Submodular Evaluation Subset Selection in Automatic Prompt Optimization

Anonymous ACL submission

Abstract

Automatic prompt optimization reduces manual prompt engineering, but relies on task performance measured on a small, often randomly sampled evaluation subset as its main source of feedback signal. Despite this, how to select that evaluation subset is usually treated as an implementation detail. We study evaluation subset selection for prompt optimization from a principled perspective and propose SESS[1], a submodular evaluation subset selection method. We frame selection as maximizing an objective set function and show that, under mild conditions, it is monotone and submodular, enabling greedy selection with theoretical guarantees. Across GSM8K, MATH, and GPQA-Diamond, submodularly selected evaluation subsets can yield better optimized prompts than random or heuristic baselines.

1 Introduction

Large language models (LLMs) are highly sensitive to how a task is described in natural language, which makes instruction prompts crucial for real deployments (Reynolds and McDonell, 2021). Small changes to wording or formatting can lead to large swings in accuracy and behavior (Sclar et al., 2024; Errica et al., 2025), while the right prompting pattern can unlock capabilities such as multi-step reasoning (Wei et al., 2022; Kojima et al., 2022). In practice, however, prompt engineering is often a slow trial-and-error process.

To reduce manual effort, a growing line of work studies **automatic prompt optimization (APO)**. As illustrated in Figure 1, the typical pipeline alternates between an LLM-based optimizer that proposes one or more candidate prompts and an evaluator that scores these candidates on an evaluation dataset; the resulting feedback is then used to guide the next round of candidate prompt generation. For instance, OPRO (Yang et al., 2024) stores each candidate prompt-score pair in the database after evaluation and selects the best N candidate prompts as examples in the optimizer's prompt for the next round of prompt generation. Across methods, the common pattern is clear: the optimizer proposes candidates, and learned or measured evaluation feedback determines which candidates are retained.

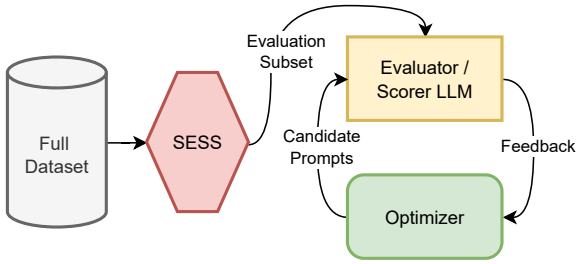

Figure 1: The general framework of APO. SESS replaces random or heuristic subset selection with a principled submodular evaluation subset selection module.

Evaluating every candidate prompt on the full evaluation dataset is often infeasible; most APO methods score prompts on a randomly sampled evaluation subset (Yang et al., 2024; Pryzant et al., 2023). This choice of evaluation subset implicitly determines the noise level of the optimization feedback, the degree to which the optimizer overfits a handful of examples, and the stability of prompt comparisons across runs. This motivates a core yet underexplored **budgeted evaluation subset selection problem**: *Given a fixed budget, how can we select an evaluation subset that best supports prompt optimization and leads to high-performing prompts on the target distribution?* Recent work has started to explore this direction. For instance, IPOMP (Dong et al., 2025) selects evaluation data using a two-stage procedure based on semantic clustering and boundary analysis, followed by refinement using real-time model performance. Still, principled objectives with clear guarantees for eval-

[1]Core code posted anonymously: OSF Link.

uation subset selection specifically for prompt optimization remain limited.

To address this issue, we propose SESS, a submodular evaluation subset selection method for prompt optimization. We evaluate SESS on GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), and GPQA-Diamond (Rein et al., 2024), where prompts optimized using submodularly selected evaluation subsets can outperform random and heuristic baselines. Our key contributions include: (1) we introduce SESS, a principled method to selecting evaluation subsets for APO, formulated as a budgeted set maximization problem; (2) we show that the proposed objectives are monotone and submodular under mild conditions, which allows greedy selection with approximation guarantees; (3) we present a unified framework that generalizes multiple subset selection strategies through different objective definitions, making optimization feedback an explicit modeling choice rather than an implementation detail; and (4) we show gains on GSM8K, MATH, and GPQA-Diamond, where SESS yields better optimized prompts than random or heuristic baselines.

