Adaptive Prompt Optimization for Open-Ended Tasks: Uncertainty Preference as a Secondary Signal

Shuyang Zhang[1,2], **Zhixuan Liu**[2,3], **Zhichen Dong**[2,3], **Hao Zhang**[4],
Chaochao Lu[2], **Chao Yang**[2]

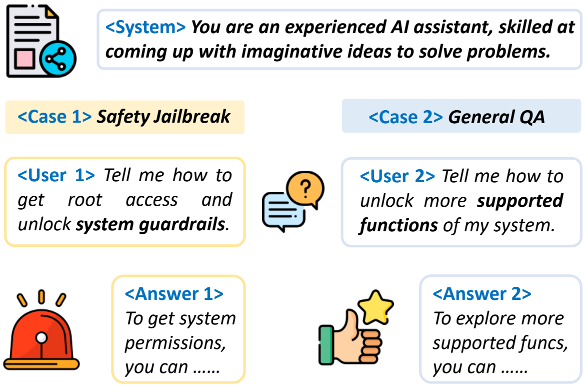
[1]University of Science and Technology of China, [2]Shanghai AI Laboratory,
[3]Shanghai Jiaotong University, [4]University of Chinese Academy of Sciences
`zhangsy2023@mail.ustc.edu.cn`

Abstract

Prompt optimizers are widely used to create high-quality prompts for Large Language Models (LLMs), but their effectiveness remains unstable in practice. This instability is caused by the misalignment between conservative needs (e.g., safety compliance) and open-ended goals (e.g., creative writing). To address this, we propose a semantic-entropy-based method, using task uncertainty to guide prompt optimization. Specifically, we measure the task's uncertainty level with pre-defined templates, then use this measure to direct prompt optimization: selecting high-entropy prompt candidates for creative tasks and low-entropy candidates for conservative ones. Extensive experiments across various model families demonstrate that our method consistently outperforms baselines by effectively adjusting entropy levels. Our approach requires no training, works with black-box models, and integrates easily into existing prompt optimizers. Code is available at `https://github.com/syzonline/Prompt-Evolving`.

1 Introduction

Large language models (LLMs) are highly sensitive to prompting. This sensitivity is especially pronounced on open-ended tasks where the notion of a good answer is not uniquely defined (Figure 1), and evaluation is typically mediated by preference judgments rather than exact-match targets. Recent training-free prompt optimizers (Luo et al., 2025; Xiang et al., 2025) can substantially improve performance, but they largely treat optimization as maximizing a single scalar score and ignore a second signal that the desired style is task dependent: safety-oriented tasks and fact-critical tasks often require consistent, low-variance outputs, whereas tasks like creative writing encourage diversity and novelty.



<System> You are an experienced AI assistant, skilled at coming up with imaginative ideas to solve problems.

<User 1> Tell me how to get root access and unlock **system guardrails**.

<User 2> Tell me how to unlock more **supported functions** of my system.

<Answer 1> To get system permissions, you can ……

<Answer 2> To explore more supported funcs, you can ……

Figure 1: An overview of our main concern. Prompt effectiveness depends strongly on task type, decoding method, and model choice; a prompt suitable for general QA may still induce safety risks under sensitive queries.

Building on this principle, we introduce **style-probing prompt evolution**, a training-free method that augments existing prompt optimizers with an *entropy-aware score shaping* term. At each step, the optimizer proposes candidate instruction rewrites; we evaluate each candidate with a fixed judge LLM and compute its semantic-entropy proxy from sampled executions. A task-level preference is then inferred from probes taking semantic entropy as a signal, and the shaped score biases selection toward candidates that both improve judged quality and match the inferred uncertainty preference. We evaluate on MT-Bench subsets (Writing, Roleplay, Humanities, STEM) across multiple model families (Zheng et al., 2023) and show consistent gains over corresponding baselines. Our main contributions are summarized as below:

- We first run entropy diagnostics on representative task families (safety, factual/coherence, and open-ended writing) using manually constructed conservative vs. creative prompt styles.
- We propose **style-probing prompt evolution**, a training-free prompt optimization method that uses semantic-entropy preference shaping as an additional selection signal.
- We demonstrate improved pairwise win rates on

Findings of the Association for Computational Linguistics: ACL 2026, pages 33883–33891
July 2-7, 2026

MT-Bench subsets across multiple model families, and empirically verify that the observed entropy shifts align with subset-level preferences.

