evidenced in table 8. In addition, the integration of D_{meta} into the self-evolution training is crucial to prevent the potential catastrophic forgetting of meta-skills. This strategy is essential for preserving the effectiveness and reliability of the self-evolution training process, as highlighted in section 3.2.2.

M SELF vs. Supervised Fine-Tuning on 7.5K GSM8k training data.

DR (%)	SR (%)	D_{QA}	D_{meta}	Self Evol. 1st	Self Evol. 2nd
28.05	-	✓			
31.23	32.98		✓		
35.43	36.22		✓		
37.87	**38.12**		✓	✓	✓
35.70	-	SFT	(GSM8K training data)		

Table 9: Comparison between SELF and Supervised Fine-Tuning on GSM8K. A "-" symbol in the table indicates self-refinement was not conducted during inference because the model lacks the necessary self-refinement skill.

When fine-tuned on the GSM8K 7.5k training set, the Vicuna model achieves an accuracy of 35.70%, which is lower than the SELF method (37.87%).

The experiments in table 9 use 7.5k meta-skill data, ensuring a fair comparison with the supervised fine-tuned model. This approach differs from the one in section 4.2.1, where only 3.5k meta-skill data are used.

table 9 indicates that, with 7.5k unlabeled training prompts for the meta-skill learning corpus, Vicuna + D_{QA} achieves 28.05%. Post meta-skill learning, direct generation results improve to 31.23%, further increasing to 32.98% after self-refinement. Subsequent self-evolution rounds lead to performance gains, reaching 37.87%(direct generation) and 38.12%(self-refinement) in the second round, outperforming supervised fine-tuning (35.70%).

Continuous Improvement of SELF vs. Supervised Fine-tuning: SELF's primary advantage lies in its ability for continuous improvement and adaptation. In contrast to supervised fine-tuning, SELF doesn't rely on human or external LLM annotations (like GPT3.5/GPT4) for training data in self-evolution training.

N Scalability of SELF Framework

To explore how SELF performs with different starting model qualities, we conduct experiments using the OpenLlama-3b model (Geng and Liu, 2023), a smaller LLM along with a stronger LLM, VicunaV1.5(finetuned from Llama2-7b)l (Chiang et al., 2023), on the GSM8K dataset. This allows us to assess SELF's adaptability to model quality. Experiments with SELF are based on the first round of self-evolution. The results are as follows:

Model	DR(%)	SR (%)
OpenLlama-3b	2.04	1.01
OpenLlama-3b + D_{QA}	12.13	10.97
OpenLlama-3b + D_{QA} + SELF	15.32	15.78
Vicuna (Llama-7b)	16.43	15.63
Vicuna + D_{QA}	24.49	24.44
Vicuna + D_{QA} + SELF	27.67	29.34
VicunaV1.5 (Llama2-7b)	18.5	17.43
VicunaV1.5 + D_{QA}	26.04	25.48
VicunaV1.5 + D_{QA} + SELF	**30.22**	**32.43**

Table 10: Scalability of the SELF framework across different models.

Applicability and Robustness of SELF Framework: The average improvement of 17.32% via direct generation and 16.87% after self-refinement underscores the framework's scalability and efficacy. It reveals a consistent positive impact of the SELF Framework across diverse models.

SELF Framework exhibits enhanced performance on more powerful models: As shown in table 10, applying SELF to VicunaV1.5 results in the most significant gains - 30.22% in direct generation and 32.43% after self-refinement, surpassing the performance on Vicuna and OpenLlama-3b. This indicates that the effectiveness of the SELF framework improves with the underlying model's capabilities.

O Impact of Meta-Skill Corpus Quality

We examine the influence of meta-skill learning quality on the self-evolution process with the following results:

Training Stage	DR (%) (GPT-3.5-turbo/GPT4)	SR (%) (GPT-3.5-turbo/GPT4)
Vicuna + D_{meta}	24.84/25.39 (0.55↑)	25.22/28.28 (3.06↑)
Vicuna + D_{meta} + SELF Evol.	25.11/27.67 (2.56↑)	25.47/29.34 (3.87↑)

Table 11: Effect of meta-skill corpus quality on model performance using GPT-3.5-turbo and GPT4.

The presented table 11 demonstrates the remarkable performance improvements achieved by using GPT-4 for generating the meta-skill corpus in our SELF framework, compared to using GPT-3.5-turbo. The table shows significant enhancements in both direct generation and self-refinement across training stages when GPT-4 is utilized. For instance, in the "Vicuna + D_{meta}" stage, direct generation performance increases from 24.84% with GPT-3.5-turbo to 25.39% with GPT-4, marking a gain of 0.55%. Similarly, in the "Vicuna + D_{meta} + SELF Evolution" stage, the self-refinement result improves from 25.47% with GPT-3.5-turbo to 29.34% with GPT-4, showing an enhancement of 3.87%.

This analysis highlights the significant impact of utilizing high-quality meta-skill training data on the performance of the Vicuna model within the SELF framework. The shift from GPT-3.5-turbo to GPT-4 for the generation of the meta-skill corpus leads to consistent improvements in both Direct Generation and Self-Refinement metrics.

