PAFT: Prompt-Agnostic Fine-Tuning

Anonymous ACL submission

Abstract

While Large Language Models (LLMs) adapt well to downstream tasks after fine-tuning, this adaptability often compromises prompt robustness, as even minor prompt variations can significantly degrade performance. To address this, we propose *prompt-agnostic fine-tuning* (PAFT), a simple yet effective approach that dynamically adjusts prompts during fine-tuning. This encourages the model to learn underlying task principles rather than overfitting to specific prompt formulations. PAFT operates in two stages: First, a diverse set of meaningful, synthetic candidate prompts is constructed. Second, during fine-tuning, prompts are randomly sampled from this set to create dynamic training inputs. Extensive experiments across diverse datasets and LLMs demonstrate that models trained with PAFT exhibit strong robustness and generalization across a wide range of prompts, including unseen ones. This enhanced robustness improves both model performance and inference speed while maintaining training efficiency. Ablation studies further confirm the effectiveness of PAFT.

1 Introduction

Large language models (LLMs) have demonstrated remarkable success across a diverse range of natural language processing (NLP) tasks (Zhao et al., 2024; Xu et al., 2023). To further enhance the performance of LLMs on specific downstream tasks, supervised fine-tuning (SFT) has emerged as a widely adopted strategy (Ouyang et al., 2022; Devlin et al., 2019). This approach typically involves augmenting input data with task-specific instructions and constructing dialogue datasets with expected outputs, enabling the model to effectively learn task-specific patterns during fine-tuning. Empirical studies have shown that SFT can substantially improve model performance on downstream tasks (Raffel et al., 2023; Hu et al., 2023b; Wei et al., 2022). However, a critical limitation of this paradigm is its reliance on fixed instruction templates (Mishra et al., 2022; Chung et al., 2022) for each downstream task. This rigidity often leads to overfitting, whereby models become excessively dependent on specific instruction patterns (Zhang et al., 2024; Kung and Peng, 2023). Consequently, during inference on downstream tasks, even minor deviations between user-provided instructions and the training instructions can result in significant performance degradation (Mialon et al., 2023; Raman et al., 2023). This issue is particularly pronounced when LLM practitioners, who may lack domain expertise, provide prompts that deviate substantially from those used during SFT. In such scenarios, carefully fine-tuned models may experience drastic performance drops, occasionally approaching random guessing levels (Voronov et al., 2024). Previous research has primarily focused on prompt tuning—introducing trainable vectors (soft prompts) to optimize performance (Liu et al., 2022; Li and Liang, 2021; Lester et al., 2021)—however,

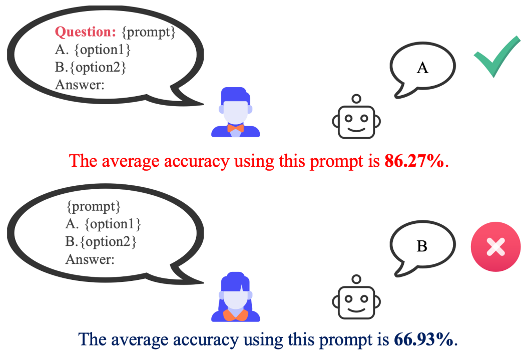

Figure 1: This figure shows how small changes in prompts can drastically affect the accuracy of a model. Two examples show the same user question, but the prompts differ by only one word, resulting in different answers. The first prompt achieves 86.27% accuracy across the entire dataset, while the second prompt drops significantly to 66.93%. This highlights how even small modifications can lead to large swings in performance if a model lacks prompt robustness.

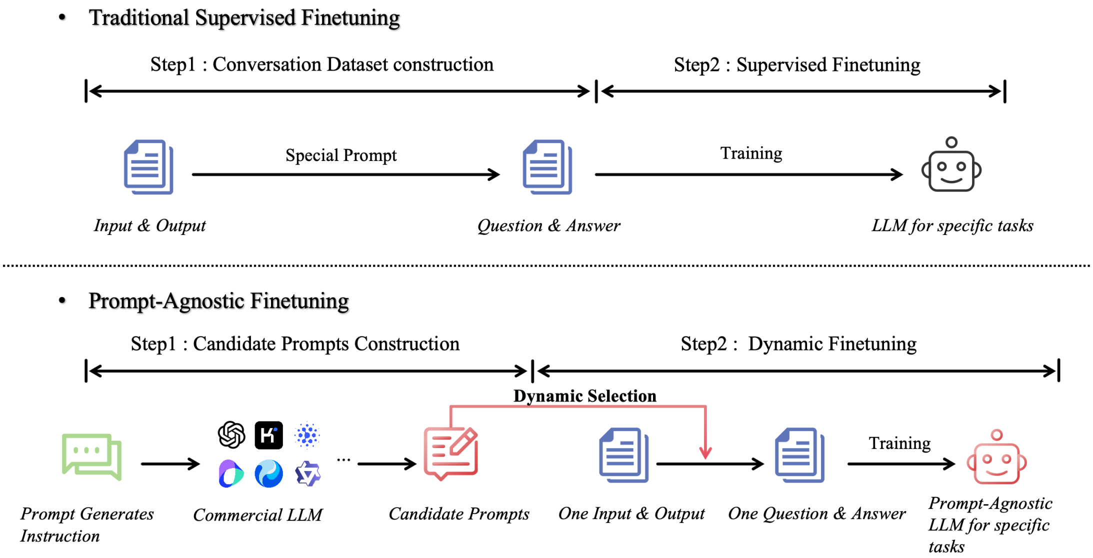

Figure 2: An overview of PAFT: This figure compares Traditional Supervised Fine-tuning (SFT) and Prompt-Agnostic Fine-Tuning (PAFT), highlighting their main differences. SFT relies on a fixed dataset and predefined prompts, which limits its robustness and generalization to different prompts. In contrast, PAFT dynamically selects prompts during training, which improves robustness and generalization to a wide range of prompts. By leveraging a commercial LLM to generate candidate prompts, PAFT provides a more general and scalable solution.

