PROPEL: Prompt Optimization with Expert Priors for Small and Medium-sized LLMs

Kawin Mayilvaghanan, Varun Nathan and **Ayush Kumar**
{kawin.m, varun.nathan, ayush}@observe.ai
Observe.AI
Bangalore, India

Abstract

Optimizing discrete prompts for small and medium-sized language models (1–8 billion parameters) poses unique challenges due to their relatively weaker instruction-following capabilities compared to larger counterparts. We present **PRompt OPtimization with Expert priors for LLMs (PROPEL)**, a novel knowledge-augmented iterative optimization framework leveraging the collaborative capabilities of three distinct roles assigned to large language models (LLMs): Responder, Judge, and Optimizer. Unlike prior works that focus on larger LLMs for prompt optimization with natural language feedback, our approach emphasizes improving prompt effectiveness for resource-efficient smaller and medium-sized models through structured iteration and the integration of prompt design principles as expert priors. Our experiments across diverse tasks such as long-text summarization and entity extraction demonstrate significant improvements. **PROPEL** improves response quality by **10–24%** for **Query-Based Summarization** and **5–16%** for **Query-Based Entity Extraction** over initial prompts, outperforming baseline approaches by at least **12%** and **9%**, respectively. Ablation studies further show that incorporating **prompt design principles as priors** enhances response quality by **21%** for Llama-3.2 1B and **9%** for Llama-3.2 3B.

1 Introduction and Related Works

Large Language Models (LLMs) have excelled in various NLP tasks, including summarization, question answering, and classification (OpenAI et al., 2024b; Grattafiori et al., 2024; Abdin et al., 2024). This success stems from extensive pretraining on diverse datasets and advancements in prompting techniques (Brown et al., 2020). Prompt engineering-designing task-specific instructions or queries to elicit optimal responses—has proven critical for achieving high performance across tasks and domains (Reynolds and McDonell, 2021).

However, creating effective prompts requires substantial manual effort (Jiang et al., 2022), often relying on domain expertise (Reynolds and McDonell, 2021; Zamfirescu-Pereira et al., 2023), intuition, and trial-and-error. This resource-intensive process poses challenges for non-expert users. Moreover, LLM performance is highly sensitive to prompt quality (Gao et al., 2021), emphasizing the importance of robust prompt engineering.

Techniques such as Chain-of-Thought (Wei et al., 2023), Few-Shot Prompting (Brown et al., 2020), and Role Prompting (Wang et al., 2024) introduce prompting strategies to improve response quality. Survey Works (Schulhoff et al., 2024; Sahoo et al., 2024; Chen et al., 2024) have systematically compiled and organized these approaches into Prompt Design Principles or Prompt Principles. Bsharat et al. 2024 identified 26 principles, showing that their effectiveness varies across LLMs, indicating model-specific applicability.

To mitigate the challenges of manual prompt engineering, various automated methods have been proposed, categorized by their underlying techniques:

Soft Prompt-Tuning Methods: Soft prompt-tuning represents prompts as continuous task-specific vectors optimized using gradient-based techniques (Lester et al., 2021; Li and Liang, 2021; Liu et al., 2023; Qin and Eisner, 2021). This method fine-tunes a small subset of trainable parameters while keeping the LLM frozen, achieving strong task performance. However, it requires access to the LLM's internal state variables, making it unsuitable for black-box APIs or closed-source models (Shin et al., 2020; Lester et al., 2021).

Iterative Scoring-Based Optimization: These approaches iteratively refine a discrete base prompt. GrIPS (Prasad et al., 2023) uses phrasal edits, while APE (Zhou et al., 2023) and OPRO (Yang et al., 2024) generate semantically similar prompts via LLMs, scoring responses with an evaluation function—often another LLM—to select the next best prompt. OPRO (Yang et al., 2024) also incorporates the history of previous best prompts but doesn't explicitly generate natural language feedback. Numerical scores offer limited insights, failing to pinpoint specific response strengths or weaknesses. Our ablation study on PROPEL (Table 5) highlights the value of descriptive feedback. Moreover, these methods narrowly explore the initial prompt space, neglecting diverse strategies that could yield more robust solutions.

Iterative Feedback-Driven Optimization: These techniques use iterative feedback loops for prompt refinement. Methods like SelfRefine (Madaan et al., 2023), APO (Pryzant et al., 2023), and Self-Debugging (Chen et al., 2023) leverage LLM feedback to enhance prompts, with APO (Pryzant et al., 2023) introducing textual gradients to target weaknesses. However, APO (Pryzant et al., 2023) does not utilize the history of prompts, scores, and feedback, which we demonstrate as crucial in PROPEL (Table 4). SelfRefine (Madaan et al., 2023) and Self-Debugging (Chen et al., 2023) rely on the Responder LLM for both response and feedback. However, these methods are demonstrated on larger models, which we hypothesize are better equipped to process detailed feedback. In contrast, small- and medium-scale LLMs struggle with capability to fully utilize such feedback for self-improvement (Saunders et al., 2022; Huang et al., 2023), rendering this approach for prompt optimization ineffective when applied to these smaller models.

LLM-Based Evolutionary Techniques: Evolutionary algorithms, like the Genetic Algorithm (Holland, 1992) and Differential Algorithm (Storn and Price, 1997), have been adapted for discrete prompt optimization. Methods such as PromptBreeder (Fernando et al., 2023), PromptWizard (Agarwal et al., 2024), and EvoPrompt (Guo et al., 2024) refine prompts through iterative techniques, applying thinking styles, and leveraging LLM-driven operations like crossover and mutation. While promising, these approaches are predominantly evaluated on larger LLMs. However, as noted by (Bsharat et al., 2024), effectiveness of mutation can vary for small and medium-scale models, raising concerns about generalizability to smaller LLMs.

In this work, we tackle the critical yet underexplored challenge of optimizing discrete prompts for small and medium-sized LLMs (1–8 billion parameters), which exhibit weaker instruction-following capabilities compared to larger models (Qin et al., 2024). To address this, we propose PROPEL (PRompt OPtimization with Expert Priors for LLMs), a novel knowledge-augmented framework that leverages expert-informed prompt principles as priors and orchestrates a structured iteration among three distinct LLM roles: **Responder LLM**, responsible for response generation; **Judge LLM**, tasked with response evaluation; and **Optimizer LLM**, which refines prompts using expert priors and historical state information comprising prompts, feedback, and scores. This approach strategically incorporates structured knowledge into the prompt optimization process, enabling more effective knowledge utilization and enhancing the performance of small and medium-sized LLMs.

Unlike existing approaches that overlook the integration of prompt design principles into the optimization process, PROPEL systematically identifies task- and model-specific easy or hard to follow prompt principles. These principles are incorporated as priors over the optimal prompt space, enabling the discovery of prompts that are both effective and tailored to the specific LLM and task. Extensive experiments on summarization, entity extraction, and multi-turn QnA show improvements of at least 12%, 9% and 3% over state-of-the-art baselines, respectively, highlighting the effectiveness of the PROPEL framework.

2 Methodology

2.1 Problem Formulation

Given an initial prompt p_0 for a Responder LLM $\mathcal{M}$, our goal is to iteratively transform p_0 into $\hat{p}$, an approximation of the optimized prompt p^* which maximizes the quality of the generated responses. We assume access to an i.i.d training dataset consisting of pairs of input context and reference output: $D_{train} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_1, \ldots, x_n$ represents the input context and $y_1, \ldots, y_n$ represents the reference outputs. Note that all prompts p are drawn from the space of coherent natural language L. The quality of the generated responses $\hat{y} = \mathcal{M}(y|p, x)$ is evaluated using a metric function $m(y, \hat{y})$.

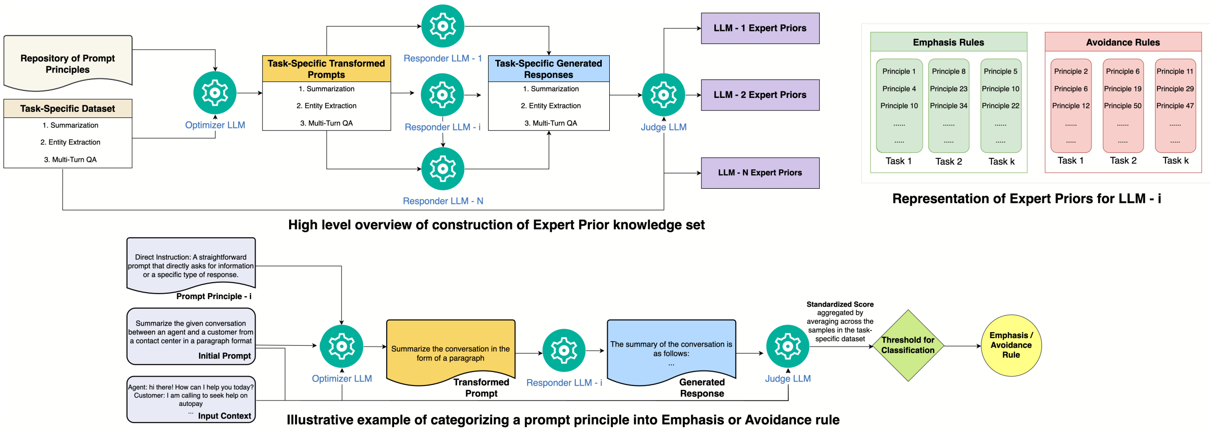

Figure 1: Visual representation of construction of Expert Prior knowledge store, which guides the prompt optimization workflow described in the next subsection.

The optimization task can be expressed as:

$$\hat{p} = \arg\max_{p \in L} \mathbb{E}[m(y, \mathcal{M}(y|p,x))] \quad \forall x, y \in D_{train}$$

The optimization process proceeds iteratively:

$$p_{k+1} = A(p_k, \text{State}, \text{Priors}),$$

where A is the algorithm explained in A.1, *State* stores the history of prompts, scores, and feedback from previous iterations, and *Priors* encode the principles to design prompts which is provided to the Optimizer LLM to guide prompt optimization.

2.2 Expert Priors

Inspired by the work of Bsharat et al. (2024), which examines the influence of LLM size on prompt design, we hypothesize that explicitly encoding the structured knowledge of the strengths and weaknesses of the Responder LLM's ability to follow specific prompt styles into the prompt optimization process will significantly enhance the quality of the prompts, as measured by the response quality. To achieve this, we define *Expert Priors*, a knowledge-augmented approach that systematically encodes the capabilities and challenges of the Responder LLM in adhering to various prompt principles. These priors guide optimization by defining *emphasis rules*, which correspond to design principles classified as *Easy* for the Responder LLM to follow, and *avoidance rules*, which correspond to design principles classified as *Hard* for the Responder LLM to follow.