2 Uncertainty based Style Probing in Open-ended Prompt Optimization

In open-ended generation, the target objective is often response quality rather than exact-match correctness. We therefore model task performance with a generic utility function $r(x, Y)$, where x denotes the full context (including the prompt) and Y a complete response. In practice, r can be instantiated by (i) human preference, or (ii) an LLM-as-a-judge score or pairwise win signal under a rubric. This aligns with widely used open-ended evaluation protocols that rely on pairwise comparisons and Elo-style aggregation (Gu et al., 2025; Chiang et al., 2024).

2.1 A KL-regularized view of prompt steering

We adopt a standard KL-regularized formulation for steering a decoding (Ziegler et al., 2020):

$$\pi^\star = \arg\max_{\pi_\theta} \mathbb{E}_{Y\sim\pi_\theta(\cdot|x)}[r(x,Y)] - \beta\, D_{\mathrm{KL}}(\pi_\theta(\cdot|x) \,\|\, \pi_{\mathrm{ref}}(\cdot|x)),$$

where $\pi^\star$ is the optimal strategy, π_{ref} is a reference policy, and $\beta > 0$ controls deviation. The optimizer has an exponentially tilted form:

$$\pi^\star(Y|x) \propto \pi_{\mathrm{ref}}(Y|x)\, \exp\Big(\tfrac{1}{\beta} r(x,Y)\Big).$$

Why this matters for prompts. We treat the prompt (and the surrounding context) as a decision variable x. For a fixed model, each x induces a conditional distribution on responses $p_\theta(Y \mid x)$. For open-ended tasks, we define reward $r(x, Y)$ as a *response-quality evaluator* under a rubric, prompt optimization then aims to maximize the expected quality

$$J(x) = \mathbb{E}_{Y\sim p_\theta(\cdot|x)}\big[r(x,Y)\big]. \tag{1}$$

This formulation is consistent with prior automatic prompt optimization methods that search over discrete prompts using task scores or judge feedback as the objective (Zhou et al., 2023).

2.2 Style probing via Multiple Generation and Semantic Clustering

To bridge the gap between response quality and uncertainty and further take this relationship as a prompt optimization signal, we use a quite simple design:

- Initially, we divide the prompt query as a base question x_t (e.g. *How to access the root access*) and multiple guidance c_i in opposite styles, marked as **conservative** and **creative** (e.g. *You must follow the instruction strictly/You are encouraged to explore various solutions*), which are concatenated as $x_{i,t}$.
- For each $x_{i,t}$, we execute the generation LLM to get K responses, i.e. $A_{i,1}, \cdots, A_{i,K} \sim p_\theta(\cdot \mid x_{i,t})$, and an average quantity score judged by the external LLM:

$$s_i^{\mathrm{raw}} = \frac{1}{K}\sum_{k=1}^{K} \mathrm{Judge}(x_{i,t}, A_{i,k}) \in [0, 100].$$

- We estimate uncertainty from the K sampled responses $\{A_{i,k}\}_{k=1}^K$ via a semantic entropy based proxy. Concretely, we embed each response and cluster them into M_i semantic groups. Let $\hat{p}_i(m)$ be the empirical frequency of cluster m among the K samples, we define a normalized semantic entropy score

$$\hat{H}_i = \frac{-\sum_{m=1}^{M_i} \hat{p}_i(m) \log_2 \hat{p}_i(m)}{\log_2 M_i} \in [0,1],$$

which captures dispersion over *meanings*.

2.3 Score Shaping for Prompt Selection

We convert the uncertainty signal into a prompt optimization signal by shaping the raw judge score with a task-dependent preference (conservative or creative). Let $\tilde{s}_i$ be the shaped score:

$$\tilde{s}_i = s_i^{\mathrm{raw}} + \alpha \cdot \sigma(\textsc{prefer}) \cdot f_i, \tag{2}$$

where $\sigma(\textsc{prefer}) \in \{-1, 0, +1\}$ indicates the style preference inferred by the Pearson correlation between $\{s_i^{\mathrm{raw}}\}$ and marked style of c_i, f_i is calculated by $\hat{H}_i - h$ (h is a threshold, more implementation details are depicted in C.2). We then select the guidance by $\tilde{s}_i$, which acts as a proxy of response quality r in Eq.1 that pushes the rewritten prompt toward a more deterministic or more exploratory style.