these methods inadvertently increase sensitivity to prompt variations (Wen et al., 2023; Qin and Eisner, 2021), resulting in significant performance fluctuations and increased costs associated with prompt engineering (Han et al., 2024; Longpre et al., 2023). Prompt robustness in SFT has received limited attention, with most existing work focusing on in-context learning (Zhu et al., 2024; Shi et al., 2024; Ishibashi et al., 2023).

To address this critical gap, we present PAFT, an innovative fine-tuning framework designed to dynamically adapt to diverse prompts during training. To our knowledge, this is the first systematic approach to enhancing prompt robustness in SFT, a vital yet under-explored area. Unlike traditional methods, which often overfit to specific prompt patterns, PAFT enables models to grasp underlying task semantics, ensuring robust performance across various human-written prompts.As shown in Figure 2, PAFT operates in two phases: (1) Candidate Prompt Construction (Section 4.1) and (2) Dynamic Fine-Tuning (Section 4.2). Initially, a diverse set of high-quality synthetic prompts is generated, capturing essential task semantics while maintaining linguistic variability. During fine-tuning, a dynamic prompt sampling strategy is employed, randomly selecting prompts from our curated set to expose the model to a wide range of formulations. Extensive evaluations reveal that PAFT achieves three primary objectives: (1) significantly boosting model robustness and generalization across diverse prompts; (2) maintaining state-of-the-art performance on downstream tasks; and (3) potentially enhancing inference speed while preserving training efficiency. These findings indicate that PAFT represents a promising direction for developing more robust and user-friendly language models. Our key contributions are: (a) Through comprehensive experiments, we demonstrate that fine-tuning with fixed prompts significantly undermines the model's robustness to prompt variations, leading to poor generalization on unseen prompts and severe performance degradation; and (b) We propose PAFT, comprising candidate prompt construction and dynamic fine-tuning, a novel approach to enhance the prompt robustness of fine-tuned models. This approach ensures consistent and robust performance across a variety of test prompts, including those not encountered during training.

2 Related Work

Prompt Optimization Effective prompt engineering is crucial for maximizing LLM performance, motivating various optimization techniques (Chang et al., 2024; Li, 2023; Diao et al., 2023; Sun et al., 2022). Methods like INSTINCT (Lin et al., 2024) utilize neural bandits and LLM embeddings for efficient prompt search, while ZOPO (Hu et al.,

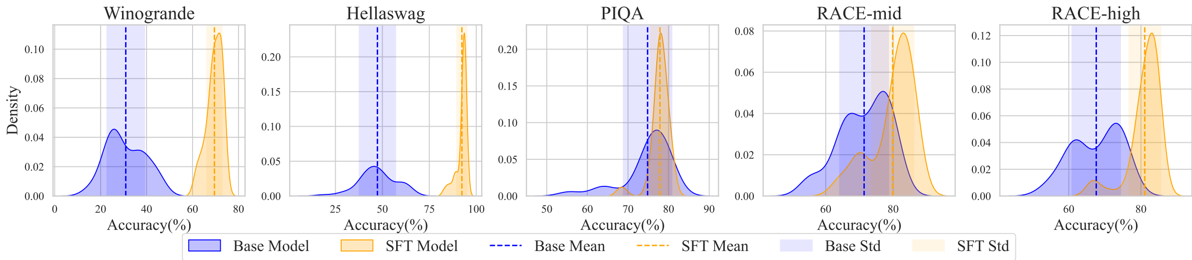

Figure 3: This figure presents the results of preliminary experiments conducted on four datasets to evaluate the accuracy of the base model and the SFT model across over 450 diverse prompts. The probability distribution plots illustrate the distribution of accuracy for models. The results show that while the SFT model has an overall improvement in accuracy compared to the base model, the accuracy of some prompts is still relatively low, and the standard deviation of the SFT model is high, indicating that the accuracy varies greatly between different prompts, which highlights the impact of prompt design and the need for further optimization through model fine-tuning.

2024) improves efficiency through localized search. BATprompt (Shi et al., 2024) incorporates robustness considerations in in-context learning by leveraging natural language perturbations. However, these methods often suffer from prompt fragility, exhibiting high sensitivity to even minor prompt alterations, particularly after fine-tuning. This limits LLM generalization in real-world applications. Our work addresses this limitation by prioritizing robustness across diverse prompt formulations, rather than optimizing for a single prompt.