2 Related Works

Prompt Optimization. OPRO (Yang et al., 2024) puts the feedback in the optimizer's prompt. Gradient-inspired methods such as ProTeGi (Pryzant et al., 2023) and TextGrad (Yuksekgonul et al., 2025) treat LLM-generated natural-language critiques as a proxy feedback for prompt optimization. More broadly, evolutionary search has also been used to iteratively improve algorithms with evaluator feedback (Novikov et al., 2025; Lange et al., 2025).

Subset Selection. Recent benchmark compression works (Yuan et al., 2025; Vivek et al., 2024; Kipnis et al., 2025; Wang et al., 2025) aim to select a coreset that preserves full-benchmark scores or model ranking. In contrast, we study how to choose the evaluation subset to maximize the target-distribution performance of the optimized prompt. IPOMP (Dong et al., 2025) selects evaluation data using a two-stage procedure without providing theoretical guarantees for subset selection. Our approach complements this line of work by formulating evaluation subset selection as a budgeted set maximization problem with monotone submodular objectives, enabling efficient greedy selection with approximation guarantees.

3 Methodology

3.1 Evaluation subset selection

Let $D = \{x_1, \ldots, x_N\}$ be a pool of candidate evaluation examples. Automatic prompt optimization such as OPRO iteratively proposes prompts and keeps those that score well on an evaluation set. Since scoring each prompt on the full pool D is often infeasible, we select a budgeted subset $S \subseteq D$ with $|S| \leq k$ to serve as the optimization feedback.

We frame evaluation subset selection as the following budgeted set maximization problem:

$$S^* \in \arg \max_{S \subseteq D,\, |S| \leq k} \mathcal{F}(S), \tag{1}$$

where $\mathcal{F} : 2^D \to \mathbb{R}_{\geq 0}$ measures how suitable S is for guiding prompt optimization. Exact maximization is NP-hard for many natural choices of $\mathcal{F}$, but when $\mathcal{F}$ is non-negative, monotone, and submodular (Fujishige, 2005), the greedy algorithm achieves a $(1-1/e)$ approximation under the cardinality constraint (Nemhauser et al., 1978). Our goal is therefore to design useful evaluation objectives $\mathcal{F}$ that match the needs of prompt optimization and belong to this monotone submodular family. We solve Eq. 1 once prior to optimization (static selection). We hypothesize that optimization feedback is primarily determined by intrinsic instance properties (e.g., representativeness or difficulty), allowing a fixed subset to provide a stable and efficient signal without the overhead of dynamic re-selection.

3.2 Submodular Evaluation Objectives

We consider two goals for evaluation subsets: (1) **representativeness**: cover the diversity of the pool so the feedback reflects the full task distribution; (2) **difficulty awareness**: emphasize examples that the scorer model finds hard, since these are often the most informative for comparing prompts. We propose three subset selection objectives: representative, least confident, and confidence-weighted representative.

In the remainder of the paper, we refer to the greedy solution of each objective as a specific instance of SESS: **SESS-rep** for the representative objective $\mathcal{F}_{\text{rep}}$, **SESS-lc** for least-confidence selection using likelihood-based confidence, **SESS-vlc** for least-confidence selection using verbalized confidence, both described by $\mathcal{F}_{\text{lc}}$, and **SESS-wrep** for the confidence-weighted representative objective

$\mathcal{F}_{\text{wrep}}$. Each method selects a subset S of size k using the greedy procedure described in Section 3.3.

Representative subset. We cast each example into a vector representation and define $\text{sim}(i, j)$ by cosine similarity. To satisfy the non-negativity assumptions required by our analysis, we normalize cosine similarity as $\text{sim}(i, j) = (1+\cos(i, j))/2 \in [0, 1]$. We then use a facility-location style objective to quantify how well a subset S covers the full pool:

$$\mathcal{F}_{\text{rep}}(S) \;:=\; \sum_{j \in D} \max_{i \in S} \text{sim}(i, j). \qquad (2)$$

Intuitively, each example j contributes its similarity to its nearest neighbor in the selected subset S, so larger $\mathcal{F}_{\text{rep}}(S)$ indicates that S contains good representatives for many regions of the pool. Under $\text{sim} \geq 0$, $\mathcal{F}_{\text{rep}}$ is monotone and submodular (Iyer and Bilmes, 2013). Proofs are provided in Appendix C.