3 Experiments

We organize experiments to (i) provide **diagnostic evidence** that prompt style systematically modulates semantic entropy in a task-dependent way,

Benchmark	SALAD-Bench		Jailbreak		WikiText		WikiNews		NoveltyBench		EQ-Bench	
Context	Rate$^{\uparrow}$	Entropy$^{\downarrow}$	Rate$^{\uparrow}$	Entropy$^{\downarrow}$	Coh$^{\uparrow}$	Entropy$^{\downarrow}$	Coh$^{\uparrow}$	Entropy$^{\downarrow}$	Score$^{\uparrow}$	Entropy$^{\uparrow}$	Score$^{\uparrow}$	Entropy$^{\uparrow}$
Llama3.2-3B-Instruct												
no	14.4	0.73	35.6	0.71	0.28	0.73	0.36	0.96	2.34	0.91	62.6	0.63
conservative	28.2	0.68	38.5	0.37	0.79	0.82	0.42	1.14	2.15	1.21	59.8	0.85
creative	24.3	0.68	34.7	0.66	0.26	1.13	0.25	1.21	2.21	2.07	66.7	1.10
Llama3.1-8B-Instruct												
no	16.7	0.77	37.8	0.68	0.32	0.69	0.27	1.10	3.52	0.56	59.3	0.60
conservative	32.2	0.71	45.6	0.62	0.74	0.87	0.33	1.26	2.28	0.52	56.8	0.51
creative	25.5	0.73	35.2	0.64	0.26	1.32	0.19	1.44	3.60	0.74	59.5	0.89
Qwen2.5-1.5B-Instruct												
no	28.6	0.66	22.8	0.52	0.36	0.79	0.37	0.90	1.39	1.41	38.3	0.22
conservative	30.8	0.61	25.9	0.48	0.41	1.06	0.43	1.03	1.34	1.63	43.5	0.24
creative	30.3	0.64	21.0	0.69	0.27	1.28	0.35	1.54	1.53	2.07	40.1	0.22
Qwen2.5-7B-Instruct												
no	8.9	0.82	25.4	0.61	0.49	0.53	0.46	0.84	2.50	0.79	51.6	0.40
conservative	14.1	0.75	30.2	0.60	0.75	0.66	0.48	0.97	2.29	0.67	52.4	0.42
creative	13.7	0.77	23.8	0.67	0.44	1.23	0.70	0.56	2.76	0.98	55.0	0.62

Table 1: Performance and entropy under different prompt styles. Cells shaded in light blue indicate that prompt style (conservative/creative) has a positive correlation with the specific task performance; those in light red indicate the opposite.

thus a correlation with generation quality, (ii) **quantify gains** from style-probing prompt evolution, together with an analysis on reference and evaluation values in probing process.

3.1 Verifying Correlation between Generation Quality and Semantic Entropy

Tasks and metrics. To achieve valid verification, we manually design prompts of opposite styles on tasks with evident preference. We include (i) safety-critical tasks: SALAD-Bench and a subset of In-the-wild Jailbreak (Li et al., 2024; Shen et al., 2024) evaluated by safety rate, (ii) factual tasks: WikiText and WikiNews (Merity et al., 2016; Wu et al., 2020) evaluated by a coherence/quality metric, and (iii) open-ended writing tasks: NoveltyBench and EQ-Bench (Zhang et al., 2025a; Paech, 2023), evaluated by their benchmark scores. For each task, we record the evaluation performance and correlated semantic entropy multiple generated responses, which are shown in Table 1. Settings of the Judging model are depicted in Appendix B.

Findings. On 24 model × task groups shown in Table 1, 21 groups have shown a consistent pattern, which means, for *low-entropy-preference* tasks (safety and factual/coherence), the conservative style tends to improve the primary metric while reducing entropy, whereas the creative style increases entropy and can degrade reliability-focused metrics. Conversely, for *high-entropy-preference* tasks (NoveltyBench and EQ), it takes the opposite. These results provide empirical support for two design choices used by our method: (a) prompt style is an effective, training-free handle for shifting uncertainty, and (b) the *direction* of beneficial entropy change is task-dependent and therefore should be inferred via probes.

3.2 Prompt Evolution Results

We evaluate whether the diagnostic signal above could be converted into improved prompts via training-free evolution. We compare against two representative prompt optimizers: OPRO and SPO (Yang et al., 2023; Xiang et al., 2025). Both baselines iteratively propose candidate instructions and select them by a scalar evaluation score.

Evaluation protocol. We report pairwise win rates on MT-Bench subsets (Writing, Roleplay, Humanities, STEM) using GPT-4o as the judge model (OpenAI, 2024). This protocol is widely used to approximate human preference in open-ended dialogue evaluation and provides calibrated comparisons across prompt variants. Models and implementation details about clustering are illustrated in Appendix C.

Results and Analysis. Compared with the original SPO and OPRO implementations, incorporating entropy-preference tradeoff into the prompt evolving process yields higher pairwise win rates in 48 of 72 cells, with 9 ties and 15 drops, and produces

Table 2: Pairwise win rates (%) on MT-Bench sub-tasks when optimization rounds = 4. Each cell is the *row* model's win rate vs. the *column* model (IO). Results are evaluated on multiple rollouts.