Supervised Fine-Tuning (SFT) SFT is a dominant paradigm for adapting LLMs, valued for its efficiency. Two main SFT approaches exist: soft prompt tuning (optimizing continuous vectors prepended to the input while freezing base model parameters) (Li and Liang, 2021; Liu et al., 2022), and full/parameter-efficient fine-tuning (PEFT) (Shu et al., 2024; Ouyang et al., 2022; Liu et al., 2021; Lester et al., 2021). Among PEFT techniques, Low-Rank Adaptation (LoRA) (Hu et al., 2022) is widely used, freezing pre-trained parameters and introducing low-rank trainable matrices. Advanced LoRA variants further aim to mitigate overfitting and enhance generalization (Chen et al., 2023; Si et al., 2024; Wei et al., 2024). However, these methods, while mitigating parameter-level overfitting, typically rely on fixed training prompts, thus neglecting prompt robustness. This is particularly problematic for soft prompt tuning, where models exhibit high sensitivity to prompt variations. Consequently, minor deviations from training prompts can drastically degrade performance. To address this, we propose PAFT, a novel framework that prioritizes prompt robustness while preserving computational advantages. By decoupling model performance from specific prompt formulations, PAFT significantly enhances the adaptability and reliability of fine-tuned models.

3 Preliminaries

To systematically study the impact of prompt variations on fine-tuned models, we use LoRA (Hu et al., 2022) as an illustrative example and conduct comprehensive preliminary experiments on multiple downstream tasks to assess prompt sensitivity and robustness. These tasks include natural language inference, question answering, and reading comprehension, using the LLaMA3-8B (Meta, 2024) model. We constructed a comprehensive set of over 450 prompts, covering a wide range of language styles, task-specific instructions, and formatting variations. Figure 3 presents a statistical analysis of the accuracy distribution for both the base model and SFT model across these prompts, revealing a key finding: prompt selection significantly influences model performance, with considerable accuracy variation observed across prompts, irrespective of the downstream task. Only a small fraction (typically less than 10%) of prompts yields near-optimal performance; some even degrade accuracy to near-random levels. Minor prompt modifications (e.g., rephrasing, punctuation, reordering) induce substantial fluctuations. For example, the addition of "Question" improves accuracy by 20% (Figure 1). This sensitivity highlights the fragility of current fine-tuning methods and their strong dependence on specific prompt formulations. These findings align with prior work (He et al., 2024; Voronov et al., 2024; Salinas and Morstatter, 2024; Min et al., 2022; Gao et al., 2021); however, we demonstrate that this sensitivity persists across tasks, suggesting a fundamental limitation of current PEFT paradigms. Motivated by these findings,

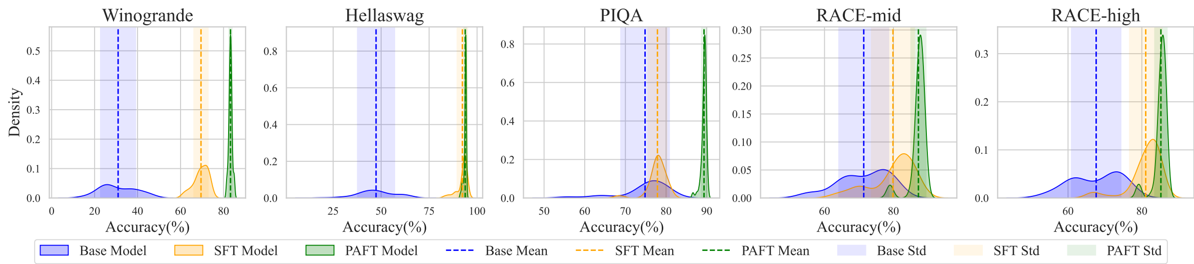

Figure 4: The performance of the base model, the SFT model, and the PAFT model is compared on multiple reasoning and reading comprehension tasks. This is a visual comparison to Figure 3 to illustrate the effectiveness of PAFT, where the probability distribution plots show the distribution of accuracy of different models on the test prompts that were not used during PAFT training. The PAFT model shows superior performance compared to the base model and the SFT model, achieving higher accuracy and lower variance in all tasks.

we propose PAFT, addressing prompt robustness by decoupling performance from specific formulations, ensuring consistent results across diverse prompts, and significantly enhancing the practical applicability of fine-tuned models in real-world scenarios where prompt variations are inevitable.

4 The PAFT Framework

To improve the prompt robustness of LLMs, we propose the PAFT framework in Figure 2. As shown in Figure 2, the PAFT framework consists of two key stages: candidate prompt construction (see Section 4.1 for details) and dynamic fine-tuning (see Section 4.2 for details).