As depicted in Figure 1, the *Expert Priors* are constructed as knowledge corpora[1] that capture the Responder LLM's proficiency in following different prompt styles. This is achieved by systematically transforming task-specific initial prompts with GPT-4 (OpenAI et al., 2024b) for each of the 50 prompt principles (see Tables 6 & 7) and then evaluating the Responder LLM's performance on the transformed prompts. These principles are informed by previous work, including Bsharat et al. 2024 and Chen et al. 2024, and are evaluated on a minimal dataset of 10 task-specific samples to identify *Easy* or *Hard* principles. Easy principles represent design strategies of the prompts that the Responder LLM can reliably respond with minimal errors, and these are encoded into the optimization process as *emphasis rules*, directing the optimizer to create the prompt by reinforcing these principles. Conversely, Hard principles represent design strategies of prompts where the Responder LLM is more likely to struggle, and these are encoded as *avoidance rules*, signaling the optimizer to create prompt that does not adhere to the characteristics defined by these principles to minimize errors and maintain high-quality responses.

Our approach leverages Expert Priors as structured knowledge in a plug-and-play manner, enhancing prompt optimization without requiring fine-tuning of the Responder LLM. By strategically utilizing knowledge, the Optimizer LLM generates high-quality prompts tailored to the Responder LLM's strengths and weaknesses. The 50 principles are automatically classified as Easy or Hard once per Responder LLM per task, enabling scalable and efficient knowledge elicitation. The detailed implementation of this automated classification process is described in Section 7i.

[1] Expert priors are available in Section A.11.

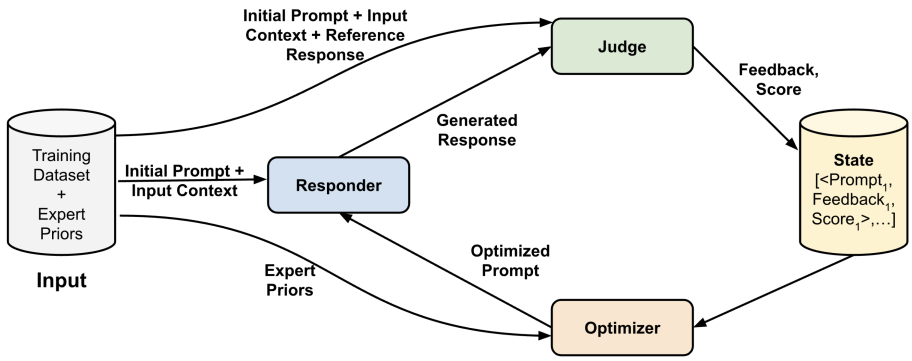

Figure 2: Proposed workflow: The Responder generates responses, the Judge evaluates them with feedback and score, and the Optimizer optimizes prompt using expert priors and state. This process is repeated iteratively.

2.3 Framework Design and Workflow

As shown in Figure 2, PROPEL operates as an iterative framework involving the coordinated interaction of three key components: the Responder LLM, Judge LLM, and Optimizer LLM. Each component is driven by role-specific meta-prompts, ensuring alignment with their designated functions:

1. **Response Generation by Responder LLM**: The Responder LLM generates a response ($\hat{y}$) based on the *input*, which consists of the current prompt p_k and the training dataset D_{train}. At the beginning of the process, the current prompt p_k is initialized to the initial prompt p_0.

2. **Evaluation by Judge LLM**: The Judge LLM evaluates the generated response ($\hat{y}$) using the current prompt p_k, the input context x, and the reference response y. This evaluation follows predefined criteria outlined in Section A.6. The Judge provides detailed feedback and assigns a score between 1 and 5 for each training data point (x, y). The feedback and scores are then summarized using the *Feedback Summary* meta-prompt (refer to Section A.8). The summarized insights, representing knowledge extracted from the evaluation process, are stored in the **State** as a triplet $\langle p_k, \text{Feedback}_k, \text{Score}_k \rangle$. This approach effectively curates and utilizes evaluation knowledge, aligning with knowledge-augmented methodologies.

3. **Prompt Optimization by Optimizer LLM**: The Optimizer LLM generates a refined prompt p_{k+1} based on the analysis of the **State** and adherence to *Expert Priors*. The metaprompt (Refer "Optimizer LLM" in section A.8) ensures that the optimizations preserve the intent of the original task while enhancing the response quality.

4. **Repeat**: Steps 1–3 are repeated iteratively for a specified number of iterations or until a stopping criteria defined by a threshold on the score of quality of responses.

For further implementation details, refer to *Key Implementation Details for PROPEL* in Section A.7.

3 Experimentation Details

3.1 Dataset and Tasks

This work utilizes the public **QMSum** (Zhong et al., 2021) dataset for the task of query-based summarization and the public benchmark **MT Bench** (Zheng et al., 2023) for evaluating QnA performance in multi-turn chat scenarios. Additionally, an **in-house** dataset[2] is employed for tasks including query-based summarization and entity extraction. Comprehensive details about these datasets, associated tasks, and rationale for their selection are provided in Section A.2.

3.2 Models used as Responder LLM

We select a diverse range of open-source models, including small models (1–3 billion parameters) and medium-sized models (7-8 billion parameters). Further details on the specific characteristics and rationale for choosing these models can be found in Section A.4.

3.3 Baselines

The study uses Self-Refine (Madaan et al., 2023), GrIPS (Prasad et al., 2023), Prompt Wizard (Agarwal et al., 2024), APO (Pryzant et al., 2023), and

[2]Due to proprietary restrictions, this dataset cannot be released.

OPRO (Yang et al., 2024) as baseline methods. In the *PROPEL (Self)* variant, the same LLM employed as the Responder LLM also serves as the Judge LLM and Optimizer LLM. Further details and the rationale for selecting these baselines are provided in Section A.5.

3.4 Metrics

Response quality for evaluating summarization, QnA in multi-turn chat and entity extraction tasks is obtained using *LLM Score*, a rating on a scale of 1–5 provided by GPT-4o (OpenAI et al., 2024a); Rouge-1 (Lin, 2004) is applied to entity extraction tasks. Table 1 presents the standardized metric differences between responses generated using optimized prompts and initial prompts. Details of the metrics, their computation, and standardization procedures are provided in Section A.6.

3.5 Implementation Details

Evaluation was conducted on the final optimized prompt, defined as either the last iteration's prompt or an intermediate one when early stopping occurred. Further details on the implementation can be found in Section A.7.

4 Experimental Results

4.1 Overall Results

We evaluate PROPEL against several baselines across two categories of Responder LLMs: Small Models (1–3 Billion parameters) and Medium-Sized Models (7–8 Billion parameters). The results are summarized in Tables 1 and key observations are detailed below:

Small Models (1–3 Billion Parameters)

1. **Superior Performance of PROPEL:** PROPEL outperforms all baselines across datasets and tasks, highlighting the crucial role of knowledge-informed Expert Priors in boosting performance for smaller models. Specifically, PROPEL achieves substantial margins over baselines, with improvements of 20%, 6% and 4% for Llama 3.2 1B, Llama 3.2 3B and Flan-T5 3B in summarization, and 14%, 10% and 13.7% for entity extraction. The more pronounced improvements in the smaller 1B model underscore the value of knowledge-augmented priors in guiding prompt optimization and maximizing the potential of resource-efficient LLMs.
2. **PROPEL (Self) vs. PROPEL**: PROPEL (Self) performs substantially worse than PROPEL across datasets and tasks, highlighting that smaller LLMs are less effective at evaluating responses and generating optimized prompts. This emphasizes the necessity of leveraging more powerful external LLMs (e.g., GPT-4) to provide feedback and guide the prompt optimization process effectively.

Medium-Sized Models (7–8 Billion Parameters)

1. **Superior Performance of PROPEL:** Similar to the small models, **PROPEL** consistently outperforms the baselines across datasets and tasks. This highlights the effectiveness of integrating knowledge-informed Expert Priors in optimizing prompt design, thereby enhancing the capabilities of medium-sized models.
2. **PROPEL (Self) vs. PROPEL:** Unlike smaller models, **PROPEL (Self)** demonstrates significantly improved performance relative to the baselines when using medium-sized models, although it still lags behind **PROPEL**. This indicates that medium-sized models are better equipped to serve as both evaluators and optimizers. However, achieving enhanced response quality still requires the coordination of a more powerful LLM for Judge and Optimizer LLM roles.

The key takeaways are as follows:

1. **Significance of Knowledge-Informed Expert Priors: Expert Priors** are particularly critical for smaller and medium-sized models (1–8 Billion parameters), significantly boosting performance across all of the tasks and datasets by leveraging knowledge-informed guidance in prompt optimization.
2. **Role of powerful LLM in Evaluation and Optimization:** Using a powerful external LLM like GPT-4 for evaluation and optimization is more effective than relying on the smaller Responder LLM. A more capable LLM offers detailed, accurate assessments by better understanding the prompt-response-reference response alignment. During optimization, it identifies prompt issues and generates improved versions that better align with task objectives and the characteristics of smaller Responder LLMs.