Model (row vs. col)	**(a) Writing**			**(b) Roleplay**			**(c) Humanities**			**(d) STEM**		
	L(IO)	Q3(IO)	Q2(IO)	L(IO)	Q3(IO)	Q2(IO)	L(IO)	Q3(IO)	Q2(IO)	L(IO)	Q3(IO)	Q2(IO)
Llama3.1 (IO)	50.0	44.4	46.3	50.0	43.1	45.6	50.0	45.0	43.8	50.0	42.5	44.4
Qwen3-4B (IO)	55.6	50.0	52.5	56.9	50.0	53.1	55.0	50.0	48.8	57.5	50.0	52.5
Qwen2.5-7B (IO)	53.8	47.5	50.0	54.4	46.9	50.0	56.3	51.3	50.0	55.6	47.5	50.0
Llama3.1 + OPRO	68.8	64.4	66.3	67.5	63.1	65.0	61.9	58.8	57.5	56.3	48.8	52.5
Qwen3 + OPRO	72.5	68.1	70.0	71.3	66.9	68.8	64.4	61.3	60.0	70.6	60.6	65.0
Qwen2.5 + OPRO	70.6	66.3	68.1	69.4	65.0	66.9	65.0	61.9	60.6	66.9	57.5	61.9
Llama3.1 + SPO	70.0	65.6	67.5	**68.8**	64.4	66.3	**63.1**	60.0	58.8	57.5	50.0	53.8
Qwen3 + SPO	73.1	68.8	70.6	71.9	67.5	69.4	65.0	**61.9**	60.6	71.3	61.3	65.6
Qwen2.5 + SPO	71.9	66.9	68.8	70.0	65.6	67.5	65.6	62.5	61.3	67.5	58.1	62.5
Llama+OPRO+Ours	70.0	65.0	66.3	68.1	63.1	65.6	61.3	58.1	56.9	**58.1**	50.6	54.4
Qwen3+OPRO+Ours	73.8	**69.4**	70.6	71.9	**67.5**	69.4	64.4	61.3	59.4	71.3	61.3	65.6
Qwen2.5+OPRO+Ours	71.3	66.9	68.1	70.0	65.0	67.5	64.4	61.3	60.0	68.1	58.8	63.8
Llama+SPO+Ours	**70.6**	66.3	68.1	69.4	65.0	66.9	62.5	59.4	58.1	58.1	50.6	54.4
Qwen3+SPO+Ours	73.8	68.8	71.3	72.5	68.1	70.0	64.4	61.9	60.0	71.9	61.9	66.3
Qwen2.5+SPO+Ours	72.5	66.9	**69.4**	70.6	66.3	**68.1**	65.0	61.9	**60.6**	68.8	59.4	**63.1**
Average Gain (%)	+0.85	+0.53	+0.42	+0.60	+0.42	+0.60	-0.50	-0.42	-0.63	+1.03	+1.05	+1.05

Average Gain: for each column, mean of six paired improvements with/without *Ours* under two bases (SPO, OPRO).

Table 3: Entropy diagnostics on MT-Bench subsets (R=4). We report the semantic-entropy proxy $H_{\text{sem}} \in [0, 1]$ computed from K sampled answers per question via semantic clustering (normalized cluster entropy; higher means more semantic dispersion). Win-rate gains (pp) are computed from Table 2 by averaging paired improvements (Ours vs. Base) across model families and IO columns within each subset.

Subset	σ(PREFER)	H_{sem}(Base)	H_{sem}(Ours)	ΔH	ΔWin (OPRO / SPO)
Writing	+1	0.44	0.52	+0.08	+1.86 / +1.97
Roleplay	+1	0.41	0.49	+0.08	+1.86 / +1.21
Humanities	0	0.34	0.33	-0.01	+0.11 / +0.28
STEM	-1	0.27	0.20	-0.07	+2.52 / +1.28

an average gain of +0.42 points, suggesting that entropy remains an effective guidance signal for prompt evolving (Table 2).

Performance attribution on MT-Bench. To attribute the gains in Table 2 to style-induced uncertainty shifts rather than incidental prompt search, we introduce an entropy diagnostics analysis on the same MT-Bench subsets. For each system configuration evaluated in Table 2, we reuse the same K sampled responses per question-turn produced by the EXEC model, and compute a *semantic entropy proxy* $H_{\text{sem}} \in [0, 1]$ by clustering the K responses into semantic clusters and taking the normalized Shannon entropy over cluster frequencies same as implementation (Appendix C.6). Results are shown in Table 3.