4.1 Candidate Prompt Construction

To ensure the robustness and effectiveness of PAFT across diverse prompts, we design a comprehensive prompt construction framework that aims to generate diverse and meaningful candidate prompts efficiently, enabling the model to generalize across different prompt formats. Our approach leverages the powerful generative capabilities of LLMs (Kohl et al., 2024) and comprises three key phases: First, recognizing the inherent variability in how different LLMs interpret downstream tasks due to variations in pre-training data, model architectures, and optimization objectives (Minaee et al., 2024; Zhao et al., 2024), we employ a multi-model approach, selecting 10 mainstream LLMs according to their generation capabilities, including models from OpenAI et al. (2024); Bai et al. (2023); Ouyang et al. (2022), and other widely used commercial LLMs, for prompt generation. This diverse selection ensures broad coverage of potential prompt formulations, capturing variations in linguistic style, task interpretation, and instructional clarity, thereby mitigating biases towards any single model's prompt generation tendencies. Second, we employ a dual-strategy approach, combining few-shot and zero-shot techniques to balance prompt quality and diversity. For few-shot prompting, we leverage principles from in-context learning, providing each LLM with carefully curated, human-crafted examples to guide the generation of semantically coherent and task-relevant prompts, ensuring meaningfulness and alignment with the intended task. For zero-shot prompting, we prioritize diversity by allowing LLMs to generate prompts without explicit examples, thus encouraging a wider range of linguistic styles, structural variations, and task formulations. Specifically, we generate 20 prompts using each strategy, resulting in a comprehensive set encompassing both high-quality prompts (derived from few-shot prompting) and diverse, potentially less optimal prompts (derived from zero-shot prompting). This balanced approach exposes the model to a realistic distribution of prompt quality during training, thereby enhancing its robustness to real-world scenarios where prompt quality may vary significantly. Finally, to rigorously evaluate the robustness of PAFT, we randomly partition the generated prompts into training and test sets using an 8:1 ratio. Crucially, the training and test sets contain entirely distinct prompts, ensuring evaluation on completely unseen formulations. This partitioning strategy enables the construction of training data that exposes the model to a wide range of prompt styles while providing a robust testbed for assessing generalization to novel prompts. By decoupling training and test prompts, we confirm that performance improvements reflect a genuine ability to handle diverse and unseen prompt formulations, rather than overfitting to specific prompt patterns. This comprehensive framework ensures that PAFT learns task semantics independently of specific prompt patterns, enabling effective generalization across a wide range of real-world scenarios, and provides a scalable and cost-effective solution for improving prompt robustness in LLMs.

4.2 Dynamic Fine-Tuning

The dynamic fine-tuning process in our PAFT framework is designed to enhance the robustness of LLMs to diverse prompt formulations while preserving high performance on downstream tasks. As illustrated in Algorithm 1, during each training epoch t, a prompt p is randomly sampled from a diverse set of synthetically generated candidate prompts $\mathbb{P}$ (line 4 in Algorithm 1), ensuring exposure to a wide range of linguistic styles and task formulations. For each data point $(x, y) \in \mathbb{D}$ (line 6 in Algorithm 1), the selected prompt p is reused for K consecutive training steps (lines 7-9 in Algorithm 1), and the input $\mathbf{I} = \text{InputConstruction}(x, p)$ is constructed by combining the prompt p with the data point x (line 7 in Algorithm 1). The model parameters θ are then updated using stochastic gradient-based optimization methods, such as SGD (Sra et al., 2011) or AdamW (Loshchilov and Hutter, 2019) (line 8 in Algorithm 1), enabling the model to learn task-specific semantics while adapting to the formulation of prompt. After every K steps, a new prompt is sampled from $\mathbb{P}$ to replace the current one (lines 10-11 in Algorithm 1), ensuring that the model is exposed to multiple prompts within a single epoch. At the end of each epoch, the model parameters θ_{t+1}^0 are initialized with the final parameters from the previous epoch, θ_t^K (line 12 in Algorithm 1), ensuring continuity in the learning process. After T epochs, the fine-tuned model parameters $\theta^* = \theta_T$ achieve consistent performance across a wide range of prompts (line 16 in Algorithm 1), including those not encountered during training. This makes PAFT particularly suitable for real-world applications where prompt quality and style may vary significantly, such as when users lack domain expertise or when prompts are generated automatically. By decoupling model performance from fixed prompt formulations, PAFT addresses a key limitation of traditional fine-tuning methods, ensuring robust performance without requiring extensive prompt engineering. The dynamic fine-tuning strategy enhances both the robustness and generalization of fine-tuned models while maintaining computational efficiency, making it a practical solution for improving the adaptability of LLMs in diverse settings.

Algorithm 1 The PAFT Framework

1: **Input:** Generate a good candidate prompt training set $\mathbb{P}$; A task-specific dataset $\mathbb{D}$; The number of training epochs T; The number of same prompt training K; Initialized trainable parameters θ_0^0; Learning rate η_θ
2: **Output:** Fine-tuned model parameters θ^*.
3: **for** each epoch $t = 0$ **to** $T - 1$ **do**
4: $\quad p \leftarrow \text{RandomlySample}(\mathbb{P})$ {Randomly select a prompt from the candidate set}
5: $\quad k \leftarrow 0$ {Initialize the step counter}
6: $\quad$ **for** each data point $(x, y) \in \mathbb{D}$ **do**
7: $\quad\quad \mathbf{I} \leftarrow \text{InputConstruction}(x, p)$ {Construct input using prompt p and data x}
8: $\quad\quad \theta_t^{k+1} \leftarrow \theta_t^k - \eta_\theta \nabla_\theta \ell(\theta, \mathbf{I})|_{\theta=\theta_t^k}$ {Update model parameters}
9: $\quad\quad k \leftarrow k + 1$ {Increment the step counter}
10: $\quad\quad$ **if** $k \mod K == 0$ **then**
11: $\quad\quad\quad p \leftarrow \text{RandomlySample}(\mathbb{P})$ {Update prompt every K steps}
12: $\quad\quad$ **end if**
13: $\quad$ **end for**
14: $\quad \theta_{t+1}^0 \leftarrow \theta_t^k$ {Carry over parameters to the next epoch}
15: **end for**
16: **return** $\theta^* = \theta_T$ {Return the final fine-tuned parameters}

5 Empirical Results

In this section, we conduct extensive experiments to evaluate the effectiveness and efficiency of our proposed PAFT framework. We begin by detailing the datasets and experimental setup in Section 5.1, followed by a comprehensive analysis of the main results in Section 5.2. Additionally, we perform ablation studies to investigate the impact of key components of our framework, as discussed in Section 5.3.