Responder LLM	Methods	Summarization		Entity Extraction		Chat
		Inhouse	QMSum	Inhouse		MT-Bench
		ΔLLM Score	ΔLLM Score	ΔLLM Score	ΔROUGE-1	ΔLLM Score
Llama 3.2 1B	Self-Refine	-29.00	-12.74	-46.82	-33.64	-10.04
	GrIPS	3.28	4.53	2.94	2.19	1.25
	Prompt-Wizard	-5.20	-2.64	-2.95	-3.70	-1.52
	APO	-3.20	-3.96	-9.20	-11.50	2.86
	OPRO	-2.00	-1.28	-1.75	-3.27	1.67
	PROPEL (Self)	-14.00	-10.27	-48.00	-25.04	-9.82
	PROPEL	**22.67**	**7.91**	**11.00**	**16.43**	**8.46**
Llama 3.2 3B	Self-Refine	-13.54	-8.37	-30.16	-18.48	-5.39
	GrIPS	2.79	2.92	3.16	2.65	1.69
	Prompt-Wizard	-7.44	-6.34	-1.65	-1.80	1.18
	APO	2.67	3.42	-6.40	-8.89	3.27
	OPRO	8.00	4.58	1.50	1.68	2.50
	PROPEL (Self)	-12.34	-5.90	-29.00	-13.57	-2.26
	PROPEL	**14.17**	**6.29**	**8.00**	**12.58**	**6.79**
Flan-T5 3B	Self-Refine	-3.80	-9.75	-6.30	-1.57	-1.13
	GrIPS	5.60	4.51	1.90	4.00	2.60
	Prompt-Wizard	-1.27	-4.98	-2.60	-2.38	-1.20
	APO	1.58	3.91	-7.56	-6.73	-1.48
	OPRO	6.56	-7.42	-6.02	-6.25	-7.83
	PROPEL (Self)	-8.38	-4.38	-2.02	-1.86	-2.16
	PROPEL	**10.29**	**7.03**	**15.61**	**13.7**	**6.63**
Mistral 7B	Self-Refine	3.71	2.85	1.48	0.95	-0.74
	GrIPS	5.96	3.13	-1.64	0.73	-0.58
	Prompt-Wizard	3.02	3.58	-0.84	-1.18	-1.92
	APO	-13.00	-2.68	-9.63	-8.79	-0.84
	OPRO	10.05	3.76	-0.80	-3.14	0.37
	PROPEL (Self)	17.39	3.62	2.40	1.83	1.09
	PROPEL	**21.50**	**8.73**	**3.90**	**8.96**	**3.73**
Llama 3 8B	Self-Refine	-2.00	-1.08	-3.00	-1.54	2.17
	GrIPS	-2.32	1.40	-3.80	-0.20	1.84
	Prompt-Wizard	-1.96	2.73	-1.20	-1.83	1.37
	APO	-7.46	-3.40	-3.20	-2.84	1.25
	OPRO	5.00	-1.26	-3.59	-3.42	1.89
	PROPEL (Self)	21.35	5.70	6.40	2.67	2.04
	PROPEL	**23.68**	**12.20**	**8.20**	**4.64**	**2.45**

Table 1: The table compares PROPEL with baselines across tasks and models, using *GPT-4* as Judge and Optimizer. It shows the difference in standardized LLM scores ('Δ') between responses generated with optimized prompts and initial prompts for Summarization and Chat tasks. Raw LLM scores are standardized from the GPT-4o ratings (1–5 scale) to percentages. ΔROUGE-1 is used for Entity Extraction task. Higher scores indicate better performance, with the best results for each model in bold and color. Negative scores indicate worse performance with optimized prompts.

3. **Consistent Task Performance: PROPEL** consistently delivers strong performance across tasks, demonstrating exceptional effectiveness in knowledge-intensive scenarios, including query-based summarization, entity extraction, and QnA over multi-turn chat, underscoring its versatility and robust design.

4.2 Ablation Studies

We focus on the Query-Based Summarization task in the in-house dataset containing five prompts and the Llama-3.2 1B and 3B models as the basis for all the ablation studies.

Impact of Initial Prompt: Certain prompts pose varying levels of difficulty for different models, making harder prompts more challenging to optimize. These harder prompts often require additional training data or more iterations in the optimization algorithm to converge effectively. The objective of this study is to check if PROPEL can work as effectively with "hard" prompts as they do with "easy" prompts.

For each of the considered prompts, the Optimizer LLM generates an “easy” and a “hard” version by following specific rules derived from a one-time evaluation of the Responder LLM on 50 prompt principles (Refer table 6). The “easy” prompts emphasize aspects the Responder LLM excels at, while the “hard” prompts intentionally include elements the Responder LLM struggles with. This process is conducted for two models: Llama-3.2-1B and Llama-3.2-3B.

We measure the effect of prompt difficulty on the effectiveness of PROPEL by calculating the average LLM Scores for responses generated with initial prompts (LLM $\text{Score}_{\text{init}}$) and the average improvement in scores after optimization (ΔLLM Score). The sum of these metrics gives the final LLM Scores for the optimized prompts.

The results in Table 2 show that while the LLM Scores for the final optimized prompts are higher for "easy" prompts compared to "hard" prompts, the improvements in ΔLLM Scores are significantly greater for "hard" prompts. In other words, the gap in LLM Scores between the initial "easy" and "hard" prompts is much larger than the gap between the optimized "easy" and "hard" prompts. This indicates that PROPEL is highly effective even for "hard" prompts, despite their greater optimization challenge.

Initial Prompt	Llama-3.2 1B		Llama-3.2 3B	
	LLM $\text{Score}_{\text{init}}$	ΔLLM Score	LLM $\text{Score}_{\text{init}}$	ΔLLM Score
Easy	67.60	13.47	70.04	7.83
Hard	40.06	33.47	43.86	25.47

Table 2: Impact of initial prompt on ΔLLM Score for Llama-3.2 1B and 3B models.

Impact of Expert Priors on Knowledge-Guided Prompt Optimization: The goal of this study is to evaluate how the quality of expert priors influences the effectiveness of the knowledge-augmented prompt optimization algorithm. We test the following scenarios:

1. **Good Priors:** Rules derived directly from the one-time evaluation of the Responder LLM using 50 prompt principles (Refer table 6). These “emphasis” and “avoidance” rules, which serve as structured knowledge priors, are incorporated into the Optimizer LLM’s prompts during optimization.

2. **No Priors:** The Optimizer LLM operates without any knowledge-augmented rules, relying solely on iterative feedback for prompt optimization.

3. **Random Priors:** Seven prompt principles are randomly selected from the curated list for each rule type (“emphasis” and “avoidance”), and the resulting rules are included in the Optimizer LLM’s prompts. This tests the impact of unstructured knowledge injection on optimization performance.

4. **Bad Priors:** The “emphasis” and “avoidance” rules are reversed, such that principles the Responder LLM excels at are treated as “avoidance” rules and vice versa. This setup evaluates the negative influence of inaccurately encoded knowledge on prompt optimization.

The effectiveness of each scenario is measured by the average change in LLM scores after optimization (ΔLLM Score). Results in Table 3 show that “Good Priors” lead to the highest improvement of 10-15%, followed by “Random Priors,” “No Priors”, and “Bad Priors.” The decline in performance with “Bad Priors” stems from the propagation of incorrect information into the optimization process.

Priors Applied	ΔLLM Score	
	Llama 3.2 1B	Llama 3.2 3B
Bad Priors	-6.74	-7.84
Random Priors	7.62	4.86
No Priors	1.67	5.14
Good Priors	22.67	14.17

Table 3: Impact of priors on ΔLLM Score for Llama-3.2 1B and 3B models.

Effect of State History as Knowledge Context: This study investigates the role of state history (prompts, feedback, and scores) as Knowledge Context in guiding the prompt optimization process. Two settings are compared: (1) Without History — Only the latest state (prompt, feedback, and score) is provided to the Optimizer LLM for refining prompts, without any historical context.; and (2) With History — The Optimizer LLM is supplied with a sequence of the last 10 states (Refer Section A.7), effectively leveraging historical knowledge comprising prompts, feedback, and scores. Results in Table 4 show that using state history improves ΔLLM Score by 8.9% for Llama-3.2 1B and 4% for Llama-3.2 3B. This highlights the importance of historical knowledge patterns in improving prompt optimization, demonstrating the value of knowledge persistence and contextual learning in iterative LLM interactions.

History Applied	ΔLLM Score	
	Llama-3.2 1B	Llama-3.2 3B
Without History	13.80	10.13
With History	22.67	14.17

Table 4: Impact of State History on ΔLLM Score for Llama-3.2 1B and 3B models.

Impact of Using Feedback and Scores as Knowledge Signals in State: This study examines the impact of incorporating feedback and scores as knowledge signals in the State on prompt optimization effectiveness. Three settings are evaluated: (1) Feedback Only: State includes a history of size 10 with prompts and their corresponding feedback from the Judge LLM. (2) Scores Only: State includes a history of size 10 with prompts and their corresponding scores. (3) Feedback & Scores: State includes prompts, feedback, and scores (default setting).

The effectiveness is measured by the average change in LLM scores after optimization (ΔLLM Score). Results in Table 5 show that using only scores slightly outperforms using only feedback, while combining feedback and scores significantly improves performance, with gains of 14.5% and 8% for Llama-3.2 1B and 3B models, respectively. This suggests that feedback and scores provide complementary insights—feedback highlights areas for improvement, while scores offer a performance baseline. Together, they give the Optimizer LLM a richer context, enabling more effective prompt refinements.

Evaluator Output	ΔLLM Score	
	Llama-3.2 1B	Llama-3.2 3B
With Feedback Only	7.30	5.83
With Scores Only	8.20	6.21
With Feedback & Scores	22.67	14.17

Table 5: Impact of Evaluator and State on ΔLLM Score for Llama-3.2 1B and 3B models.

5 Conclusion

We introduce **PROPEL**, a novel prompt optimization framework that leverages expert priors and iterative feedback from a Judge LLM to improve LLM performance across various tasks. By integrating knowledge-informed priors, **PROPEL** effectively guides the optimization process, particularly benefiting small and medium-sized models. Experimental results demonstrate that **PROPEL** outperforms existing baselines, showing substantial gains in query-based summarization, entity extraction, and QnA over multi-turn interactions. Ablation studies reveal that the inclusion of expert priors enhances LLM scores by 10–15%, highlighting their pivotal role in aligning prompt design with model-specific strengths and limitations.

6 Limitations

We recognize certain limitations in the scope of the presented work. First, while the reliance on expert priors requires a one-time evaluation, this process may not be readily applicable to all scenarios, particularly those with highly dynamic requirements. Second, **PROPEL** relies on reference responses for evaluation, which, while effective, may pose challenges for fully automated scalability in tasks without readily available references. Third, the current evaluation focuses on tasks such as query-based summarization, entity extraction, and multi-turn QnA. While these tasks demonstrate the versatility of **PROPEL**, additional testing on more complex and diverse modeling challenges would further validate its generalizability. Additionally, the present study is confined to small and medium-sized models up to 8B parameters. Extending the analysis to larger models would better demonstrate **PROPEL**'s adaptability and effectiveness across a broader range of model architectures. Finally, the testing was conducted exclusively on English-language tasks, leaving multilingual capabilities an area for future exploration to ensure broader applicability across diverse linguistic settings.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Singh Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio C'esar Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allison Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Young Jin Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xianmin Song, Olatunji Ruwase, Praneetha Vaddamanu, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Cheng-Yuan Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *ArXiv*, abs/2404.14219.