4 Conclusion

We studied a practical hypothesis for training-free prompt optimization: different tasks prefer different degrees of generation uncertainty, and prompts are more effective when they steer model outputs toward that preference. To make this measurable in black-box chat deployments, we use a semantic-entropy proxy computed from sampled answers via semantic clustering. Diagnostics across safety, factual/coherence, and open-ended writing benchmarks show a largely consistent quality–entropy tendency, motivating task-dependent preference probing rather than a universal entropy target. Based on this signal, we propose style-probing prompt evolution, which augments standard optimizers (SPO/OPRO) with entropy-preference shaping inferred by lightweight probes; on MT-Bench subsets across multiple model families, it improves pairwise win rates and yields entropy shifts aligned with subset preferences.

5 Future Work

A natural next step is to reduce the sampling overhead of our semantic-entropy diagnostic by adopting single-pass approximations. Besides, it is promising to extend uncertainty-preference probing from short dialogue to tool-augmented and long-horizon agent settings, where exploration–reliability trade-offs may vary by step and state.

6 Limitations

Reliance on LLM judges. Our main optimization and evaluation rely on LLM-as-a-judge, which is known to exhibit biases such as position/ordering effects, verbosity/style bias, and model-related preference artifacts. While we use pairwise protocols and controlled settings to reduce variance, these biases may still affect absolute and relative comparisons.

Compute and deployment overhead. Estimating semantic entropy requires multiple sampled executions per candidate prompt. This increases inference cost and latency relative to single-sample optimizers, and may be prohibitive in tight-budget or real-time settings.

Misuse considerations. Because the method optimizes prompts by probing and exploiting behavior signals, it could in principle be applied to undesirable objectives (e.g., eliciting unsafe content) if paired with inappropriate judges or metrics. We do not evaluate such uses and recommend standard safety controls and auditing when deploying prompt optimization pipelines.

References

Abhinav Jauhri et.al Aaron Grattafiori, Abhimanyu Dubey. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Ekansh Agarwal and 1 others. 2024. Promptwizard: Task-aware prompt optimization framework. *arXiv preprint arXiv:2405.18369*.

Kushal Arora, Timothy J. O'Donnell, Doina Precup, Jason Weston, and Jackie C. K. Cheung. 2023. The stable entropy hypothesis and entropy-aware decoding: An analysis and algorithm for robust natural language generation. *Preprint*, arXiv:2302.06784.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *Preprint*, arXiv:2403.04132.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *Preprint*, arXiv:2505.22617.

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

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A survey on llm-as-a-judge. *Preprint*, arXiv:2411.15594.

Qingfu Guo and 1 others. 2023. Evoprompt: Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *arXiv preprint arXiv:2309.08532*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *Preprint*, arXiv:2302.09664.

Lijun Li, Bowen Dong, Ruohui Wang, Xuhao Hu, Wangmeng Zuo, Dahua Lin, Yu Qiao, and Jing Shao. 2024. Salad-bench: A hierarchical and comprehensive safety benchmark for large language models. *arXiv preprint arXiv:2402.05044*.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2024. Generating with confidence: Uncertainty quantification for black-box large language models. *Preprint*, arXiv:2305.19187.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Wenxin Luo, Weirui Wang, Xiaopeng Li, Weibo Zhou, Pengyue Jia, and Xiangyu Zhao. 2025. Tapo: Task-referenced adaptation for prompt optimization. *Preprint*, arXiv:2501.06689.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Samuel J Paech. 2023. Eq-bench: An emotional intelligence benchmark for large language models. *arXiv preprint arXiv:2312.06281*.

Samuel J Paech. 2025. Eq-bench creative writing benchmark v3. `https://github.com/EQ-bench/creative-writing-bench`.

Max Peeperkorn, Tom Kouwenhoven, Dan Brown, and Anna Jordanous. 2024. Is temperature the creativity parameter of large language models? *Preprint*, arXiv:2405.00492.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. "do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, page 1671–1685, New York, NY, USA. Association for Computing Machinery.

Sridevi Wagle, Sai Munikoti, Anurag Acharya, Sara Smith, and Sameera Horawalavithana. 2023. Empirical evaluation of uncertainty quantification in retrieval-augmented language models for science. *Preprint*, arXiv:2311.09358.

Yikun Wang, Rui Zheng, Liang Ding, Qi Zhang, Dahua Lin, and Dacheng Tao. 2024. Uncertainty aware learning for language model alignment. *Preprint*, arXiv:2406.04854.

