SIPDO: CLOSED-LOOP PROMPT OPTIMIZATION VIA SYNTHETIC DATA FEEDBACK

Yaoning Yu*, Ye Yu*, Peiyan Zhang, Kai Wei, Haojing Luo, Haohan Wang

Motivation

- Existing prompt optimization methods mostly rely on fixed datasets, assuming static input distributions and therefore missing unseen failure modes.
- Static optimization offers limited support for iterative self-improvement, since the prompt is not continuously challenged by new, failure-revealing examples.

Contribution

- We introduce a feedback-driven framework SIPDO that integrates **synthetic data** generation into **prompt optimization**.
- We develop a method to construct synthetic examples that dynamically stress-test prompts, revealing failure modes and guiding refinement.
- We empirically demonstrate that augmenting prompt optimization with synthetic data improves performance across multiple reasoning benchmarks, surpassing existing prompt tuning methods.

Method

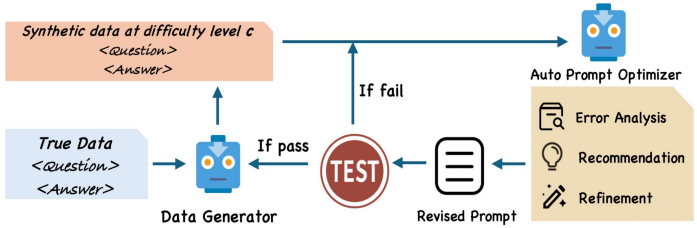

Figure 1: **Data Generator** first produces a synthetic question-answer pair at difficulty level c. The **Auto Prompt Optimizer** evaluates the prompt via three sub-modules-error analysis, recommendation, and refinement-and outputs a revised prompt. The revised prompt is tested on failures and previously solved examples. If errors remain, it returns for refinement; otherwise, it moves to the next sample (with higher c). The cycle repeats until no errors remain or the budget is reached, yielding a self-improved prompt.

The data generator first draws a target label $\tilde{y} \sim p^*(y)$ where $\{(x_i, y_i)\}_{i=1}^N \sim S$. By sampling a latent variable $z \sim g_\phi(z|S)$ that captures the structure of few-shot from true data distribution S, the decoder q_ψ produces $\tilde{x} = q_\psi(z, \tilde{y}, c)$ where c is a controlled difficulty tier. The generator is prompted to produce label-realistic yet failure-revealing examples by optimizing

$$\min_{\psi} \ \mathrm{KL}(q_\psi(y) \| p^*(y)) + \lambda \, \mathbb{E}_{q_\psi}[L(f(p, \tilde{x}), \tilde{y})],$$

At iteration $t \in \{1, \ldots, M\}$ and using the synthetic history $D_t = \{(\tilde{x}_j, \tilde{y}_j)\}_{j=1}^t \subseteq \mathcal{X} \times \mathcal{Y}$, SIPDO defines the error set $E^{(t)} = \{(\tilde{x}, \tilde{y}) \in D_t : f(p^{(t)}, \tilde{x}) \neq \tilde{y}\}$, analyzes these failures, generates a textual patch $\Delta^{(t)}$, and updates the prompt via $\tilde{p}^{(t)} = U_\theta(\Delta^{(t)}, p^{(t)}, E^{(t)})$. If errors are found in previously generated questions, the prompt is further refined; otherwise, the process moves on to the next difficulty level. The cycle repeats until no errors remain in any previously generated questions or the optimization budget is reached.

Results

MMLU Subject	Method	1st iteration	2nd iteration	3rd iteration	Final Result (Comparative Acc.)
College Computer Science	TextGrad	85	88	86	89(↓ 4.0)
	M-TextGrad	85	87	87	89(↓ 4.0)
	REVOLVE	85	88	89	90(↓ 3.0)
	SIPDO	–	–	–	**93**
Machine Learning	TextGrad	84.8	87.5	82.1	88.4(↓ 5.4)
	M-TextGrad	85.5	85.4	85.3	85.0(↓ 8.8)
	REVOLVE	85.7	86.6	85.7	88.4(↓ 5.4)
	ANN	–	–	–	90.1(↓ 3.7)
	SIPDO	–	–	–	**93.8**
College Biology	TextGrad	95.1	97.2	95.1	96.5(↓ 0.0)
	REVOLVE	96.5	96.1	97.2	96.5(↓ 0.0)
	SIPDO	–	–	–	**96.5**

Table 1: Results on MMLU Machine Learning, College Computer Science, and College Biology subject by GPT-4o, demonstrating SIPDO's effectiveness on different subjects