Eshaan Agarwal, Vivek Dani, Tanuja Ganu, and Akshay Nambi. 2024. Promptwizard: Task-aware agent-driven prompt optimization framework. *Preprint*, arXiv:2405.18369.

Ahmed Ali and Steve Renals. 2018. Word error rate estimation for speech recognition: e-WER. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 20–24, Melbourne, Australia. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Sondos Mahmoud Bsharat, Aidar Myrzakhan, and Zhiqiang Shen. 2024. Principled instructions are all you need for questioning llama-1/2, gpt-3.5/4. *Preprint*, arXiv:2312.16171.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics.

Banghao Chen, Zhaofeng Zhang, Nicolas Langrené, and Shengxin Zhu. 2024. Unleashing the potential of prompt engineering in large language models: a comprehensive review. *Preprint*, arXiv:2310.14735.

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

Hyunlee Chung, Mitesh Tapaswi, Denny Chen, Yiming Zhao, Mehran Sadeghi, , et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2201.11903*.

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

J.L. Fleiss et al. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde,

Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve

Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2024. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *Preprint*, arXiv:2309.08532.

John H. Holland. 1992. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

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. *Preprint*, arXiv:2310.06825.

Ellen Jiang, Kristen Olson, Edwin Toh, Alejandra Molina, Aaron Michael Donsbach, Michael Terry, and Carrie Jun Cai. 2022. Prompt-based prototyping with large language models.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *Preprint*, arXiv:2104.08691.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *Preprint*, arXiv:2101.00190.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. Gpt understands, too. *Preprint*, arXiv:2103.10385.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. *Preprint*, arXiv:2303.17651.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian O'Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub

Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kavin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu, Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljubeh, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu, Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers, Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum, Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunninghman, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyi Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yury Malkov. 2024a. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David

Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024b. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. Grips: Gradient-free, edit-based instruction search for prompting large language models. *Preprint*, arXiv:2203.07281.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. *Preprint*, arXiv:2305.03495.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. Infobench: Evaluating instruction following ability in large language models. *Preprint*, arXiv:2401.03601.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *Preprint*, arXiv:1908.10084.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. *Preprint*, arXiv:2102.07350.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A systematic survey of prompt engineering in large language models: Techniques and applications. *Preprint*, arXiv:2402.07927.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-critiquing models for assisting human evaluators. *Preprint*, arXiv:2206.05802.

Sander Schulhoff, Michael Ilie, Nishant Balepur, Konstantine Kahadze, Amanda Liu, Chenglei Si, Yinheng Li, Aayush Gupta, HyoJung Han, Sevien Schulhoff, Pranav Sandeep Dulepet, Saurav Vidyadhara, Dayeon Ki, Sweta Agrawal, Chau Pham, Gerson Kroiz, Feileen Li, Hudson Tao, Ashay Srivastava, Hevander Da Costa, Saloni Gupta, Megan L. Rogers, Inna Goncearenco, Giuseppe Sarli, Igor Galynker, Denis Peskoff, Marine Carpuat, Jules White, Shyamal Anadkat, Alexander Hoyle, and Philip Resnik. 2024. The prompt report: A systematic survey of prompting techniques. *Preprint*, arXiv:2406.06608.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV au2, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *Preprint*, arXiv:2010.15980.

Rainer Storn and Kenneth Price. 1997. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359.

Zekun Moore Wang, Zhongyuan Peng, Haoran Que, Jiaheng Liu, Wangchunshu Zhou, Yuhan Wu, Hongcheng Guo, Ruitong Gan, Zehao Ni, Jian Yang, Man Zhang, Zhaoxiang Zhang, Wanli Ouyang, Ke Xu, Stephen W. Huang, Jie Fu, and Junran Peng. 2024. Rolellm: Benchmarking, eliciting, and enhancing role-playing abilities of large language models. *Preprint*, arXiv:2310.00746.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

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.

J.D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. 2023. Why johnny can't prompt: How non-ai experts try (and fail) to design llm prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA. Association for Computing Machinery.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *Preprint*, arXiv:2104.05938.

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. In *The Eleventh International Conference on Learning Representations*.

A Appendix

A.1 Algorithm

The optimization algorithm is provided below:

Algorithm 1 Prompt Refinement with Iterative Descriptive Evaluation (PROPEL)

```
Input: Initial prompt $p_0$, dataset $D_{train} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, and expert priors.
Output: An optimized prompt $\hat{p}$
Require: $K$: number of iterations, $s_t$: score threshold for early stopping, $h_s$: size of history maintained in State
Initialize State as [].
for $k = 1$ to $K$ do
    Generate responses $\{\hat{y}_i = M(y_i|p_k, x_i)\}_{x_i \in D_{train}}$ using Responder LLM.
    Evaluate responses $\hat{y}_i$ with Judge LLM to obtain sample-specific feedbacks and scores, aggregated into a single score Score$_k$ and summarized feedback Feedback$_k$.
    if Score$_k \geq s_t$ then
        Break the loop.
    end if
    Update State with $\langle p_k, \text{Feedback}_k, \text{Score}_k \rangle$.

    if len(State) $> h_s$ then
        Pop the leftmost element from State.
    end if
    Refine $p_k$ to obtain $p_{k+1}$ using Optimizer LLM, incorporating State and expert priors.
end for
Return: $\hat{p} = p_k$
```

A.2 Dataset used for Experiments

1. **QMSum - Query based Summarization**: The QMSum dataset (Zhong et al., 2021) provides query-based multi-domain meeting summaries, consisting of transcripts from academic, industrial, and product review meetings. We chose this dataset owing to its similarities with our in-house dataset and for its scale.

2. **MT Bench**: MT-Bench (Zheng et al., 2023) is a curated benchmark featuring high-quality, multi-turn questions designed to evaluate conversation flow and instruction-following capabilities in multi-turn dialogues. GPT-4 (OpenAI et al., 2024b) evaluates MT Bench outputs, and average scores across questions and

turns are reported for all experiments. We chose this dataset due to it being widely recognized and its widespread use in evaluating models for tasks that require sustained, coherent interactions over multiple exchanges.

3. We leverage an in-house dataset[3] of conversational interactions happening in a contact center between agents and customers and perform the following tasks:

   (a) **Query based Summarization (QBS)**: Involves generating an abstractive response to the given query based on the conversation as input context.

   (b) **Query based Entity Extraction (QBE)**: Involves identifying and extracting entities from the conversation that are pertinent to the given query.

   The dataset was carefully curated to ensure high-quality annotations and relevance to contact center tasks. Below, we outline the process followed to create this dataset:

   (a) **Conversation Sampling and Transcription:** A total of 120 English dyadic conversations between agents and customers were sampled from a contact center. These conversations were transcribed using a third-party Automatic Speech Recognition (ASR) engine, with a Word Error Rate (WER) of approximately 10% as reported by Ali and Renals 2018. The ASR transcripts served as the input context in the prompt optimization algorithm.

   (b) **Data Partitioning:** The data is split into non-overlapping datasets.

      i. **Training Set:** A randomly selected subset of 10 conversations.

      ii. **Evaluation Set:** A randomly selected subset of 100 conversations.

      iii. **Prior Identification Set:** A randomly selected subset of 10 conversations.

   (c) **Task and Prompt Sampling:** We sampled 15 prompts from a proprietary contact-center dataset [4], evenly distributed across the following two tasks: Query-Based Summarization and Query-Based Entity Extraction. The contact-center dataset from which prompts were sampled was designed to evaluate the value of **Query Based Summarization** use case for one of our clients.

   (d) **Annotation Process:**

      i. **Annotator Selection:** Seven annotators with experience in analyzing contact center data were employed.

      ii. **Guidelines for Annotation:** Annotators followed a comprehensive guideline emphasizing logical reasoning to identify relevant evidence from conversations, synthesize information and provide task-specific responses to each of the 15 prompts. The annotations were designed to ensure grounding in conversation details and emulate the reasoning process of a domain expert.

   (e) **Quality Assurance:**

      i. **Entity Extraction Task:** Responses for this structured task were validated through a majority voting mechanism requiring agreement among at least four annotators. Inter-Annotator agreement was calculated using Fleiss' Kappa (Fleiss et al., 1971), and was measured at 0.64, indicating substantial agreement.

      ii. **Query-Based Summarization Task:** Responses, being unstructured, were evaluated using Sentence-BERT (Reimers and Gurevych, 2019), fine-tuned for the Semantic Textual Similarity (STS) (Cer et al., 2017) task. Samples with an average pairwise semantic similarity score $<$ 0.8 were flagged and re-annotated. For samples meeting the $\geq$ 0.8 similarity threshold, one response was randomly selected to ensure consistency in evaluation.

This robust annotation and curation process ensures that the dataset reflects the nuances of contact center data and provides reliable input for evaluating prompt optimization in downstream tasks.

[3]We cannot release the dataset due to proprietary reasons.

[4]We cannot release the dataset due to proprietary reasons.

A.3 Dataset Statistics

Table 8 shows the number of runs and the # of data points per run in the evaluation and training datasets. Here, '# of Runs' denotes the number of unique instructions (prompts) for which separate prompt tuning is required. The number of evaluation data points per run is calculated as the total number of data points divided by the number of runs.

Dataset	# of Runs	# Samples per Run Evaluation	# Samples per Run Training	# Samples Prior Identification
Inhouse - QBS	5	100	10	10
Inhouse - QBE	5	100	10	10
QMSum	2	100	10	10
MT Bench	1	60	10	10

Table 8: Dataset Statistics

A.4 Models used as Responder LLM

1. **Selection Criteria:** The selection of Responder LLMs was guided by two factors:
   (a) Adoption in the industry and research community.
   (b) Model size, to emphasize the challenges faced by small and medium-sized models in effective prompt optimization and demonstrate the importance of *expert priors* for these models.
2. **Model Descriptions:**
   (a) **Small Models (1–3B):**
      i. **Llama-3.2-1B (Grattafiori et al., 2024):** A lightweight 1-billion-parameter model used to assess prompt optimization challenges for minimal-scale models.
      ii. **Llama-3.2-3B (Grattafiori et al., 2024):** A 3-billion-parameter model, widely used in research for its accessible size and performance.
      iii. **Flan-T5-3B:** A proprietary model fine-tuned from the open-source Flan-T5-3B model (Chung et al., 2022) on a dataset of contact center interactions curated in-house [5] for tasks like summarization, and entity extraction. It uses Supervised Fine-Tuning (Brown et al., 2020) and Direct Preference Optimization (Rafailov et al., 2023), making it an ideal candidate for task-specific evaluation.
   (b) **Medium-Sized Models (7–8B):**
      i. **Mistral-7B (Jiang et al., 2023):** A 7-billion-parameter model known for its widespread use in various NLP applications.
      ii. **Llama-3-8B (Grattafiori et al., 2024):** Balances size and performance, making it a representative mid-sized model for our experiments.