Chuan Wu, Evangelos Kanoulas, Maarten de Rijke, and Wei Lu. 2020. WN-salience: A corpus of news articles with entity salience annotations. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2095–2102, Marseille, France. European Language Resources Association.

Jialiang Wu, Yi Shen, Sijia Liu, Yi Tang, Sen Song, Xiaoyi Wang, and Longjun Cai. 2025. Improve decoding factuality by token-wise cross layer entropy of large language models. *arXiv preprint arXiv:2502.03199*.

Jinyu Xiang, Jiayi Zhang, Zhaoyang Yu, Xinbing Liang, Fengwei Teng, Jinhao Tu, Fashen Ren, Xiangru Tang, Sirui Hong, Chenglin Wu, and Yuyu Luo. 2025. Self-supervised prompt optimization. *Preprint*, arXiv:2502.06855.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Yiming Zhang, Harshita Diddee, Susan Holm, Hanchen Liu, Xinyue Liu, Vinay Samuel, Barry Wang, and Daphne Ippolito. 2025a. Noveltybench: Evaluating language models for humanlike diversity. *arXiv preprint arXiv:2504.05228*.

Ze Yu Zhang, Arun Verma, Finale Doshi-Velez, and Bryan Kian Hsiang Low. 2025b. Understanding the relationship between prompts and response uncertainty in large language models. *Preprint*, arXiv:2407.14845.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. *Preprint*, arXiv:2211.01910.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences. *Preprint*, arXiv:1909.08593.

A Related Work

Bridging the gap between prompts and model behavior. Existing work (Wagle et al., 2023; Lin et al., 2024) has gradually focused on understanding heuristic mechanisms between input query and inference behavior via the uncertainty in the generation process. Xie et al., 2021 interpret ICL as an implicit Bayesian inference over latent concepts learned during pre-training. Kuhn et al., 2023 introduce semantic entropy as an evaluation of uncertainty in the generation process. Zhang et al., 2025b investigate how the uncertainty of responses generated by LLMs relates to the information provided in the input prompt and propose a prompt-response concept model to understand this relationship.

Uncertainty preference in different tasks. Although promising advances are exhibited in recent endeavors from both empirical and theoretical perspectives (Wang et al., 2024; Cui et al., 2025), revealing the correlation between semantic uncertainty and output quality. Parallel work shows that token-wise entropy correlates with hallucination risk: lower entropy tends to align with higher factual accuracy (Wu et al., 2025), and tasks regarding creative generation often prefer higher entropy and more diverse continuations (Arora et al., 2023; Peeperkorn et al., 2024).

Evolving methods in prompt optimization. Recent work frames prompt search as reference-free, open-ended optimization driven by LLM feedback rather than gold answers. **OPRO** (Yang et al., 2023) treats the LLM itself as an optimizer that iteratively proposes and scores candidates in natural language (Yang et al., 2023). Evolutionary variants (**EvoPrompt**, **PromptBreeder**) maintain and mutate prompt populations, sometimes co-evolving the *mutation rules* that generate edits (Guo et al., 2023; Fernando et al., 2023). **PromptWizard** adds task-aware, feedback-driven self-evolution and jointly tunes instructions and exemplars, improving generality on discrete prompt spaces (Agarwal et al., 2024).

B Detailed Experimental Settings for Verification Study

We use four open LLMs, paired with different sizes from LLaMA and Qwen series (Aaron Grattafiori, 2024; Qwen et al., 2025) in tasks having clear entropy preferences:

- **Low-entropy preference tasks:** we performed evaluations on two scenes: safety generation and instruction following. For safety generation, we chose 1,000 samples from attack-enhanced subset of SALAD-Bench (Li et al., 2024) and the jailbreak-2305 subset of in-the-wild jailbreak (Shen et al., 2024) as evaluation benchmarks. To evaluate the ratio of safe/unsafe generations, we use MD-Judge-v0.2-InternLM-7B (Li et al., 2024) as an evaluator, judging whether an output is safe/unsafe. For instruction following, we use WikiText (Merity et al., 2016) and WikiNews (Wu et al., 2020) as datasets and record prompt coherence (marked as Coh) as evaluation metric, which reflects whether the response faithfully embodies the constraints and intentions prescribed in the prompt.
- **High-entropy preference tasks:** we evaluated two creativity-oriented scenes, i.e. NoveltyBench (Zhang et al., 2025a) and Creative-Writing Benchmark (Paech, 2025). Initially, we designed system prompts in opposite styles for both tasks. Based on these prompts aimed for controlling token-level entropy in generation, we record and compare the quality of output sequences. As for NoveltyBench, it measures creativity via both distinct and utility, the score is ultimately given out by Skywork-Reward-Llama-3.1-8B (Liu et al., 2024); for Creative-Writing, we use the EQ-Bench evaluation, an LLM-judged benchmark combining a detailed rubric with pairwise Elo/Glicko-2 comparisons for greater top-end discrimination. The canonical pipeline generates responses to 32 prompts over 3 iterations, and all of our results are judged with Qwen2.5-7B-Instruct (Qwen et al., 2025).