5.1 Datasets and Setup

To evaluate the performance of our proposed PAFT method, we focus on reasoning and reading comprehension tasks, as these domains are particularly susceptible to prompt variations. As PAFT is the first work to address the prompt robustness problem in large language models (LLMs) through training, we generate task-specific candidate prompts for each downstream task. Following the dataset selection process of Hu et al. (2023a); Wei et al. (2024), we select the Winogrande (Sakaguchi et al., 2019), PIQA (Bisk et al., 2019), and Hellaswag (Zellers et al., 2019) reasoning benchmarks and additionally include the RACE (Lai et al., 2017) reading comprehension benchmark. These datasets are widely recognized for their ability to assess reasoning and comprehension, provide independent training, val-

Table 1: Performance comparison of different fine-tuning methods on the test prompt sets across various reasoning and reading comprehension tasks using the LLaMA3-8B (Meta, 2024) with LoRA rank 8. Results are reported as average accuracy, standard deviation, and percentage of test prompts exceeding a specific score threshold (90% for Hellaswag, 80% for Winogrande, and 85% for other datasets). The **Base Model** represents the pre-trained model without fine-tuning, **user-specified prompt** (Wei et al., 2024) refers to fine-tuning with LoRA using human-designed prompts, **TopAccuracy prompt** refers to fine-tuning with LoRA using the prompt exhibiting the highest accuracy on the training set, **BATprompt** refers to fine-tuning with LoRA using the most robust prompt generated by BATprompt (Shi et al., 2024), and **ZOPO prompt** refers to fine-tuning with LoRA using the optimal prompt selected by ZOPO (Hu et al., 2024) from the training prompt set. **PAFT** (our proposed method) demonstrates superior performance, achieving the highest accuracy and lowest variance across all tasks. The last rows show the comparison of PAFT with the second-best performing method (underlined). The Top column indicates the percentage of test prompts with a correct rate of 90% for Hellaswag, 80% for Winogrande, and 85% for other datasets.

Methods	**Hellaswag**			**PIQA**			**Winogrande**			**RACE-mid**			**RACE-high**			**Average**		
Metric	Mean	Std	Top	Mean	Std	Top	Mean	Std	Top	Mean	Std	Top	Mean	Std	Top	Mean	Std	Top
Base Model	47.36	±9.78	0%	74.68	±6.24	0%	45.15	±11.78	0%	71.39	±7.33	0%	67.62	±6.78	0%	61.24	±8.38	0%
user-specified prompt	92.35	±2.78	0%	77.87	±2.36	0%	78.16	±7.97	0%	79.88	±6.32	22%	81.05	±4.45	4%	81.86	±4.78	5%
TopAccuracy prompt	91.27	±2.79	86%	75.96	±3.89	0%	66.77	±3.94	0%	84.81	±4.06	59%	82.45	±3.26	14%	80.25	±3.63	32%
BATprompt	90.30	±1.79	78%	83.41	±1.74	16%	69.01	±4.45	0%	83.92	±5.38	65%	81.33	±4.21	12%	81.56	±3.51	34%
ZOPO prompt	92.46	±2.43	86%	83.52	±2.23	27%	74.75	±3.81	0%	83.50	±5.05	51%	82.36	±4.53	35%	83.32	±3.61	40%
PAFT	**93.83**	**±0.70**	**100%**	**89.33**	**±0.63**	**100%**	**82.09**	**±0.81**	**100%**	**87.26**	**±2.23**	**94%**	**85.17**	**±1.71**	**73%**	**87.57**	**±1.57**	**94%**
PAFT Improvement	+1.37	-1.09	14%	+5.81	-1.11	73%	+3.93	-3.00	100%	+2.45	-1.83	29%	+2.72	-1.55	38%	+4.25	-1.94	54%

Table 2: Comparison of inference time (in hours) for different fine-tuning methods. The base model represents the pre-trained model without fine-tuning, while the other rows show the inference time of models fine-tuned with LoRA using different prompts. PAFT shows better inference efficiency than other methods. The last line shows the multiple of PAFT improvement.