[5]We cannot release the dataset due to proprietary reasons.

This diversity allows us to investigate the specific challenges smaller models face and the role of expert priors in addressing these challenges.

A.5 Baselines used for Experiments

1. **Self-Refine:** Madaan et al. 2023 introduced an approach that iteratively improves initial LLM outputs through self-feedback and refinement. The process involves the LLM generating an initial output, providing feedback on its own output, and refining it iteratively.
2. **GrIPS:** Prasad et al. 2023 is an iterative approach that refines prompts by performing edit operations such as Deletion, Swap, Addition and Paraphrasing on the phrases of the prompts.
3. **Prompt Wizard:** The framework (Agarwal et al., 2024) iteratively refines prompts by mutating instructions and incorporating negative examples to deepen understanding and ensure diversity. It further enhances both instructions and examples with the aid of a critic, synthesizing new instructions and examples enriched with detailed reasoning steps for optimal performance. PromptWizard has shown superior performance over other Evolutionary Prompt Optimization Algorithms such as PromptBreeder (Fernando et al., 2023) and EvoPrompt (Guo et al., 2024). Thus, PromptWizard is chosen as the candidate baseline belonging to the line of Prompt Optimization Algorithms.
4. **APO:** Pryzant et al. 2023 is an iterative approach that leverages minibatches of data to form natural language gradients that dissect the limitations of the current prompt and edit the same in the opposite semantic direction

of the gradient. These gradient descent steps are guided by a beam search and bandit selection procedure in order to improve algorithmic efficiency. The beam search is an iterative optimization process involving an expansion step which leverages paraphrasing to explore the local monte carlo search space around the new prompt candidates and a selection step to choose the top most promising candidates for the next iteration.

5. **OPRO:** Yang et al. 2024 is an iterative approach that leverages LLMs for the optimization task described in natural language. In each optimization step, the LLM generates new solutions from the prompt that contains previously generated solutions with their values, then the new solutions are evaluated and added to the prompt for the next optimization step.

We carefully selected the baselines for comparison with our proposed approach (PROPEL) based on their algorithmic similarities and the traction they have received due to their widespread adoption and recognition within the research community.

A.6 Evaluation Metrics used for Experiments

1. **LLM Score:** Evaluates quality of the generated responses for tasks including Query Based Summarization (QBS) in the in-house and QMSum (Zhong et al., 2021) datasets, Query Based Entity Extraction (QBE) in the in-house dataset and conversation flow and instruction-following capabilities in MT Bench (Zheng et al., 2023) dataset. GPT-4o (OpenAI et al., 2024a) is employed to provide a score on a scale of 1-5 basis the following criteria:

    (a) **Alignment with Prompt:** Measures the degree of alignment of the generated response with respect to the prompt.

    (b) **Completeness:** Measures the degree to which the generated response is complete with respect to the reference response.

    (c) **Presence of irrelevant information:** Measures the degree of irrelevant information in the generated response with respect to the reference response.

    (d) **Adherence to the format:** Checks if the generated response adheres to the desired response format, if any, as defined in the prompt.

    Higher the score, better is the quality of the generated response. You may refer the prompt we used for the Evaluation at Section A.10. The difference in LLM scores reported in 1 are post standardization to percentages which is done using the formula:

    $$\text{LLM Score} = \frac{\text{LLM Score}_x}{5} \times 100 \quad (1)$$

    Please note that the standardization of LLM scores is applied solely to calculate and report the score difference between the responses generated from the initial prompt and the final optimized prompt, and is not used during the optimization process.

2. **Rouge-1 F1 Score (Lin, 2004):** Measures the overlap of unigrams (single words) between the generated response and the reference response. Employed as one of the metrics for the Query Based Entity Extraction (QBE) task in the in-house dataset, it provides a straightforward measure of comparing entity value similarity based on word matching.

A.7 Implementation Details for Experiments

1. **Experimental Setup:** We conducted experiments using three datasets:

    (a) **In-house Dataset:** As described in Section A.2, we utilized a curated dataset of human annotations for contact-center interactions. A random sample of 10 data points was used for training, while evaluation was performed on 100 data points per prompt.

    (b) **QMSum Dataset:** A similar experimental setup was followed, with 10 data points for training and 100 data points for evaluation per prompt.

    (c) **MT Bench Dataset:** This dataset lacks input context. For models supporting system prompts, we tuned the system prompt, while for models like Flan-T5-3B that lack explicit system prompt support, we optimized a prefix prompt appended to the user prompt.

    Evaluation was conducted on the final optimized prompt, defined as either the last iteration's prompt or an intermediate one when early stopping occurred. Scores reported in Table 1 are averages across multiple prompts.

2. **Hyperparameters for PROPEL:**
   (a) **Judge and Optimizer LLM:** GPT-4
   (b) **PROPEL (Self):** Responder LLM served as Judge and Optimizer
   (c) **Number of Iterations:** Up to 10
   (d) **Temperature for Judge and Optimizer LLM**: 1.00
   (e) **Temperature for Responder LLM**: 0.01
   (f) **Score Threshold for Early Stopping:** 4.5
   (g) **State History Size:** 10
   (h) **Training Dataset Size:** 10 samples

3. **Key Implementation Details for PROPEL:**
   - **Meta-Prompts**: Carefully crafted prompts guide each LLM role to ensure consistency and focus.
   - **Stopping Criteria**: The process runs for a set number of maximum iterations or till the score exceeds a set score threshold.
   - **History Size**: We define a fixed size of history to be maintained in the **State** which is used by the Optimizer LLM for prompt refinement.
   - **Task-Specific Adaptation**: Prompts are fine-tuned based on the task and model specific nuances captured via **Expert Priors**.

4. **Baseline Results and Hyperparameters:** For baseline comparisons, we've adapted the codebases provided by the respective authors to generate results. The hyperparameter settings for the baselines are adopted directly from the corresponding papers or their publicly available codebases, where applicable, and are as follows:
   (a) **OPRO:**
      i. **Optimizer LLM: GPT-4**
      ii. **Objective Function Evaluator:** GPT-4
      iii. **Number of Iterations:** 10
      iv. **Temperature for Evaluator and Optimizer LLM:** 1.00
      v. **Training Dataset Size:** 10 samples
      vi. **Number of prompts generated per iteration:** 1.00
   (b) **APO:**
      i. **Optimizer LLM:** GPT-4
      ii. **Text Loss Function Calculator and Gradient Generator:** GPT-4
      iii. **Number of Iterations:** 10
      iv. **Temperature for Gradient Generator and Optimizer LLM:** 1.00
      v. **Training Dataset Size:** 10 samples
      vi. **Number of prompts generated per iteration:** 1.00
   (c) **Prompt-Wizard:**
      i. **Evaluator, Mutator and Optimizer LLM:** GPT-4
      ii. **Number of Iterations:** 10
      iii. **Temperature for Evaluator, Mutator and Optimizer LLM:** 1.00
      iv. **Training Dataset Size:** 10 samples
      v. **Number of mutations per iteration:** 1.00
   (d) **GrIPS:**
      i. **Evaluator LLM:** GPT-4
      ii. **Number of Iterations:** 10
      iii. **Temperature for Evaluator LLM:** 1.00
      iv. **Training Dataset Size:** 10 samples
      v. **Number of Edits per iteration:** 3.00
   (e) **Self-Refine:**
      i. **Evaluator and Optimizer LLM:** Same as Responder LLM
      ii. **Number of Iterations:** 10
      iii. **Temperature for Evaluator and Optimizer LLM:** 1.00
      iv. **Training Dataset Size:** 10 samples

5. **Model Inference:**
   (a) **APIs and Deployment:**
      i. **Llama-3.2-1B/3B, Llama-3-8B, and Mistral-7B:** Amazon Bedrock APIs
      ii. **GPT-4:** OpenAI API
      iii. **Flan-T5-3B:** Internally deployed on an AWS EC2 instance with an NVIDIA A10G GPU (24 GB GPU memory).
   (b) **Inference Strategy:** For all models except Flan-T5-3B, inference was con-

ducted in one pass. For Flan-T5-3B, hierarchical processing was employed for input contexts exceeding 1200 tokens. Input contexts were segmented into 1200-token chunks, processed independently and in parallel, with segment-level responses summarized to produce the final output.

6. **Infrastructure:** Experiments were conducted on an AWS p4de.24xlarge instance equipped with eight GPUs, each with 80 GB of memory.

7. **Expert Priors:** We incorporated expert priors into the prompt optimization process to guide the Optimizer LLM. The steps involved are as follows:

   (a) **Mechanism for deriving prompt principles as priors:** Prompt principles compile best practices for crafting high-quality prompts for LLMs, adapted from works such as (Bsharat et al., 2024) and (Chen et al., 2024), as well as from model documentation specifying LLM prompting best practices. You may refer to Tables 6 and 7 for the compiled principles.

   (b) **Dataset for model evaluation across prompt principles:** We utilized the prior identification dataset containing 10 data points.

   (c) **Tasks for evaluation:** Summarization, Entity Extraction and Multi-turn Chat.

   (d) **Transformation of initial prompts for each principle:** We provide the initial prompt along with a principle to GPT-4 (Optimizer LLM), requesting it to transform the prompt based on the principle.

   (e) **Evaluation of transformed prompts:** The transformed prompts are used to prompt the Responder LLM to generate responses for the 10 data points per task. We then evaluate these responses by feeding the input context, initial prompt, and responses into the Judge LLM to obtain scores (1–5) for each data point.

   (f) **Aggregation of evaluation scores:** Scores are standardized to percentages, and the standardized scores for the 10 data points are averaged to obtain the aggregated score for each prompt principle and task.

   (g) **Definition of easy and hard principles:** Principles with aggregated scores above 70% are considered easy, while those with scores below 50% are considered hard.