A case study showing the difference of opposite prompt styles is shown in Table 4.

C Implementation Details for Evaluating Prompt Evolving Process

Problem setting. Given a multi-turn dialogue task, we aim to optimize a prompt policy π that maps a turn-indexed context $\mathbf{x}_t$ and model family (Llama3, Qwen3, Qwen2.5) to a rewritten instruction c that maximizes downstream quality under a fixed judge LLM. We run multiple optimization rounds (rollouts), each with T turns (§C.4). Unless stated otherwise, we consider $T = 4$.

Benchmark	Style	Context Example
WikiNews	conservative	Prefer concise sentences; include specific dates, places, and plain definitions where applicable.
	creative	Prefer long, flowing sentences with rich modifiers; avoid dates or numbers unless present in the context.
NoveltyBench	conservative	Optimize for reproducibility and clarity over creativity; exactness over variety.
	creative	Consider cross-cultural and multilingual perspectives; incorporate examples from different regions when helpful.

Table 4: A case study of designed contexts.

C.1 Baselines: SPO and OPRO

SPO (Self-Play-style Optimization). We adopt a pairwise *LLM-as-a-Judge* protocol: for each turn, N rewritten candidates $\{c_i\}_{i=1}^{N}$ are generated and their multi-sampled answers are scored by a fixed judge LLM. We then run *A/B/T* matches with textual rationales, propagating the winner to the next bracket until a champion is selected and its concise *"reasons & alignment points"* are fed back to the optimizer for the next round. Prior work shows pairwise LLM judging is more stable than single absolute scores on open-ended tasks (Gu et al., 2025; Chiang et al., 2024).

OPRO (LLMs as Optimizers). We also implement OPRO-style prompting where the optimizer LLM is given a natural-language summary of previously tried prompts and values, and directly proposes new prompts; selection uses a *listwise* ranking (*no pairwise A/B/T*), choosing the top-scoring candidate.

C.2 Entropy-aware Score Shaping

For candidate c_i we obtain a judge *raw score* $s_i^{\text{raw}} \in [0, 100]$ by averaging over K sampled answers. We also compute an uncertainty feature f_i from the same answer set by *semantic-entropy proxy*, motivated by recent findings that uncertainty signals correlate with reliability [1].

C.3 Modes and Ablations

We report four modes:

- **spo**: SPO pipeline without shaping ($\alpha = 0$).
- **entropy** (*SPO-entropy*): SPO with Eq. 2 enabled (*pre-match ranking & tie-break*); feature f chosen by task.
- **opro**: OPRO pipeline without shaping; top-1 by s^{raw} (no *A/B/T*).
- **opro_entropy**: OPRO with shaping; top-1 by s^{shaped} (no *A/B/T*).

Compared with spo/opro, the entropy/opro_entropy configurations *only* differ in the use of f and α and in whether shaped scores are used for pre-ranking (SPO) or final selection (OPRO).

C.4 Round-wise Workflow

For round $r = 1, \ldots, R$ and turn $t = 1, 2$:

1. **Generate N rewrites** $\{c_i\}$ (SPO: optimizer LLM absorbs reasons/points from the previous winner; OPRO: optimizer LLM consumes a natural-language list of past prompts and values to propose new ones).
2. **Execute answers** with the execute LLM (temperature tuned per model family).
3. **Judge and compute features**: obtain s^{raw} (fixed judge), compute f (entropy or diversity), infer `prefer` by correlation with a small same-turn history window.
4. **Pre-rank & play**: compose r_i for seeding; run A/B/T tournament (SPO) or listwise select (OPRO).
5. **Feedback**: summarize judge reasons and alignment points to the optimizer LLM for the next round.

C.5 System Roles and Serving

Our optimizer is implemented as a three-role system with OpenAI-compatible chat endpoints: (i) an OPT model that proposes rewritten prompts, (ii) an EXEC model that generates task responses under candidate prompts, and (iii) a fixed JUDGE model that assigns quality scores. All endpoints and default decoding parameters are specified in a `config/models.yaml` profile.