Inference time/h	**Hellaswag**	**PIQA**	**Winogrande**	**RACE**	**Average**
Base Model	3.97	1.35	1.72	6.24	3.32
user-specified prompt	6.52	0.98	3.27	8.23	4.75
TopAccuracy prompt	5.75	1.13	2.76	7.56	4.30
BATprompt	4.57	1.57	3.14	7.98	4.32
ZOPO prompt	5.12	0.87	3.23	8.28	4.38
PAFT	**1.19**	**0.39**	**0.45**	**2.08**	**1.02**
PAFT Improvement	×3.3	×2.23	×3.82	×3.00	×3.25

idation, and test sets, and employ accuracy as the performance metric. As described in Section 4.1, we generate a diverse set of 400 training prompts and 50 test prompts, ensuring that the test prompts are distinct from the training prompts, see the Appendix C for details. This separation rigorously evaluates the ability of model to generalize to unseen prompt formulations. We establish five baselines for comparison to isolate the impact of prompt engineering on fine-tuning: the pre-trained model without fine-tuning (Base Model); fine-tuning with human-designed prompts (User-Specified Prompt) as in Wei et al. (2024); fine-tuning with the prompt exhibiting the highest accuracy on the training set (Top-Accuracy Prompt); fine-tuning with the most robust prompt generated by BATprompt (Shi et al., 2024) (BATprompt); and fine-tuning with the optimal prompt selected by ZOPO (Hu et al., 2024) from the training prompt set (ZOPO Prompt). The key distinction between these methods lies in the prompt selection for fine-tuning. Critically, all models, including the baselines, are evaluated using the same set of 50 test prompts. This consistent evaluation protocol allows us to directly compare performance consistency and variation across methods. Our implementation leverages the Llama-factory framework (Zheng et al., 2024) and is evaluated using the Opencompass framework (Contributors, 2023). Detailed experimental configurations are provided in Appendix A. All experiments are conducted on NVIDIA A100, V100, 4090, and L40 GPUs to ensure efficient and scalable evaluation.

5.2 Main Results

PAFT demonstrates strong prompt robustness As shown in Table 1, Figure 4, and Figure 6, PAFT exhibits remarkably low variance across all evaluation tasks, indicating excellent prompt robustness. Compared to other methods, PAFT achieves significantly lower variance, attributable to its unique dynamic prompt selection strategy. This strategy continuously adjusts the prompt during training, compelling the model to learn essential task features rather than overfitting to a specific prompt format. This contrasts sharply with the other baseline models. User-specified prompts rely on manually designed prompts, making it challenging to ensure both quality and diversity, especially without domain expertise. While TopAccuracy and ZOPO select the prompt exhibiting the highest accuracy on the training set, they are prone to overfitting to specific prompts and exhibit poor generalization. Although BATprompt also considers prompt

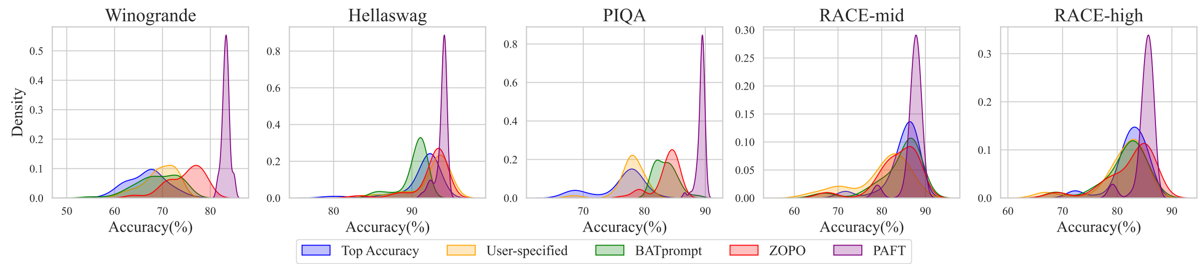

Figure 5: The performance of TopAccuracy, User-specified, BATprompt, ZOPO, and PAFT models is compared on multiple reasoning and reading comprehension tasks. Results are reported in terms of their correct distribution. The tests are conducted on a test set of 50 unseen prompts, different from the ones used in training. The PAFT model shows superior performance compared to other baselines, achieving higher accuracy and lower variance in all tasks.

robustness, its generated robust prompts are less effective than PAFT. In summary, the low variance of PAFT implies more stable performance and stronger generalization across diverse prompts, leading to higher reliability in practical applications. Specifically, models trained with PAFT can be used to develop more user-friendly question-answering systems, agent systems independent of input-output formats, and even to better decouple LLM capabilities from prompts, enabling more accurate LLM evaluation. PAFT achieves top performance on the majority of prompts, significantly outperforming all baselines (Table 1, Top column). Furthermore, PAFT maintains high training efficiency, A detailed discussion of training efficiency is provided in Appendix B.

PAFT achieves state-of-the-art performance As shown in Table 1, Figure 4, and Figure 6, PAFT achieves the highest average accuracy across all evaluated reasoning and reading comprehension tasks, significantly outperforming other baseline models. Specifically, PAFT surpasses other methods on tasks such as HellaSwag, PIQA, Winogrande, RACE, demonstrating its excellent performance across diverse natural language processing tasks. This superior performance stems from PAFT's prompt robustness, enabling the model to better grasp the core essence of each task and maintain high performance across diverse prompt formulations. For instance, strong performance of PAFT on the open text generation task (HellaSwag) can be attributed to its dynamic prompt selection strategy, facilitating improved capture of contextual information. Its success on the physical common sense reasoning task (PIQA) can be attributed to its enhanced ability to utilize common sense knowledge. Similarly, its performance on the reference resolution task (Winogrande) can be attributed to its improved understanding of sentence structure and semantic relations, while its success on the reading comprehension task (RACE) can be attributed to its improved capture of topic and key information. In essence, this performance gain arises from PAFT's decoupling of the prompt from the task itself, allowing the model to focus on learning the fundamental aspects of the downstream tasks.

