Paper	Task	Using Oracle Info for Feedback	Weak Prompt for Initial Responses	Comments
RCI (2023, §3.1)	Computer Tasks	✓ stop condition	–	Using ground-truth answers and do not update correct responses, which unfairly ignores false-positive correction
Reflexion (2023, §4.2)	HotpotQA (Context)	✓ feedback	–	Feedback is the exact match between the responses and ground-truth answers
CAI Revisions (2022)	Detoxification	–	✓	Initial generation is not prompted to remove harmful outputs
Self-Refine (2023)	Math, Coding, Dialogue	–	✓	Unfairly weak or wrong instructions or few-shot demonstrations for initial response generation

Table 4: Unfair settings in prior studies of self-correction with prompting, over-evaluating self-correction.

difficult to generate reliable feedback on their responses only by prompting themselves (Gou et al., 2024; Huang et al., 2024a; Olausson et al., 2024; Chen et al., 2024f).

Unrealistic or Unfair Settings. The conflicting positive and negative results motivate us to analyze when LLMs can self-correct *only by prompting themselves*. Specifically, we assess whether prior studies satisfy the requirements to verify that [RQ1] LLMs can self-correct their responses based solely on their inherent capabilities. As in Table 4, we find that many studies use either oracle information in the self-correction processes (unrealistic frameworks) or weak prompts that can be easily improved for generating initial responses (unfair settings), which over-evaluate self-correction. Consequently, we conclude that no major work shows successful self-correction of responses from LLMs using feedback generated by prompting themselves under fair settings in general tasks. **Oracle Information:** RCI Prompting (Kim et al., 2023) uses ground-truth answers and does not apply self-correction when the initial responses are correct, which unfairly ignores mistakes caused by updating correct responses incorrectly. Reflexion (Shinn et al., 2023) generates feedback by using an exact match between the generated and ground-truth answers, which cannot be accessed in real-world applications. **Weak Initial Responses:** Detoxifying harmful responses is a popular task in self-correction research, but prior studies often study in situations where initial response generation is not instructed to generate harmless responses (Bai et al., 2022; Wang et al., 2024b). Although detecting harmful contents using LLMs is a reasonable research topic, this setting is not the self-correction from best-possible initial responses, since we can improve the initial response generation process by instructing not to generate harmful responses. As more obvious weak prompts, Self-Refine (Madaan et al., 2023) uses instructions or few-shot examples that do not correctly correspond to the target task only for initial response generation (e.g., providing wrong target labels in few-shot examples), while using appropriate instructions for self-correction, as shown in Table 9 and 10. These settings evaluate improvement from weak initial responses, which over-evaluate the improvement by self-correction.

Tasks in which Self-Correction is Exceptionally Effective. Although our analysis of prior studies shows that intrinsic self-correction is difficult in general, some tasks have properties that make feedback generation easy and enable intrinsic self-correction. For example, CoVe (Dhuliawala et al., 2024) is an intrinsic self-correction method for tasks of generating multiple answers, such as *Name some politicians who were born in NY, New York.* Generated responses include multiple answers, but the feedback generation can be decomposed into easier sub-tasks of verifying each answer. Tasks with **decomposable responses** are one of the few groups of tasks for which verification is clearly easier than generation, which enables intrinsic self-correction. However, many real-world tasks do not satisfy this property.

5 Self-Correction with External Information

> [RQ2] Can LLMs self-correct their best-possible initial responses *assisted by external information?*

This section analyzes self-correction frameworks that make use of external tools, external knowledge, and fine-tuning.