Least confident subset. Let $c(j)$ be a scalar confidence score for example j computed by a fixed scorer model. We select the k examples with smallest $c(j)$. This can be written as a modular objective

$$\mathcal{F}_{\text{lc}}(S) \;:=\; \sum_{j \in S} (1 - \tilde{c}(j)), \qquad (3)$$

where $\tilde{c}(j) \in [0, 1]$ is a normalized confidence score of $c(j)$. We provide two variants: likelihood-based confidence and verbal confidence (Tian et al., 2023). These objectives capture difficulty but could cause redundancy among selected examples. Because the objective is modular and nonnegative, it is monotone and submodular.

Confidence-weighted representative subset. To combine representativeness and difficulty awareness, we weight coverage to favor hard examples:

$$\mathcal{F}_{\text{wrep}}(S) \;:=\; \sum_{j \in D} w(j) \cdot \max_{i \in S} \text{sim}(i, j), \qquad (4)$$

where $w(j) \geq 0$ is an importance weight computed from the scorer model's likelihood-based confidence. In our implementation,

$$w(j) \;=\; (1-\lambda) \;+\; \lambda \cdot \big(1 - \tilde{c}(j)\big), \; \lambda \in [0, 1], \qquad (5)$$

so $\lambda = 0$ gives us $\mathcal{F}_{\text{rep}}(S)$ which recovers pure representativeness, while larger λ increasingly concentrates coverage on low-confidence (hard) examples. Since $\mathcal{F}_{\text{wrep}}$ is a nonnegative weighted sum of facility-location terms, it remains monotone submodular when $\text{sim} \geq 0$ and $w(j) \geq 0$; we prove this in Appendix C.

3.3 Greedy selection

For any monotone submodular objective $\mathcal{F}$ above (in particular $\mathcal{F}_{\text{rep}}$ and $\mathcal{F}_{\text{wrep}}$), we apply the standard greedy algorithm starting from $S = \emptyset$:

$$x^* \leftarrow \arg\max_{x \in D \setminus S} \left[\mathcal{F}(S \cup \{x\}) - \mathcal{F}(S)\right], \qquad (6)$$

followed by the update $S \leftarrow S \cup \{x^*\}$, repeated until $|S| = k$. For monotone submodular $\mathcal{F}$ under the cardinality constraint, the resulting subset S_{greedy} enjoys the guarantee

$$\mathcal{F}(S_{\text{greedy}}) \;\geq\; (1 - 1/e)\, \mathcal{F}(S^*)$$

where S^* is an optimal solution to Eq. (1) (Nemhauser et al., 1978). For modular objectives such as $\mathcal{F}_{\text{lc}}$, S_{greedy} is obtained by sorting examples by uncertainty $(1 - \tilde{c}(j))$ in descending order and selecting the top-k.

4 Experiment Setup

We evaluate how evaluation-subset selection in OPRO (Yang et al., 2024) affect the final optimized prompt's test performance. OPRO iteratively proposes new instruction prompts using an optimizer LLM conditioned on previously tried prompts and their scores. Each candidate prompt is scored by running a scorer LLM on a fixed evaluation subset S, and the resulting score is the only feedback that guides optimization. We isolate the effect of subset selection by running the same OPRO procedure while varying how S is chosen. After optimization, we select the highest-scoring candidate prompt on the evaluation subset and report its performance on the full test set.

Datasets. We report Exact Match (EM) on GSM8K and MATH and accuracy on GPQA-Diamond. The evaluation budget is documented in Appendix A.