PAFT enhances inference efficiency In addition to robustness and performance, PAFT also significantly enhances inference efficiency. By fundamentally enhancing the ability of model to understand the core semantics of tasks, PAFT enables the model to solve problems more effectively, generating fewer tokens. This capability directly translates to faster inference speeds, as the model avoids redundant or unnecessary outputs and focuses on concise, accurate responses. To quantify this improvement, we measured the average end-to-end inference time across all test prompts and datasets, from the input prompt to the final output. As shown in Table 2, models trained with PAFT consistently achieve the fastest inference speeds compared to the baseline methods. This improvement is a direct result of PAFT's inherent prompt robustness. By decoupling model performance from the specific prompt wording, PAFT operates consistently and efficiently regardless of the input prompt. In essence, PAFT promotes more effective generalization and eliminates the need for prompt-specific adaptation during inference. Additionally, our training regime covers a wide range of prompt wordings, avoiding the potential performance degradation or increased computation typically required to handle unexpected or unevenly distributed prompts during inference. This consistency and efficiency is especially valuable in real-world applications that require fast response times, such as dialogue sys-

Table 3: Performance comparison of PAFT with varying hyperparameters K (number of iterations per prompt) and T (number of epochs) across multiple reasoning and reading comprehension tasks. Results are reported as mean accuracy ($\pm$ standard deviation) on the Hellaswag, PIQA, Winogrande, RACE-mid, and RACE-high datasets. The best results for each metric are highlighted in bold.

# K and T	**Hellaswag**	**PIQA**	**Winogrande**	**RACE-mid**	**RACE-high**	**Average**
$K=1, T=3$	93.58 ($\pm$ 1.47)	**89.33**($\pm$ 0.63)	81.78 ($\pm$ 1.11)	86.30 ($\pm$ 2.73)	84.35 ($\pm$ 2.24)	87.07 ($\pm$ 1.64)
$K=2, T=3$	93.59 ($\pm$ 1.24)	88.37 ($\pm$ **0.49**)	**82.09** ($\pm$ **0.81**)	86.30 ($\pm$ 2.64)	84.02 ($\pm$ 2.24)	86.87 ($\pm$ 1.48)
$K=4, T=3$	**93.83**($\pm$ 1.10)	89.07 ($\pm$ 0.53)	81.96 ($\pm$ 1.15)	**87.26** ($\pm$ **2.23**)	**85.17** ($\pm$ 1.71)	**87.46** ($\pm$ **1.34**)
$K=8, T=3$	**93.83** ($\pm$ **0.70**)	88.99 ($\pm$ 0.59)	82.69 ($\pm$ 0.97)	86.25 ($\pm$ 2.75)	84.36 ($\pm$ 2.06)	87.22 ($\pm$ 1.41)
$K=1, T=6$	93.37 ($\pm$ 1.47)	88.32 ($\pm$ 0.68)	81.05($\pm$ 3.44)	84.40 ($\pm$ 2.30)	83.34($\pm$ **1.66**)	86.10 ($\pm$ 1.91)

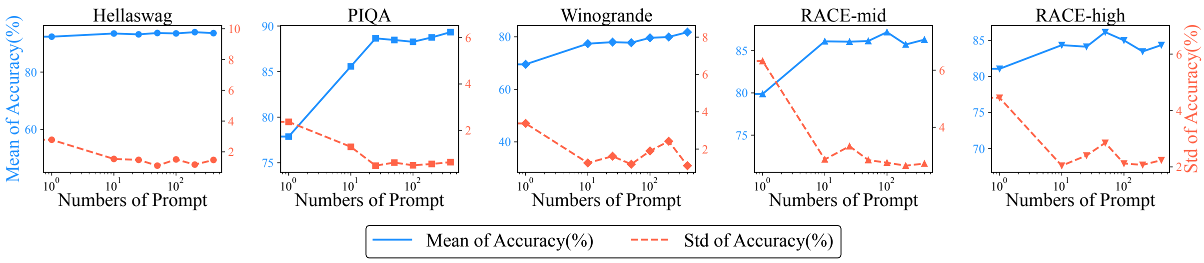

Figure 6: Scaling Law of Training Prompt Numbers: Mean and Standard Deviation of Accuracy Across Different Datasets. The x-axis represents the number of prompts on a logarithmic scale, while the y-axis shows the mean accuracy (left) and standard deviation of accuracy (right) for each dataset.

tems or time-sensitive information retrieval. Our enhanced inference efficiency translates to a better user experience and reduced computational resources required for deployment, making it a more practical and scalable solution.

5.3 Ablation Studies

Hyperparameter robustness This ablation study demonstrates the robustness of PAFT to the hyperparameters K (iterations per prompt) and T (epochs). As shown in Table 3, PAFT achieves stable performance across a broad range of K (1 to 8) and T (3 to 6) values, with minimal fluctuations in accuracy and variance. Notably, PAFT achieves near-optimal performance with default settings ($K=4, T=3$), attaining an average accuracy of 87.46%($\pm$1.34) across all tasks. This robustness reduces the need for extensive hyperparameter tuning, making PAFT a practical and efficient solution for real-world applications.