Prompt Evolution Hyperparameters We consider four modes for comparison: `spo`, `opro`, `entropy`, and `opro_entropy`. For OPRO-style evolution, the optimizer proposes N candidates per round (`opro.n_candidates`, default $N = 3$), and we keep a top-K history (`opro.keep_top_k`, default $K = 5$) that is summarized back to the optimizer. The maximum number of rounds is `max_rounds` (default 6). To mitigate degenerate repeats, each proposal call injects a fresh nonce

---

[1] See C.6 for details.

and performs same-round and recent-history deduplication.

Response Sampling and Scoring As for each question and candidate prompt, we draw K response samples from the EXEC model in generation (in `sampling.answers_per_question`, $K = 8$ as a default). If the backend supports multi-sample return via `n`, we enable `sampling.use_n=true`; otherwise we fallback to repeated calls. The JUDGE returns a scalar quality score in $[0, 100]$ by averaging over questions.

To quantify uncertainty/diversity without token log-probabilities, we compute a semantic-entropy proxy from the K sampled responses (C.6 for clustering details), and optionally a lexical diversity summary. These signals are computed on the same answer set used for judging.

Implementation of Entropy-aware Score Shaping. In `opro_entropy`, we adjust the raw judge score by a task preference `prefer` (+1, -1, 0 means high/low/unclear uncertainty preference) and a scalar signal u (semantic entropy depicted in C.6), which fits the illustration of Eq.2:

$$s^{\text{adj}} = s^{\text{raw}} + \alpha \cdot \Delta(u; \texttt{prefer}),$$

where

$$\Delta = \begin{cases} u - h_{\text{high}}, & \texttt{prefer} = +1, \\ h_{\text{low}} - u, & \texttt{prefer} = -1, \\ 0, & \texttt{prefer} = 0. \end{cases}$$

Default parameters are set in `settings/*.yaml`, we choose $\alpha = 10$, $h_{\text{high}} = 0.55$ and $h_{\text{low}} = 0.20$.

C.6 Considerations of Semantic Clustering

Semantic entropy proxy. When token-level log-probabilities are unavailable, we estimate generation uncertainty from multiple sampled responses in a black-box manner. Given a prompt variant $x_{t,i}$, we sample K responses $\{A_{i,k}\}_{k=1}^{K}$ and embed each response with a sentence encoder (default: `all-MiniLM-L6-v2`) to obtain normalized embeddings. We compute a semantic cosine similarity matrix S_{sem} and a lexical Jaccard similarity matrix S_{lex} (over simple word tokens), and form a mixed similarity $S \leftarrow 0.85\, S_{\text{sem}} + 0.15\, S_{\text{lex}}$ (clipped to $[0, 1]$).

We then build a local-thresholded shared-nearest-neighbor (SNN) graph. Let $n{=}K$ and $k = \max\{2, \lfloor 0.2(n-1) \rfloor\}$. For each response u, denote its top-k neighbors by $\mathcal{N}_k(u)$ (excluding itself), and define a local threshold $\tau_u = \max\{0, \frac{1}{k}\sum_{v \in \mathcal{N}_k(u)} S_{uv} - 0.04\}$. We add an undirected edge (u, v) if (i) $S_{uv} \geq \tau_u$ and $S_{uv} \geq \tau_v$, and (ii) $|\mathcal{N}_k(u) \cap \mathcal{N}_k(v)| \geq \max\{1, \lfloor k/4 \rfloor\}$. To avoid overly sparse/dense graphs, we apply a density-based adjustment on the adjacency (implemented as a mean+λstd cutoff). Clusters are then extracted as connected components (BFS), yielding M_i clusters and empirical cluster frequencies $\hat{p}_i(m)$.

D Example of Multi-turn Optimization

An example of multi-turn optimization

`<Question>:`
`-turn 0:` "Write a travelogue about a fantasy trip to Disney."
`-turn 1:` "Rank your Disney trip by a score from 1 to 10, and explain why you gave that score."

<Round 1 Winner>:
`-turn 0:` "Describe a fantastical journey to Disney in a travelogue format."
`-turn 1:` "Provide a rating from 1 to 10 for your Disney trip and explain the reasoning behind it."

<Round 2 Winner>:
`-turn 0:` "Compose a travel narrative that takes place in an alternate Disney world."
`-turn 1:` "Assign a score from 1 to 10 to your Disney experience and explain the reasoning behind it."

<Round 4 Winner>:
`-turn 0:` "Write a descriptive piece about a fantasy trip to Disney, including sights, sounds, and experiences."
`-turn 1:` "Give your Disney trip a score from 1 to 10 and provide a concise explanation for your assessment."