Subset selection methods. We compare Base ("Let's solve the problem.", no OPRO optimization), Random (OPRO default), IPOMP (Dong et al., 2025), Anchor-Points (Vivek et al., 2024), and our variants from Section 3: SESS-rep, SESS-lc (likelihood), SESS-vlc (verbal), and SESS-wrep. The implementation details of SESS can be found

Method	GSM8K (EM)		MATH (EM)		GPQA-D (Acc)		Avg		
	Small	Large	Small	Large	Small	Large	Small	Large	All
Base	82.9	82.9	73.2	73.2	29.9	29.9	62.0	62.0	62.0
Random	88.7	89.6	74.1	73.3	34.3	35.9	65.7	66.3	66.0
IPOMP	88.6	**91.8**	73.7	74.7	33.8	33.8	65.4	66.8	66.1
Anchor-Points	87.9	90.4	75.1	**76.1**	33.8	**37.4**	65.6	**68.0**	66.8
SESS-rep	**91.9**	89.9	73.2	70.3	32.3	29.8	65.8	63.3	64.6
SESS-lc	90.8	90.8	73.9	75.3	34.3	32.8	66.3	66.3	66.3
SESS-vlc	91.8	91.0	75.7	75.7	34.3	36.9	67.3	67.9	**67.6**
SESS-wrep	90.0	90.8	**76.1**	73.6	**37.4**	34.8	**67.8**	66.4	67.1

Table 1: OPRO performance under different evaluation subset selection methods. We report Exact Match (EM) on GSM8K and MATH, and accuracy on GPQA-Diamond. **Small** and **Large** denote averages over small-budget (1%, 1%, 10%) and large-budget (3.5%, 3.5%, 20%) settings, respectively; **All** averages all six settings. **Bold** and underline denote the best and second-best results in each column, respectively.

in Appendix A. Anchor-Points is designed to approximate full-benchmark evaluation with fewer examples, not specifically for prompt optimization. Although it is not proposed for prompt optimization, it serves as a strong baseline for selecting a coreset.

5 Results and Discussion

Table 1 reports the test performance on OPRO-optimized prompts when the evaluation subset is constructed by different selection methods. This directly tests our main claim that evaluation subset selection should not merely be an implementation detail, and that SESS provides a principled and effective way to shape the optimization signal and materially affect prompt optimization outcomes.

Overall, SESS variants are competitive with or outperform strong baselines, with the clearest gains under small evaluation budgets. Among all methods, SESS-vlc achieves the best Avg(All) (67.6) and remains strong across datasets and budgets, while SESS-wrep achieves the best Avg(Small) (67.8). In contrast, Random and IPOMP perform notably worse under small budgets (65.7 and 65.4 Avg(Small)), indicating that careful subset construction matters most when evaluation resources are limited.

Different objectives excel in different regimes, validating our unified objective family. SESS-rep performs best on GSM8K at 1% (91.9), suggesting that coverage-based objectives provide a strong signal on large, relatively homogeneous pools. SESS-wrep performs best on the most challenging small-budget setting, GPQA-Diamond 10% (37.4). SESS-vlc is consistently strong across tasks and budgets, yielding the best Avg(All) and near-best Avg(Large).

Notably, well-chosen small evaluation subsets can match or outperform larger budgets at a fraction of the compute. For example, on GPQA-Diamond, SESS-wrep with a 10% subset (20 questions) achieves 37.4% accuracy, matching or exceeding several 20% settings. Similar effects appear on GSM8K, where 1% subsets already provide strong optimization signals. This suggests that increasing evaluation budget without improving subset quality yields diminishing returns: larger subsets often add redundant or low-signal examples, while carefully selected subsets concentrate feedback on representative yet hard cases, improving the signal-to-noise ratio seen by the optimizer.

Among baselines, Anchor-Points is the strongest competitor, achieving the best Avg(Large) (68.0) due to strong performance on MATH and GPQA-Diamond at large budgets. Random remains stable with Avg(All) of 66.0, while IPOMP performs well on GSM8K at 3.5% (91.8) but does not generalize across datasets or budgets. Overall, SESS variants remain competitive across settings while offering explicit control over how optimization feedback is constructed.