5.1 Self-Correction with External Tools or Knowledge

Given the observation that feedback generation is a bottleneck of self-correction (§4), improving feedback using external tools or knowledge is a promising direction. External tools used for self-

Paper	Main Task	External Tools or Knowledge	
		For Initial Response Generation	For Feedback Generation
Reflexion (2023, §4.1, 4.3)	Games, Coding	–	Game Envs, Code Interpreter
CRITIC (2024)	GSM8k, SVAMP	–	Python interpreter
Self-Debug (2024e)	Text-to-Code	–	Code Interpreter
CRITIC (2024)	HotpotQA	Web Search	Web Search
FLARE (2023b)	2WikiMultihopQA, StrategyQA, ASQA	Web Search	Web Search
RARR (2023)	NQ, SQA, QReCC	–	Web Search
ReFeed (2023)	NQ, TriviaQA, HotpotQA	–	Wikipedia

Table 5: Self-correction with external tools or knowledge (with in-context learning).

correction include code interpreters for code generation tasks (Chen et al., 2024e; Gou et al., 2024) and symbolic reasoners for logical reasoning tasks (Pan et al., 2023). A popular source of knowledge is search engines, which are often used with queries generated from initial responses to retrieve information for validating their correctness (Gao et al., 2023; Jiang et al., 2023b). These prior studies widely agree that self-correction can improve LLM responses when reliable external tools or knowledge suitable for improving feedback are available.

Unfair self-correction with external information. Although using external tools or knowledge is known to be effective in self-correction, we raise caution that the way of using external tools or knowledge influences the research questions we can verify (§3.1). As shown in Table 5, some prior studies (Gao et al., 2023; Yu et al., 2023; Zhao et al., 2023) use external knowledge only for self-correction, while they can also directly use external knowledge to improve the initial response generation process. For example, RARR (Gao et al., 2023) uses external knowledge to detect mistakes in initial responses, while it does not use any external knowledge when generating initial responses. These methods are reasonable when only focusing on [RQ3] the performance of final responses, but it is not fair to use them for evaluating [RQ2] whether self-correction can improve from the best-possible initial responses. In contrast, using code interpreters for self-correction (Gou et al., 2024; Chen et al., 2024e) can be regarded as using best-possible initial responses because there is no easy way to improve the initial response generation directly.

Verifiable Tasks. Some tasks have a property that allows the correctness of the responses to be verified easily, even without external information. For example, the constrained generation task evaluated in Self-Refine (Madaan et al., 2023) is a task to generate a sentence that includes five specified words. We can easily evaluate the correctness by checking whether the five words are included in the generated sentence. Tree-of-thought (Yao et al., 2023) is a generate-and-rank method for verifiable tasks,[1] such as Game of 24, the task to obtain 24 using basic arithmetic operations ($+, -, \times, \div$) and provided four integers. For Game of 24, we can easily verify the answer by checking whether the generated answer is 24. We consider self-correction to work well in these tasks because they are in the same situations as using strong external tools or the oracle information to generate feedback.

5.2 Self-Correction with Fine-tuning

Prior work shows that fine-tuning LLMs for generating feedback or refining responses improves the self-correction capability. A common approach fine-tunes feedback models to generate reference feedback given initial responses and fine-tunes refinement models to generate reference answers given the initial responses and reference feedback (Ye et al., 2023; Lee et al., 2024; Saunders et al., 2022). **Frameworks:** The first approach fine-tunes *the same model* to correct its own responses. In this approach, most methods fine-tune models for all stages: initial responses, feedback, and refinement (Saunders et al., 2022; Ye et al., 2023; Lee et al., 2024). Another approach corrects responses from larger models using *smaller fine-tuned models*. This cross-model correction approach often instructs the larger models to refine their own responses using feedback from the smaller fine-tuned models (Yang et al., 2022b; Welleck et al., 2023; Akyurek et al., 2023; Paul et al., 2024), which can be viewed as using the small fine-tuned models as external tools. **Training Strategies:** A popular approach is supervised fine-tuning, which fine-tunes self-correction modules on human-annotated feedback (Saunders et al., 2022), feedback from

[1] Tree-of-thought is a generate-and-rank method and not a self-correction method in our definition.