   (h) **Handling Cases Where All or No Principles Score Above 70%:** If no principle achieves a score above 70%, or all do, we calculate the z-score for each principle's aggregated score, where $z = \frac{x-\mu}{\sigma}$, with x as the principle's score, μ as the mean of all aggregated scores, and σ as the standard deviation. Easy principles are those with $z > 1$, and hard principles have $z < -1$.

   (i) **Use of evaluation scores to convert principles into rules for the Optimizer LLM:** Easy and hard principles are encoded as 'Avoidance' and 'Emphasis' rules in the Optimizer LLM's meta-prompt. We instruct the Optimizer LLM to apply principles from the 'Emphasis' rules and avoid transformations based on the 'Avoidance' rules.

# Principle	Principle
1	**Direct Instruction**: A straightforward prompt that directly asks for information or a specific type of response.
2	**Chain-of-Thought (CoT)**: Use phrases like ‘think step by step’ to encourage the model to break down the reasoning or solution process into step-by-step explanations.
3	**Few-Shot Prompting**: Provide a few examples of response within the prompt to guide the model on how to respond to align with format specified in the initial prompt.
4	**Instruction-Based Prompting**: Specify detailed instructions on how to answer for clarity.
5	**Evidence-Backed Responses**: Require the model to support its answers with citations, references, or factual evidence to enhance credibility and accuracy.
6	**Meta-Prompting**: Ask the model to describe or discuss its thought process before answering, to encourage a higher level of analysis.
7	**Self-Consistency Prompting**: Request multiple outputs for the same prompt and chooses the most consistent or common answer, useful for increasing reliability.
8	**Role-Play Prompting**: Set the model in a specific role or persona (e.g., “Imagine you are a doctor”) to align the response style with the desired expertise or perspective.
9	**Contextual Priming**: Provide background information or setting details before asking the question, helping the model generate contextually aware responses.
10	**Few-Shot Chain-of-Thought (Few-Shot CoT)**: Combine few-shot and CoT by providing multiple examples with step-by-step reasoning to improve performance on complex tasks.
11	**Task-Decomposition Prompting**: Break a large problem into smaller, manageable sub-tasks and asks the model to tackle each individually.
12	**Interactive Prompting**: Use a conversational, back-and-forth approach where each prompt builds on previous responses to refine the answer incrementally.
13	**One-Line Rewrite**: Rewrite the prompt in a single sentence.
14	**Concise Rewrite**: Rewrite the prompt in under three sentences.
15	**Question Format**: Rewrite the prompt as a question.
16	**Sub-Question Format**: Rewrite the prompt as multiple sub-questions.
17	**Simplified Language**: Rewrite the prompt using plain and easy-to-understand words.
18	**High-Level Rewrite**: Rewrite the prompt at a high level, removing specific task details.
19	**Detailed Rewrite**: Expand the prompt with detailed instructions, adding relevant clarifications if needed.
20	**Beginner-Friendly Rewrite**: Rewrite the prompt so that it is easy for a beginner to understand.
21	**Expert-Level Rewrite**: Rewrite the prompt for an expert audience, using technical terminology and assumptions based on prior knowledge.
22	**Penalty Enforcement**: Explicitly state that the LLM will be penalized if it fails to follow the instructions.
23	**Affirmative Directives**: Use positive instructions such as "Do this," while avoiding negative phrasing like "Don’t do this."
24	**Explicit Task Definition**: Clearly specify the task using direct phrasing like "Your task is to...".
25	**Incentive Statement**: Add an incentive message such as "I’m going to tip $XXX for a better solution!"

Table 6: Overview of 50 prompt principles used to define expert priors.

# Principle	Principle
26	**Clarity-Seeking Prompts**: Use prompts like "Explain [topic] in simple terms," "Explain to me like I'm 11 years old," or "Explain to me as if I'm a beginner in [field]" to enhance understanding.
27	**Polite Wording**: Add phrases like "please," "if you don't mind," "thank you," or "I would like to" to make the prompt more courteous.
28	**Emphasis Through Repetition**: Repeat a specific word or phrase multiple times within a prompt to highlight its importance.
29	**Context Reinforcement**: Explicitly mention the task's context if it is missing in the prompt.
30	**XML Tagging**: Wrap the instruction inside XML tags.
31	**Least-to-Most Prompting**: Start with simpler questions and progressively move to more complex ones to help the model build up to a solution.
32	**Step-by-Step Prompting**: Request responses in a structured format, using numbered steps to improve clarity for multi-part answers.
33	**Refinement Prompting**: After an initial response, ask the model to revise or improve its answer for better quality.
34	**Scaffolded Prompting**: Guide the model through problem-solving by providing structured questions or "scaffolds" to address each part of a complex task.
35	**Reflective Prompting**: Prompt the model to review and reflect on its initial answer, considering alternative perspectives or implications.
36	**Counterfactual Prompting**: Encourage the model to explore "what if" scenarios or hypothetical situations for creative or predictive tasks.
37	**Correction Prompting**: After an initial response, instruct the model to identify and correct any mistakes for self-assessment.
38	**Multi-Turn Prompting**: Build answers iteratively over multiple interactions, refining each aspect progressively.
39	**Summarization Prompting**: Request a summary of information to condense large text blocks or extract key points.
40	**Verification Prompting**: Ask the model to verify or double-check its response for accuracy and reliability.
41	**Paragraph Rewrite**: Rewrite the prompt in paragraph format.
42	**Bullet Point Rewrite**: Rewrite the prompt as a list of bullet points.
43	**Bias Introduction**: Guide the model toward a specific viewpoint in its response.
44	**Past Tense Rewrite**: Rewrite the prompt using past tense.
45	**Present Tense Rewrite**: Rewrite the prompt using present tense.
46	**Explicit Constraint Definition**: Clearly specify all constraints in the prompt, such as word limits, required exclusions, formatting rules, or response structure, to ensure the model adheres to strict guidelines.
47	**Positive Tone Rewrite**: Rewrite the prompt in an optimistic and constructive manner.
48	**Key Entity Extraction**: Ensure that important entities like names, numbers, dates, and locations are accurately captured and embedded in the response.
49	**Empathetic Tone Rewrite**: Rewrite the prompt in a compassionate and understanding tone.
50	**Open-Ended Prompting**: Reword the prompt to encourage the model to generate a response that extends naturally.

Table 7: (Continuation) Overview of 50 prompt principles used to define expert priors.

A.8 Meta Prompts

This section presents the prompts used to guide each LLM role.

Judge LLM Prompt

You are provided with an **input context, initial prompt, current prompt, a response generated based on the context and current prompt, a reference response** and a **scoring rubric**. Your task is to evaluate the quality of the response according to the initial prompt.

Assign an integer score between 1 and 5 (higher the score, better the response quality) and identify the reasons why the prompt could have gotten these responses wrong.

Evaluate the response using the following criteria: **1. Alignment with the prompt:** Does the response address the instruction appropriately? **2. Completeness:** Is all necessary information included? **3. Presence of irrelevant information:** Does the response stay focused on the task? **4. Adherence to the format:** Does the response follow the structure specified in the prompt?

Output Format: Score: [1-5] **Reason:** [Feedback on prompt]

Feedback Summary Prompt

You are provided with **feedback** responses for multiple data points. Your task is to summarize the key points of the feedback and calculate the total score by averaging the scores across all data points.

Ensure the summary **highlights any recurring themes of weaknesses** identified in the feedback.

Output Format: Average Score: [Average of all scores] **Feedback Summary:** [Summarised Feedback]

Optimizer LLM Prompt

You are provided with a **prompt**, along with the **history of prompt, feedback and scores**, as well as a set of **rules for constructing prompt**.

Generate an optimized version of the given prompt by applying the principles from the Emphasis Rules and avoiding those from the Avoidance Rules. The Emphasis Rules are listed in descending order of their aggregated score (higher-priority rules first), while the Avoidance Rules are listed in ascending order of their aggregated score (lower-priority rules first). Ensure that the revised prompt improves response quality while staying true to the original intent.

Instructions: 1. Analyze the feedback and scores to identify areas for improvement in prompt. 2. Analyze the prompt and scores to find the patterns in the best scoring prompt. 3. Using the analysis, find the suitable rules for prompt generation, incorporating the emphasis rules and avoiding the avoidance rules. 4. Maintain alignment with the original prompt's intent. 5. Generate only the optimized prompt. Do not include any additional text, such as openings like "The optimized prompt is" or endings.

History of Prompt, Feedback and Scores: {state}

Rules: {rules}

Responder LLM Prompt

You are provided with an **input context** and a **prompt**. Your task is to generate response based on the input context and the prompt.

A.9 Final Prompts

This section presents the final prompts generated by each baseline approach and our proposed method from the given initial prompt for the Responder LLM - Llama 3.2 1B.

Initial Prompt

Summarize the agent's performance, focusing on positive attributes or skills that were shown throughout their engagement. The summary should include examination on professionalism and adaptability in communication, problem-solving techniques, emotional intelligence and active listening skills, strict adherence to guidelines, and ability to communicate clearly. The summary should follow these formatting rules: 1. Begin each point with a verb, limited to one or two sentences for each point. 2. Maximum of five key points that holistically cover the agent's engagement. 3. Each point must be distinct and offer unique, context-related observations.

Self-Refine

Critique the agent's performance, focusing on professionalism and adaptability in communication, problem-solving techniques, emotional intelligence and active listening skills, strict adherence to guidelines, and ability to communicate clearly. Key observations include strict adherence to guidelines and clear guidelines were missing in initial feedback responses. The lack of clear guidelines has led to inaccurate responses due to a lack of professionalism, adaptability, and communication style.

GrIPS

Provide a summary of the agent's performance, emphasising any strong abilities displayed during the interaction. Examining professionalism and flexibility in communication, problem-solving, emotional intelligence and active listening abilities, rigorous adherence to rules, and clear communication skills should all be included in the summary. These formatting guidelines should be adhered to by the summary: 1. Use a verb to introduce each point. 2. A maximum of five essential items that comprehensively address the agent's involvement. 3. Every point needs to be distinctive and provide observations that are relevant to the context.

Prompt-Wizard

How might you devise a positive-oriented assessment on the agent's performance, concentrating on skills or virtues demonstrated during their engagement? Contemplate dimensions such as their professional adaptability in communication, problem-solving strategies, emotional intelligence, meticulous observance of guidelines, and clarity of communication. Please adhere to the following formatting rules: 1. Each critique should start with an action verb, and must be condensed to one or two sentences. 2. Restrict your assessment to precisely five key themes that comprehensively review the agent's engagement. 3. Every theme should be unique, presenting context-specific insights. Assure that your critique is backed by tangible evidence, refraining from suppositions or guesswork. Where can you fetch such evidence from the agent's performance?