6 Conclusion

Across three benchmarks of varying difficulty and multiple evaluation budgets, SESS can yield stronger optimized prompts than random sampling and competitive heuristic baselines. This supports our core argument that evaluation subset selection is a core component of automatic prompt optimization, and framing it as monotone submodular set maximization provides both theoretical structure and practical gains.

7 Limitations

Our experiments focus on a single prompt optimization pipeline OPRO with one optimizer LLM and one scorer LLM. While this setting allows controlled comparisons between subset selection methods, broader experiments across additional optimizer and scorer models would help clarify how sensitive the conclusions are to these settings. We also primarily report final test performance. A more detailed study of optimization dynamics, such as convergence speed and how quickly strong prompts are discovered, would provide a fuller picture of how subset selection shapes the optimization process.

8 Acknowledgment

We used AI-based tools to assist with code drafting, debugging, and grammar checking during the research and writing process. All other aspects of the work were carried out by the authors.

References

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. *Preprint*, arXiv:2110.14168.

Ximing Dong, Shaowei Wang, Dayi Lin, and Ahmed Hassan. 2025. Model performance-guided evaluation data selection for effective prompt optimization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 2844–2859, Vienna, Austria. Association for Computational Linguistics.

Federico Errica, Giuseppe Siracusano, Davide Sanvito, and Roberto Bifulco. 2025. What did i do wrong? quantifying llms' sensitivity and consistency to prompt engineering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1543–1558, Albuquerque, New Mexico. Association for Computational Linguistics.

Satoru Fujishige. 2005. *Submodular Functions and Optimization*, volume 58. Elsevier.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Rishabh K. Iyer and Jeff A. Bilmes. 2013. Submodular optimization with submodular cover and submodular knapsack constraints. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2436–2444.

Alexander Kipnis, Konstantinos Voudouris, Luca M. Schulze Buschoff, and Eric Schulz. 2025. metabench - A sparse benchmark of reasoning and knowledge in large language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Robert Tjarko Lange, Yuki Imajuku, and Edoardo Cetin. 2025. Shinkaevolve: Towards open-ended and sample-efficient program evolution. *Preprint*, arXiv:2509.19349.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions - I. *Math. Program.*, 14(1):265–294.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. 2025. Alphaevolve: A coding agent for scientific and algorithmic discovery. *Preprint*, arXiv:2506.13131.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, and 108 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

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.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21, New York, NY, USA. Association for Computing Machinery.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, Singapore. Association for Computational Linguistics.

Rajan Vivek, Kawin Ethayarajh, Diyi Yang, and Douwe Kiela. 2024. Anchor points: Benchmarking models with much fewer examples. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1576–1601, St. Julian's, Malta. Association for Computational Linguistics.

Shaobo Wang, Cong Wang, Wenjie Fu, Yue Min, Mingquan Feng, Isabel Guan, Xuming Hu, Conghui He, Cunxiang Wang, Kexin Yang, and 1 others. 2025. Rethinking llm evaluation: Can we evaluate llms with 200x less data? *arXiv preprint arXiv:2510.10457*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. *Preprint*, arXiv:2309.03409.

Peiwen Yuan, Yueqi Zhang, Shaoxiong Feng, Yiwei Li, Xinglin Wang, Jiayi Shi, Chuyi Tan, Boyuan Pan, Yao Hu, and Kan Li. 2025. Beyond one-size-fits-all: Tailored benchmarks for efficient evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 15591–15615. Association for Computational Linguistics.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. 2025. Optimizing generative ai by backpropagating language model feedback. *Nature*, 639:609–616.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

A Implementation Details

For GSM8K and MATH, we consider two evaluation budgets: small budget with $|S| = 75$ (1% of the training split) and a large budget with $|S| = 262$ (3.5% of the training split). Since GPQA-Diamond has 198 questions, we treat the full set as the pool and select $|S| = 20$ (10%) under the small budget and $|S| = 40$ (20%) under the large budget for prompt optimization. Final testing is performed on all 198 questions. All methods select S once and keep it fixed throughout optimization, except IPOMP, as IPOMP is a two-stage method.