Model	Method	Accuracy (%)						Avg. Acc. (%) (Comparative Acc.)
		Penguins	Geome.	Epistemic	Obj. Count	Temporal	Causal	
GPT-4o	CoT	79.8	79.1	79.3	85.2	98.0	67.8	81.5(↓ 7.6)
	APE	84.8	65.3	84.8	86.0	99.2	74.0	82.4(↓ 6.7)
	PromptAgent	96.1	**83.0**	**91.6**	88.2	98.4	77.8	**89.2**(↑ 0.1)
	SIPDO	**96.4**	82.2	86.3	**91.1**	**99.3**	**79.0**	89.1
GPT-4o-mini	CoT	75.8	68.6	**85.2**	81.5	94.9	63.6	78.3(↓ 9.0)
	APE	83.7	44.5	81.6	86.3	97.2	75.6	78.2(↓ 9.1)
	PromptAgent	89.8	72.0	86.0	84.3	94.6	84.6	85.2(↓ 2.1)
	SIPDO	**92.1**	**73.2**	85.1	**87.5**	**98.0**	**88.0**	**87.3**
Gemini-1.5-flash	CoT	70.4	68.3	85.5	90.1	94.0	66.8	79.2(↓ 3.7)
	APE	37.6	49.4	**88.8**	84.7	**99.4**	69.4	71.6(↓ 11.3)
	PromptAgent	67.4	**70.3**	81.6	86.3	94.2	67.9	78.0(↓ 4.9)
	SIPDO	**77.3**	68.9	87.0	**92.3**	98.2	**73.2**	**82.9**
Gemini-1.5-pro	CoT	**81.8**	59.1	82.6	**92.8**	**98.9**	61.5	79.5(↓ 3.9)
	APE	40.2	56.6	88.7	78.6	86.0	65.7	69.3(↓ 14.1)
	PromptAgent	73.6	59.3	83.8	72.6	98.4	74.2	76.8(↓ 6.6)
	SIPDO	79.3	**64.3**	**89.3**	91.3	98.0	**78.3**	**83.4**

Table 2: Results on BIG-Bench tasks across multiple LLMs. SIPDO consistently outperforms standard prompting baselines (CoT, APE, PromptAgent) across most tasks and models, demonstrating generalization and effectiveness of the prompt optimization by synthetic data feedback.

Tasks	GPT-4o					GPT-4o-mini				
	Vanilla	Neuro-S	CoT	REVOLVE	SIPDO	Baseline	Neuro-S	CoT	REVOLVE	SIPDO
ProofWriter	58.5	**81.6**	72.3	54.0	79.6	52.6	**79.7**	61.8	48.6	79.3
FOLIO	71.2	79.2	72.6	65.7	**83.2**(↑ 4.0)	51.2	73.2	69.3	62.8	**81.1**(↑ 7.9)
PrOntoQA	80.4	85.2	95.6	85.4	**96.3**(↑ 0.7)	74.6	79.3	89.3	83.4	**91.3**(↑ 2.0)
Average	70.0	82.0	80.2	68.4	**86.4**(↑ 4.2)	59.5	77.4	73.5	64.9	**83.9**(↑ 6.5)

Table 3: Performance(%) on ProofWriter, FOLIO, and PrOntoQA by Neuro-Symbolic, CoT, REVOLVE, SIPDO, and Baseline Prompting methods across GPT-4o and GPT-4o-mini.

Model	Penguins	Geome.	Epistemic	Obj.Cnt.	Temporal	Causal	Avg.
GPT-4o	73.2 (↓ 24.1%)	68.1 (↓ 17.2%)	81.9 (↓ 3.1%)	53.8 (↓ 40.9%)	97.0 (↓ 2.3%)	68.4 (↓ 13.4%)	73.7 (↓ 17.3%)
GPT-4o-mini	69.6 (↓ 24.4%)	47.5 (↓ 35.1%)	80.0 (↓ 6.0%)	39.9 (↓ 54.4%)	92.1 (↓ 6.0%)	67.4 (↓ 23.4%)	66.1 (↓ 24.3%)

Table 4: Accuracy (%) after removing the difficulty gradient. Numbers in parentheses show the absolute drop (↓) relative to the performance with difficulty gradient placed.

Key Points & Insights

- SIPDO shows that prompt optimization can be improved through a closed-loop framework rather than static dataset.
- By combining synthetic data generation with iterative prompt refinement, SIPDO achieves stronger robustness and better performance on reasoning tasks.
- Progressive difficulty is key to stable refinement, forcing the prompt to improve gradually from challenging synthetic data instead of failing on overly hard examples.
- Prompt optimization can be feedback-driven, not limited to fixed benchmarks.