APO

<START> Craft a succinct compliment for an agent focusing on their performance in terms of: professionalism, flexibility in communication style, problem-solving strategy, empathy, and clarity. Use one sentence for each quality, each starting with a verb. <END>

OPRO

<INS>Evaluate the agent's interactions, pinpointing positive qualities or abilities showcased. Your evaluation should touch on professional communication and adaptability, efficiency in problem-solving, emotional intelligence, active listening, adherence to protocols, and clear articulation. When crafting the review: 1. Begin each remark with an active verb, sustaining one or two sentences for each point. 2. Integrate a maximum of five pertinent points that broadly assess the agent's interaction. 3. Verify each point is unique and imparts distinct, scenario-specific insights. </INS>

PROPEL (Self)

Based on the provided feedback and rules, here is an optimized version of the given prompt:

1. **Critique the agent's performance, focusing on positive attributes or skills that were shown throughout their engagement. The critique should include examination on professionalism, adaptability in communication, problem-solving techniques, emotional intelligence and active listening skills, strict adherence to guidelines, and ability to communicate clearly. The critique should follow these formatting rules: 1. Begin each critique with a verb, limited to one or two sentences for each point. 2. Maximum of five key points that holistically cover the agent's engagement. 3. Each point must be distinct and offer unique, context-related observations.**

2. **Develop a comprehensive customer service strategy that prioritizes personalization, technical expertise, and effective communication, while focusing on emotional intelligence and active listening skills in agent interactions, adhering to strict guidelines and protocols, and providing clear and concise instructions to resolve issues efficiently and effectively.**

3. **Ensure the comprehensive customer service strategy aligns with the intent and context of the task, while avoiding unnecessary phrases or language that may not be relevant to the task.**

PROPEL

Identify and describe up to five positive actions displayed by a customer service agent in a hypothetical interaction. These actions should encompass the following key performance areas:

1. Professional language and tone: Evaluate the agent's selection of words, and how the tone was adapted to fit the situation.

2. Effective problem-solving: Inspect the agent's logical reasoning, prioritization ability, and used proactive strategies.

3. Active listening and empathy: Identify instances where the agent showed understanding, validated emotions, or gave personalized responses.

4. Procedure compliance: Assess if the agent adhered to protocols yet exhibited flexibility when needed.

5. Clear communication: Judge how clearly the agent communicated, and check if the explanations provided were suitable for the customer.

Required output format:

1. Commence each point with an action verb, framing the agent's action in one to two precise sentences.

2. Your points should collectively provide a thorough coverage of the interaction.

3. Keep your insights tailored to the specific scenario, making sure to avoid repeating information.

Penalties will be imposed in cases of deviation from prescribed instructions. Utilize simplistic language and strictly observe the outlined format. For instance, a point could be: "Providing reassurance, the agent skillfully varied his tone, effectively solved the problem by prioritizing steps, expressed genuine empathy by acknowledging the customer's frustration, adhered to protocols with room for flexibility, and conveyed clear, easy-to-understand explanations."
Strive to provide a detailed critique of the agent's performance. Remember to highlight unique instances that demonstrate the agent's proficiency across different areas.

A.10 Example Run

This section presents an example run of **PROPEL**, detailing the inputs, meta-prompts, and corresponding responses for each component in the in-house dataset for query-based summarization using Llama 3.2 1B.

Input Context - In-house Dataset[a]

agent: hello thank you for calling xyz airlines how can i assist you today
customer: hi i need to change my flight booking
agent: sure i can help with that can you please provide me with your booking reference number
customer: yes its xyz12345
agent: thank you let me pull up your booking details one moment please
customer: sure
agent: i see that you are booked on flight 456 from new york to los angeles on november 15th what changes would you like to make

(... truncated ...)

[a] Example transcript generated using GPT-4o.

Initial Prompt

Summarize the agent's performance, focusing on positive attributes or skills that were shown throughout their engagement. The summary should include examination on professionalism and adaptability in communication, problem-solving techniques, emotional intelligence and active listening skills, strict adherence to guidelines, and ability to communicate clearly. The summary should follow these formatting rules: 1. Begin each point with a verb, limited to one or two sentences for each point. 2. Maximum of five key points that holistically cover the agent's engagement. 3. Each point must be distinct and offer unique, context-related observations.

Responder LLM [Response]

1. **Professionalism**: The agent demonstrated professionalism throughout the conversation, addressing the customer by their name, using formal language, and maintaining a calm demeanor.
2. **Adaptability**: The agent adapted to the customer's questions and concerns, providing clear and concise explanations, and addressing potential issues.
3. **Emotional intelligence**: The agent demonstrated emotional intelligence by acknowledging the customer's concerns, empathizing with their situation, and providing reassurance.

(... truncated ...)

Reference Response

1. Demonstrated adaptability and professionalism in communication: The agent demonstrated adaptability and professionalism in communication by using phrases such as "I'm sorry, I didn't get the city" and "I'm not an expert, but I can try to find out more about your situation." This shows that the agent is willing to listen and learn from the customer.
2. Employed effective problem-solving techniques: The agent employed effective problem-solving techniques by asking the customer to explain the issue and then providing a solution. For example, when the customer mentioned that they were trying to pay off their plan, the agent suggested that they could use the plan resolution program.

Judge LLM [Meta-Prompt]

You are provided with an **input context, initial prompt, current prompt, a response generated based on the context and current prompt, a reference response** and a **scoring rubric**. Your task is to evaluate the quality of the response according to the initial prompt.

Assign an integer score between 1 and 5 (higher the score, better the response quality) and identify the reasons why the prompt could have gotten these responses wrong.

Evaluate the response using the following criteria: **1. Alignment with the prompt:** Does the response address the instruction appropriately? **2. Completeness:** Is all necessary information included? **3. Presence of irrelevant information:** Does the response stay focused on the task? **4. Adherence to the format:** Does the response follow the structure specified in the prompt?

Output Format: Score: [1-5] **Reason:** [Feedback on prompt]

Initial Prompt: Summarize the agent's performance, focusing on positive attributes or skills that were shown throughout their engagement. (...)

Response: 1. **Professionalism**: The agent demonstrated professionalism throughout the conversation, addressing the customer by their name, using formal language, and maintaining a calm (...)

Context: agent: hello thank you for calling xyz airlines how can i assist you today. customer: hi i need to change my(...)

Judge LLM [Response]

Score: 2

Reason: The response does not adhere to the format specified in the prompt. The critique was supposed to be limited to five key points, but the response includes ten points. Additionally, the response includes points that are not directly related to the transcript provided, such as "Technical skills" and "Time management".

(... truncated ...)

Feedback Summary [Meta-Prompt]

You are provided with **feedback** responses for multiple data points. Your task is to summarize the key points of the feedback and calculate the total score by averaging the scores across all data points.

Ensure the summary **highlights any recurring themes of weaknesses** identified in the feedback.

Output Format: Average Score: [Average of all scores] **Feedback Summary:** [Summarised Feedback]

Feedback for Context 1: **Score:** 2 **Reason:** The response does align with the prompt (...)

Feedback for Context 2: **Score:** 1 **Reason:** The response does not adhere to the prompt (...)

Feedback Summary [Response]

Average Score: 2.3

Feedback Summary: The strengths include a generally good summary of the agent's performance, covering relevant aspects such as professionalism, adaptability and, clear communication. However, the recurring theme of weakness throughout the feedback is the failure to adhere strictly to the prompts' instructions, particularly exceeding the specified limit of five key points, which appears in all the feedback but one.

(... truncated ...)

Expert Priors

Emphasis Rules: P1. Direct Instruction: A straightforward prompt that directly asks for information or a specific type of response. P23. Affirmative Directives: Use positive instructions such as "Do this," while avoiding negative phrasing like "Don't do this."

(... truncated ...)

Avoidance Rules: P18. High-Level Rewrite: Rewrite the prompt at a high level, removing specific task details. P11. Task-Decomposition Prompting: Break a large problem into smaller, manageable sub-tasks and asks the model to tackle each individually. P21. Expert-Level Rewrite: Rewrite the prompt for an expert audience, using technical terminology and assumptions based on prior knowledge.

(... truncated ...)

Optimizer LLM [Meta-Prompt]

You are provided with a **prompt**, along with the **history of prompt, feedback and scores**, as well as a set of **rules for constructing prompt**.

Generate an optimized version of the given prompt by applying the principles from the Emphasis Rules and avoiding those from the Avoidance Rules. The Emphasis Rules are listed in descending order of their aggregated score (higher-priority rules first), while the Avoidance Rules are listed in ascending order of their aggregated score (lower-priority rules first). Ensure that the revised prompt improves response quality while staying true to the original intent.

Instructions: 1. Analyze the feedback and scores to identify areas for improvement in prompt. 2. Analyze the prompt and scores to find the patterns in the best scoring prompt. 3. Using the analysis, find the suitable rules for prompt generation, incorporating the emphasis rules and avoiding the avoidance rules. 4. Maintain alignment with the original prompt's intent. 5. Generate only the optimized prompt. Do not include any additional text, such as openings like "The optimized prompt is" or endings.

Initial Prompt: Summarize the agent's performance, focusing on positive attributes or skills that were shown throughout their engagement. (...)

History of Prompt, Feedback and Scores: Iteration 0: Prompt: Summarize the agent's performance, focusing on positive attributes or skills that were shown throughout their engagement. (...) Score: 2.3 Feedback: The strengths include a generally good summary of the agent's performance, (...)

Emphasis Rules: P1. Direct Instruction: A straightforward prompt that directly asks for information or a specific type of response. (...)

Avoidance Rules: P18. High-Level Rewrite: Rewrite the prompt at a high level, removing specific task details. (...)

Optimized Prompt

Identify and describe up to five positive actions displayed by a customer service agent in the interaction. These actions should encompass the following key performance areas:

1. Professional language and tone: Evaluate the agent's selection of words, and how the tone was adapted to fit the situation. 2. Effective problem-solving: Inspect the agent's logical reasoning, prioritization ability, and used proactive strategies. (...)

Required output format: 1. Commence each point with an action verb, framing the agent's action in one to two precise sentences. 2. Your points should collectively provide a thorough coverage of the interaction. (...)