P Single-Round vs. Iterative Self-Evolution Training

Given an equal number of unlabeled prompts, we evaluate the effectiveness of training within a single-round versus iterative training. The former method uses a single model to self-curate training data from all available unlabeled prompts at once. In contrast, the latter method involves dividing the unlabeled prompts into multiple parts. For the iterative approach, the model is initially trained on a portion of the unlabeled prompts and self-curated labels. Following this, the trained model is employed to create new training data based on previously unused prompts. As described in our main text, we divide the unlabeled prompts into three parts, enabling the model to undergo three iterative rounds of self-evolution.

table 12 shows that in the "Single-Round" training, the performance is 28.40% for direct generation and 30.55% for self-refinement. In con-

Training Method	DR (%)	SR (%)
SELF (Single-Round)	28.40	30.55
SELF (Iterative)	29.64	31.31

Table 12: Comparison of single-round training and iterative training.

trast, the iterative approach yields higher scores of 29.64% for direct generation and 31.31% for self-refinement.

Advantages of Iterative Training: Iterative training benefits from the enhanced capabilities of LLMs in subsequent rounds, which generate higher-quality training data and lead to improved test performance.

Q Analysis on Data Filtering with Self-Feedback

Filter Strategy	Training Acc. (%)	Test Acc. (%)
Unfiltered	29.89	26.90
Filtered	**44.10**	**27.67**

Table 13: Impact of Data Filtering with Self-Feedback on GSM8K. "Training Acc." shows the accuracy of the self-evolution training data post-filtering, and "Test Acc." represents the model's test performance post-training on these filtered data.

table 13 presents an analysis of filtering self-evolution training data using self-feedback (section 3.2.1) on GSM8K, focusing on training data quality and its influence on self-evolution training. The filtering criteria are detailed in appendix J.

The combination of self-refinement and self-feedback filtering results in higher self-evolution training data accuracy (+14.21%) and improved fine-tuned model performance (+0.77%). Despite the significant training data accuracy improvement, the performance gain is modest due to the reduced data size (from 4K to 1.8K) after filtering.

R General Test Details

Five Open LLM Leaderboard datasets Tt is noteworthy that limitations were observed in the Winogrande task. Specifically, incorporating external data, the Vicuna + D_{QA} model failed to enhance performance on the Winogrande task and even contributed to model degradation after self-evolution. This observation suggests that the SELF-evolution process aims to unlock and amplify the inherent potential of the base model rather than distilling unknown skills.

Vicuna and Evol-instruct Test Evaluations We utilize GPT-4 to evaluate the models' responses on both test sets. We follow the assessment methodology proposed by (Xu et al., 2023), which mitigated the order bias presented in the evaluation procedures.

S Other Related Works

Recent advancements in autonomous improvements of large language models (LLMs) have spurred significant research into methodologies aimed at aligning LLM behavior with human intentions. Alignment strategies such as Reinforcement Learning from Human Feedback (RLHF) have gained traction, wherein a reward model is trained to approximate human preferences, and subsequently, an LLM is fine-tuned through reinforcement learning to maximize this estimated human preference. Several comparative studies shed light on distinct approaches. For instance, SELF, compared to Promptbreeder (Fernando et al., 2023) and AutoCoT (Zhang et al., 2023), focuses on internal self-enhancement rather than prompt evolution or diversity generation. In contrast to CRITIC (Gou et al., 2023), which employs external tools for validation, SELF relies on internal language feedback for self-refinement. While Multiagent Debate (Du et al., 2023) enhances factuality through debate formats, SELF operates as a single-agent framework. Constitutional AI (Bai et al., 2022b) emphasizes harmlessness principles, whereas SELF offers a more general approach without specific constraints. Unlike open-ended learning (Team et al., 2021), which aims at creating generally capable agents in diverse environments, SELF concentrates on language-based self-improvement within a single-agent framework. SPIN (Chen et al., 2024) aims to iteratively improve the LLM's performance by leveraging both ground truth and synthetic data it generates, thereby narrowing the quality gap between human-labeled and LLM-generated responses. Conversely, SELF autonomously refines its capabilities without relying on ground truth data. Self-Rewarding (Yuan et al., 2024) resembles the Reinforcement Learning with Human Feedback (RLHF). It assigns single numerical values as feedback via LLM-as-a-Judge prompting, using Direct Preference Optimization (DPO) for self-improvement training. In contrast, SELF provides comprehensive language feedback, evaluating and guiding self-refinement, and employs Supervised Fine-Tuning (SFT) on self-refined responses, which is a more clear and coherent training framework.

T Ablation Findings

(1) Meta-skill Training Elevates Performance: Training with the meta-skills dataset D_{meta} as defined in eq. (1), and setting $\beta = 1$ for a fair comparison with the question-answer dataset D_{QA}, improves **direct response** performance. Specifically, we observe an increase of +0.90% on the GSM8K dataset and +1.9% on the SVAMP dataset, compared to using the D_{QA} dataset alone. This underscores the interesting finding that arming the model with self-feedback and self-refinement meta-skills implicitly elevates its capacity to generate superior responses directly, even without explicit self-refinement. We offer an insight in appendix A.2.

(2) Continuous Improvement through Self-Evolution: The results reveal that three self-evolution rounds consecutively yield performance enhancements (e.g., $25.39\% \xrightarrow{+2.28\%} 27.67\% \xrightarrow{+0.99\%} 28.66\% \xrightarrow{+0.98\%} 29.64\%$ on GSM8K). This shows that the model actively self-evolves, refining its performance autonomously without additional manual intervention.

(3) Persistent Efficacy of Self-Refinement: After meta-skill learning, regardless of model variation, executing self-refinement consistently results in notable performance improvements. This shows that the self-refinement meta-capability learned by SELF is robust and consistent across evolution steps.