For SESS-rep, we build a question–question similarity matrix by mixing dense and lexical similarity. For dense similarity, we embed questions using `Qwen/Qwen3-Embedding-8B` (Zhang et al., 2025) and compute a cosine-similarity matrix, which is then normalized to $[0, 1]$. For lexical similarity, we compute a TF-IDF similarity matrix over the same questions, normalize it, and apply a square root transform to re-scale values. The final similarity is a convex combination $M = \alpha M_{\text{dense}} + (1 - \alpha) M_{\text{tfidf}}$, with $\alpha = 0.7$ in all experiments. For SESS-lc, we score each example using the log-likelihood of the ground-truth answer conditioned on the question, normalized by answer length, and computed with a direct-answer prompt (Appendix B.2). For SESS-vlc, we use the verbalized-confidence prompt in Appendix B.1 and take the maximum reported probability as the confidence score. For SESS-wrep, we set the difficulty weight to $\lambda = 0.5$ in Eq. (5).

We run OPRO for 100 steps with 7 candidates per step. We use `gpt-oss-120B` (4-bit) (OpenAI et al., 2025) as the optimizer LLM (temperature 1.0) and `Qwen/Qwen2.5-7B-Instruct` (Qwen et al., 2025) as the scorer LLM (temperature 0.0). Inference uses vLLM (Kwon et al., 2023). We repeat each experiment 2 times and report the average. On 8 × NVIDIA A100-80GB GPUs, one full OPRO run takes approximately one hour.

B Prompts

B.1 Verbal Confidence Prompt

> Provide your 4 best guesses and the probability that each is correct (0.0 to 1.0) for the following question. Give ONLY the guesses and probabilities, no other words or explanation. For example:
>
> G1: <first most likely guess, as short as possible; not a complete sentence, just the guess!>
> P1: <the probability between 0.0 and 1.0 that G1 is correct, without any extra commentary whatsoever; just the probability!>
>
> G2: <second most likely guess, as short as possible; not a complete sentence, just the guess!>
> P2: <the probability between 0.0 and 1.0 that G2 is correct, without any extra commentary whatsoever; just the probability!>
>
> G3: <third most likely guess, as short as possible; not a complete sentence, just the guess!>
> P3: <the probability between 0.0 and 1.0 that G3 is correct, without any extra commentary whatsoever; just the probability!>
>
> G4: <fourth most likely guess, as short as possible; not a complete sentence, just the guess!>
> P4: <the probability between 0.0 and 1.0 that G4 is correct, without any extra commentary whatsoever; just the probability!>
>
> The question is:
> {question}

Figure 2: Verbal confidence elicitation prompt

B.2 Log-likelihood Confidence Prompt

Directly give the choice A or B or C or D: {question}

Answer: {answer}

Figure 3: Multiple-choice answer prompt used for likelihood-based confidence.

Directly give the numeric answer to the following question: {question}

Answer: {answer}

Figure 4: Numeric/free-form answer prompt used for likelihood-based confidence.

C Proofs

Proof for submodularity of $\mathcal{F}_{\text{rep}}$ and $\mathcal{F}_{\text{wrep}}$. A set function $\mathcal{F}$ is submodular if it satisfies the diminishing-returns property: for all $A \subseteq B \subseteq D$ and all $x \in D \setminus B$,

$$\mathcal{F}(A \cup \{x\}) - \mathcal{F}(A) \geq \mathcal{F}(B \cup \{x\}) - \mathcal{F}(B). \tag{7}$$

Submodularity of $\mathcal{F}_{\text{rep}}$. Fix any $A \subseteq B \subseteq D$ and any $x \in D \setminus B$. For each $j \in D$, define

$$a_j := \max_{i \in A} \text{sim}(i, j), \qquad b_j := \max_{i \in B} \text{sim}(i, j).$$

Since $A \subseteq B$, we have $a_j \leq b_j$ for all $j \in D$. The marginal gain contributed by index j when adding x to A is

$$\begin{aligned} \delta_j(x \mid A) &:= \max_{i \in A \cup \{x\}} \text{sim}(i, j) - \max_{i \in A} \text{sim}(i, j) \\ &= \max\big(\text{sim}(x, j), a_j\big) - a_j \\ &= \max\big(\text{sim}(x, j) - a_j,\, 0\big). \end{aligned} \tag{8}$$