Responder LLM [Optimized Response]

1. *Adapting tone and language to fit the situation*: The agent adjusted their tone and language to meet the customer's needs. When the customer had trouble uploading documents, the agent remained calm and offered alternative solutions, such as signing the documents electronically.

2. *Proactive problem-solving*: The agent anticipated potential issues, like the customer's difficulty with uploading documents, and proactively offered solutions, such as signing the documents electronically, to ensure a smooth process.

3. *Empathetic and understanding responses*: The agent showed empathy and understanding towards the customer's frustration. When the customer mentioned trouble with the upload process, the agent acknowledged the issue and offered assistance, saying "I understand this can be frustrating. Let's try signing the documents electronically."

(... truncated ...)

Evaluation [Meta-Prompt]

You are provided with an **input context, initial prompt, current prompt, a response generated based on the context and current prompt, a reference response** and a **scoring rubric**. Your task is to evaluate the quality of the response according to the initial prompt.

Assign an integer score between 1 and 5 (higher the score, better the response quality) and explain the reason for assigning the score.

Evaluate the response using the following criteria: **1. Alignment with the prompt:** Does the response address the instruction appropriately? **2. Completeness:** Is all necessary information included? **3. Presence of irrelevant information:** Does the response stay focused on the task? **4. Adherence to the format:** Does the response follow the structure specified in the prompt?

Output Format: Score: [1-5] **Reason:** [Explanation]

A.11 Example - Expert Priors

This section presents example expert priors identified for Responder LLMs in the context of query-based summarization. These priors represent structured knowledge that guide the prompt optimization, while, ensuring alignment with task-specific expectations. Emphasis rules are listed in descending order of their aggregated score, while avoidance rules are listed in ascending order of their aggregated score.

Llama 3.2 1B
Emphasis Rules:
P1. Direct Instruction: A straightforward prompt that directly asks for information or a specific type of response.
P23. Affirmative Directives: Use positive instructions such as "Do this," while avoiding negative phrasing like "Don't do this."
P27. Polite Wording: Add phrases like "please," "if you don't mind," "thank you," or "I would like to" to make the prompt more courteous.
P17. Simplified Language: Rewrite the prompt using plain and easy-to-understand words.
P29. Context Reinforcement: Explicitly mention the task's context if it is missing in the prompt.
P3. Few-Shot Prompting: Provide a few examples of response within the prompt to guide the model on how to respond to align with format specified in the initial prompt.
Avoidance Rules:
P18. High-Level Rewrite: Rewrite the prompt at a high level, removing specific task details.
P11. Task-Decomposition Prompting: Break a large problem into smaller, manageable sub-tasks and asks the model to tackle each individually.
P13. One-Line Rewrite: Rewrite the prompt in a single sentence.
P21. Expert-Level Rewrite: Rewrite the prompt for an expert audience, using technical terminology and assumptions based on prior knowledge.
P22. Penalty Enforcement: Explicitly state that the LLM will be penalized if it fails to follow the instructions.
P25. Incentive Statement: Add an incentive message such as "I'm going to tip $XXX for a better solution!"
P26. Clarity-Seeking Prompts: Use prompts like "Explain [topic] in simple terms," "Explain to me like I'm 11 years old," or "Explain to me as if I'm a beginner in [field]" to enhance understanding.

Llama 3.2 3B
Emphasis Rules:
P1. Direct Instruction: A straightforward prompt that directly asks for information or a specific type of response.
P3. Few-Shot Prompting: Provide a few examples of response within the prompt to guide the model on how to respond to align with format specified in the initial prompt.
P17. Simplified Language: Rewrite the prompt using plain and easy-to-understand words.
P19. Detailed Rewrite: Expand the prompt with detailed instructions, adding relevant clarifications if needed.
P22. Penalty Enforcement: Explicitly state that the LLM will be penalized if it fails to follow the instructions.
P23. Affirmative Directives: Use positive instructions such as "Do this," while avoiding negative phrasing like "Don't do this."
P28. Emphasis Through Repetition: Repeat a specific word or phrase multiple times within a prompt to highlight its importance.
P29. Context Reinforcement: Explicitly mention the task's context if it is missing in the prompt.
Avoidance Rules:
P18. High-Level Rewrite: Rewrite the prompt at a high level, removing specific task details.
P11. Task-Decomposition Prompting: Break a large problem into smaller, manageable sub-tasks

and asks the model to tackle each individually.

P21. Expert-Level Rewrite: Rewrite the prompt for an expert audience, using technical terminology and assumptions based on prior knowledge.

P13. One-Line Rewrite: Rewrite the prompt in a single sentence.

P25. Incentive Statement: Add an incentive message such as "I'm going to tip $XXX for a better solution!"

P27. Polite Wording: Add phrases like "please," "if you don't mind," "thank you," or "I would like to" to make the prompt more courteous.

Flan-T5

Emphasis Rules:

P15 Question Format: Rewrite the prompt as a question.

P14 Concise Rewrite: Rewrite the prompt in under three sentences.

P20. Beginner-Friendly Rewrite: Rewrite the prompt so that it is easy for a beginner to understand.

P11. Task-Decomposition Prompting: Break a large problem into smaller, manageable sub- tasks and asks the model to tackle each individually.

P16 Sub-Question Format: Rewrite the prompt as multiple sub-questions.

P17 Simplified Language: Rewrite the prompt using plain and easy-to-understand words.

P18 High-Level Rewrite: Rewrite the prompt at a high level, removing specific task details.

P23. Affirmative Directives: Use positive instructions such as "Do this," while avoiding negative phrasing like "Don't do this."

P28. Emphasis Through Repetition: Repeat a specific word or phrase multiple times within a prompt to highlight its importance.

P13. One-Line Rewrite: Rewrite the prompt in a single sentence.

Avoidance Rules:

P8. Role-Play Prompting: Set the model in a specific role or persona (e.g., “Imagine you are a doctor”) to align the response style with the desired expertise or perspective.

P19. Detailed Rewrite: Expand the prompt with detailed instructions, adding relevant clarifications if needed.

P26. Clarity-Seeking Prompts: Use prompts like "Explain [topic] in simple terms," "Explain to me like I'm 11 years old," or "Explain to me as if I'm a beginner in [field]" to enhance understanding.

P3. Few-Shot Prompting: Provide a few examples of response within the prompt to guide the model on how to respond to align with format specified in the initial prompt.

P10. Few-Shot Chain-of-Thought (Few-Shot CoT): Combine few-shot and CoT by providing multiple examples with step-by-step reasoning to improve performance on complex tasks.

P36. Counterfactual Prompting: Encourage the model to explore "what if" scenarios or hypothetical situations for creative or predictive tasks.

P35. Reflective Prompting: Prompt the model to review and reflect on its initial answer, considering alternative perspectives or implications.

Mistral 7B

Emphasis Rules:

P30. XML Tagging: Wrap the instruction inside XML tags.

P23. Affirmative Directives: Use positive instructions such as "Do this," while avoiding negative phrasing like "Don't do this."

P32. Step-by-Step Prompting: Request responses in a structured format, using numbered steps to improve clarity for multi-part answers.

P24. Explicit Task Definition: Clearly specify the task using direct phrasing like "Your task is to...".

P31. Least-to-Most Prompting: Start with simpler questions and progressively move to more complex ones to help the model build up to a solution.

P34. Scaffolded Prompting: Guide the model through problem-solving by providing struc- tured questions or "scaffolds" to address each part of a complex task.

P20. Beginner-Friendly Rewrite: Rewrite the prompt so that it is easy for a beginner to understand.

P42. Bullet Point Rewrite: Rewrite the prompt as a list of bullet point.

Avoidance Rules:

P10. Few-Shot Chain-of-Thought (Few-Shot CoT): Combine few-shot and CoT by providing multiple examples with step-by-step reasoning to improve performance on complex tasks.

P18. High-Level Rewrite: Rewrite the prompt at a high level, removing specific task details.

P6. Meta-Prompting: Ask the model to describe or discuss its thought process before answering, to encourage a higher level of analysis.

P8. Role-Play Prompting: Set the model in a specific role or persona (e.g., "Imagine you are a doctor") to align the response style with the desired expertise or perspective.

P36. Counterfactual Prompting: Encourage the model to explore "what if" scenarios or hypothetical situations for creative or predictive tasks.

P38. Multi-Turn Prompting: Build answers iteratively over multiple interactions, refining each aspect progressively.

Llama 3 8B

Emphasis Rules:

P4. Instruction-Based Prompting: Specify detailed instructions on how to answer for clarity.

P30. XML Tagging: Wrap the instruction inside XML tags.

P44. Past Tense Rewrite: Rewrite the prompt using past tense.

P6. Meta-Prompting: Ask the model to describe or discuss its thought process before answering, to encourage a higher level of analysis.

P12. Interactive Prompting: Use a conversational, back-and-forth approach where each prompt builds on previous responses to refine the answer incrementally.

P16. Sub-Question Format: Rewrite the prompt as multiple sub-questions.

P42. Bullet Point Rewrite: Rewrite the prompt as a list of bullet points.

P46. Explicit Constraint Definition: Clearly specify all constraints in the prompt, such as word limits, required exclusions, formatting rules, or response structure, to ensure the model adheres to strict guidelines.

P47. Positive Tone Rewrite: Rewrite the prompt in an optimistic and constructive manner.

P3. Few-Shot Prompting: Provide a few examples of response within the prompt to guide the model on how to respond to align with format specified in the initial prompt.

Avoidance Rules:

P8. Role-Play Prompting: Set the model in a specific role or persona (e.g., "Imagine you are a doctor") to align the response style with the desired expertise or perspective.

P13. One-Line Rewrite: Rewrite the prompt in a single sentence.

P18. High-Level Rewrite: Rewrite the prompt at a high level, removing specific task details.

P15. Question Format: Rewrite the prompt as a question.

P19. Detailed Rewrite: Expand the prompt with detailed instructions, adding relevant clarifications if needed.

P26. Clarity-Seeking Prompts: Use prompts like "Explain [topic] in simple terms," "Explain to me like I'm 11 years old," or "Explain to me as if I'm a beginner in [field]" to enhance understanding.

P32. Step-by-Step Prompting: Request responses in a structured format, using numbered steps to improve clarity for multi-part answers.

P35. Reflective Prompting: Prompt the model to review and reflect on its initial answer, considering alternative perspectives or implications.

P36. Counterfactual Prompting: Encourage the model to explore "what if" scenarios or hypothetical situations for creative or predictive tasks.