Paper	Main Task	Cross-Model	SFT Tasks	Initial Responses		Feedback Generation			Refinement	
				Model	SFT Target	Model	SFT Target	Size	Model	SFT Target
SelFee (2023)	MT-Bench	–	General Tasks	Llama (7B,13B)	ChatGPT Responses	Llama (7B,13B)	ChatGPT Feedback	178K	Llama (7B,13B)	ChatGPT Refinement
Volcano (2024)	Visual Reasoning	–	General Tasks	LLaVA (7B, 13B)	GPT-3.5-T, Human	LLaVA (7B, 13B)	GPT-3.5-T Feedback	274K	LLaVA (7B, 13B)	Reference Answers
Self-Critique (2022)	Topic-based Summarization	–	Target Task	Instruct GPT	Human Summaries	Instruct GPT	Human Feedback	100K	Instruct GPT	Human Refinement
REFINER (2024)	Math, Logic, Moral Stories	✓	Target Task	GPT-3.5	–	T5-base	Synthetic Data	20K - 30K	GPT-3.5	–
Self-Edit (2023a)	Code Generation	✓	Target Task	GPT-3	–	(Code Executor and Test Cases)			PyCodeGPT 110M	Reference Code

Table 6: Self-correction with supervised fine-tuning. Most methods require large training datasets. "–" on the "SFT Target" columns represents no fine-tuning.

stronger models (Ye et al., 2023), or synthetic negative responses (Paul et al., 2024). As other approaches, to avoid the cost of collecting human feedback, self-corrective learning (Welleck et al., 2023) selects model-generated feedback that successfully refines responses as training data, GLoRe (Havrilla et al., 2024) constructs a synthetic refinement dataset using model-generated feedback, and RL4L (Akyurek et al., 2023) uses reinforcement-learning. **External Tools:** Some works fine-tune models to refine responses given feedback from external tools. Self-Edit (Zhang et al., 2023a) uses the results on test cases evaluated by code executors for code generation, and Baldur (First et al., 2023) uses proof assistants for improving proof generation.

Large Training Data for SFT of Feedback. As shown in Table 6, many methods with supervised fine-tuning for feedback generation rely on training data with more than 100K instances. These studies often use feedback generated by stronger models to simulate human annotation, but this approach requires large-scale human annotations to be implemented on state-of-the-art models. We expect future research to explore approaches that do not require large-scale human annotations (§11).

Unfair Fine-tuning. Some studies (Welleck et al., 2023) apply stronger fine-tuning for self-correction models than initial response generation models, which do not use best-possible initial responses in the available resources (§3.2). This approach can be used to evaluate [RQ3] the performance of the final responses to compare with other methods but cannot be used to evaluate [RQ2] the improvement from best-possible initial responses.

6 Strong Baselines

> [RQ3] Are the final outputs from self-correction *better than other methods?*

Self-correction involves multiple LLM calls for generating feedback and refinement. Therefore, to claim that [RQ3] the performance of the final outputs from self-correction frameworks is better than other approaches, it should be compared with sufficiently strong baselines, possibly relying on additional LLM calls or computational cost. Many self-correction studies do not compare their methods with strong baselines, although some studies pointed out this issue and compare self-correction with self-consistency (Gou et al., 2024; Huang et al., 2024a) or pass@k in code generation (Zhang et al., 2023a; Olausson et al., 2024). We encourage future research to compare self-correction with strong baselines, including self-consistency and generate-and-rank, to further explore RQ3.

Self-Consistency (Wang et al., 2023) is an approach that generates multiple responses for the same input and takes the majority vote of the final answers in reasoning tasks. The idea of selecting good responses using the consistency between multiple responses from the same model has also been extended to other tasks such as text generation (Manakul et al., 2023; Elaraby et al., 2023; Chen et al., 2024d) and code generation (Shi et al., 2022).

Generate-and-Rank is an approach that generates multiple responses and selects the best response using verifiers. **Post-hoc** approach ranks responses using self-evaluation (Weng et al., 2023; Zhang et al., 2023b), confidence (Manakul et al., 2023), fine-tuned verifiers (Cobbe et al., 2021; Shen et al., 2021; Lightman et al., 2024), or verifiers with external tools (Shi et al., 2022; Chen et al., 2023; Ni et al., 2023). **Feedback-guided**