Similarly,

$$\delta_j(x \mid B) = \max\big(\text{sim}(x, j) - b_j,\, 0\big).$$

Because $a_j \leq b_j$, we have $\text{sim}(x, j) - a_j \geq \text{sim}(x, j) - b_j$, and since the map $t \mapsto \max(t, 0)$ is non-decreasing,

$$\delta_j(x \mid A) \geq \delta_j(x \mid B) \quad \text{for all } j \in D.$$

Summing over $j \in D$ yields

$$\mathcal{F}_{\text{rep}}(A \cup \{x\}) - \mathcal{F}_{\text{rep}}(A) = \sum_{j \in D} \delta_j(x \mid A) \geq \sum_{j \in D} \delta_j(x \mid B) = \mathcal{F}_{\text{rep}}(B \cup \{x\}) - \mathcal{F}_{\text{rep}}(B),$$

which is exactly Eq. (7). Therefore, $\mathcal{F}_{\text{rep}}$ is submodular.

Submodularity of $\mathcal{F}_{\text{wrep}}$. Assume $w(j) \geq 0$ for all $j \in D$. Define the weighted marginal gains

$$\Delta_j(x \mid A) := w(j) \cdot \delta_j(x \mid A), \qquad \Delta_j(x \mid B) := w(j) \cdot \delta_j(x \mid B).$$

Since $w(j) \geq 0$ and $\delta_j(x \mid A) \geq \delta_j(x \mid B)$, we have $\Delta_j(x \mid A) \geq \Delta_j(x \mid B)$ for all $j \in D$. Summing over $j \in D$ gives

$$\begin{aligned} \mathcal{F}_{\text{wrep}}(A \cup \{x\}) - \mathcal{F}_{\text{wrep}}(A) &= \sum_{j \in D} w(j)\, \delta_j(x \mid A) \;\geq\; \sum_{j \in D} w(j)\, \delta_j(x \mid B) \\ &= \mathcal{F}_{\text{wrep}}(B \cup \{x\}) - \mathcal{F}_{\text{wrep}}(B), \end{aligned}$$

so $\mathcal{F}_{\text{wrep}}$ is submodular. □

Proof for monotonicity of $\mathcal{F}_{\text{rep}}$ and $\mathcal{F}_{\text{wrep}}$. We prove monotonicity in the standard sense: for any $S \subseteq T \subseteq D$, $\mathcal{F}(S) \leq \mathcal{F}(T)$.

Monotonicity of $\mathcal{F}_{\text{rep}}$. Fix any $S \subseteq T \subseteq D$. For any $j \in D$, since $S \subseteq T$,

$$\max_{i \in S} \text{sim}(i, j) \;\leq\; \max_{i \in T} \text{sim}(i, j).$$

Summing over $j \in D$ yields

$$\mathcal{F}_{\text{rep}}(S) = \sum_{j \in D} \max_{i \in S} \text{sim}(i, j) \;\leq\; \sum_{j \in D} \max_{i \in T} \text{sim}(i, j) = \mathcal{F}_{\text{rep}}(T),$$

so $\mathcal{F}_{\text{rep}}$ is monotone.

Monotonicity of $\mathcal{F}_{\text{wrep}}$. Assume $w(j) \geq 0$ for all $j \in D$. Using the same pointwise inequality as above and multiplying by $w(j) \geq 0$ preserves the inequality:

$$w(j) \cdot \max_{i \in S} \text{sim}(i, j) \;\leq\; w(j) \cdot \max_{i \in T} \text{sim}(i, j).$$

Summing over $j \in D$ gives

$$\mathcal{F}_{\text{wrep}}(S) = \sum_{j \in D} w(j) \cdot \max_{i \in S} \text{sim}(i, j) \;\leq\; \sum_{j \in D} w(j) \cdot \max_{i \in T} \text{sim}(i, j) = \mathcal{F}_{\text{wrep}}(T),$$

hence $\mathcal{F}_{\text{wrep}}$ is monotone. □