PAFT achieves strong performance with limited training prompts We conduct an ablation study to investigate the impact of varying numbers of training prompts on model performance, thus validating the effectiveness of PAFT. The experimental results, shown in Figure 5, demonstrate that as the number of prompts increases, the average accuracy of the model significantly improves, while the standard deviation decreases, indicating more stable and reliable performance. However, the performance gains diminish as the number of prompts increases, with only marginal improvements observed beyond a certain threshold. This suggests that while adding prompts can enhance performance, PAFT achieves competitive results with a minimal number of prompts, rendering excessive prompts unnecessary. In most cases, PAFT achieves strong performance with as few as 10 high-quality prompts, and further increases yield only marginal gains. The efficiency of PAFT is particularly notable, as it delivers excellent performance with a minimal number of prompts, making it highly suitable for resource-constrained scenarios where computational efficiency is critical. These findings underscore the practicality and efficiency of PAFT, offering a robust and efficient solution for real-world applications.

6 Conclusion

PAFT offers a compelling solution for enhancing the prompt robustness of LLMs. By dynamically adjusting prompts during fine-tuning, PAFT significantly improves model generalization and performance across diverse prompt formulations. Notably, PAFT boosts inference speed with maintained training cost. This approach paves the way for more reliable and efficient LLM deployment in real-world applications.

Limitations

In this section, we discuss potential limitations of PAFT and outline promising directions for future research. While PAFT demonstrates significant progress in enhancing the prompt robustness of Large Language Models (LLMs), certain aspects warrant further investigation. A key area for improvement lies in the dynamic prompt selection strategy employed during fine-tuning. Currently, PAFT utilizes a random sampling approach, which, while exposing the model to a diverse range of prompts, may not be the most efficient or effective method. Exploring more sophisticated sampling techniques, such as curriculum learning or importance sampling, could potentially optimize the training process and further enhance robustness. For instance, prioritizing prompts that induce higher loss or those that are more representative of the overall prompt distribution could lead to faster convergence and improved generalization. Furthermore, integrating adversarial learning into the dynamic fine-tuning phase presents a compelling avenue for future work. Generating adversarial prompts on-the-fly, perhaps through gradient-based updates, could further challenge the model and encourage it to learn more robust task representations. This approach could be particularly beneficial in mitigating the impact of maliciously crafted or unexpected prompts. However, the well-known instability of adversarial training remains a significant hurdle. Stabilizing the training process, perhaps through techniques like robust optimization or regularization, is crucial for realizing the full potential of this approach. Investigating different adversarial prompt generation strategies and their impact on model robustness would be a valuable contribution.

Ethics Statement

We have manually reevaluated the dataset we created to ensure it is free of any potential for discrimination, human rights violations, bias, exploitation, and any other ethical concerns.

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2019. Piqa: Reasoning about physical commonsense in natural language.

Kaiyan Chang, Songcheng Xu, Chenglong Wang, Yingfeng Luo, Xiaoqian Liu, Tong Xiao, and Jingbo Zhu. 2024. Efficient prompting methods for large language models: A survey.

Tianyi Chen, Tianyu Ding, Badal Yadav, Ilya Zharkov, and Luming Liang. 2023. Lorashear: Efficient large language model structured pruning and knowledge recovery.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Shizhe Diao, Zhichao Huang, Ruijia Xu, Xuechun Li, LIN Yong, Xiao Zhou, and Tong Zhang. 2023. Black-box prompt learning for pre-trained language models. *Transactions on Machine Learning Research*.

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.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey.

Jia He, Mukund Rungta, David Koleczek, Arshdeep Sekhon, Franklin X Wang, and Sadid Hasan. 2024. Does prompt formatting have any impact on llm performance?

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Wenyang Hu, Yao Shu, Zongmin Yu, Zhaoxuan Wu, Xiaoqiang Lin, Zhongxiang Dai, See-Kiong Ng, and Bryan Kian Hsiang Low. 2024. Localized zeroth-order prompt optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023a. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. 2023b. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Yoichi Ishibashi, Danushka Bollegala, Katsuhito Sudoh, and Satoshi Nakamura. 2023. Evaluating the robustness of discrete prompts. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2373–2384, Dubrovnik, Croatia. Association for Computational Linguistics.

Jens Kohl, Luisa Gloger, Rui Costa, Otto Kruse, Manuel P. Luitz, David Katz, Gonzalo Barbeito, Markus Schweier, Ryan French, Jonas Schroeder, Thomas Riedl, Raphael Perri, and Youssef Mostafa. 2024. Generative ai toolkit – a framework for increasing the quality of llm-based applications over their whole life cycle.

Po-Nien Kung and Nanyun Peng. 2023. Do models really learn to follow instructions? an empirical study of instruction tuning.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Yinheng Li. 2023. A practical survey on zero-shot prompt design for in-context learning. In *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, pages 641–647, Varna, Bulgaria. INCOMA Ltd., Shoumen, Bulgaria.

Xiaoqiang Lin, Zhaoxuan Wu, Zhongxiang Dai, Wenyang Hu, Yao Shu, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024. Use your INSTINCT: INSTruction optimization for LLMs using neural bandits coupled with transformers. In *Forty-first International Conference on Machine Learning*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Meta. 2024. Introducing meta llama 3: The most capable openly available LLM to date. *Meta Blog*.

Grégoire Mialon, Roberto Dessi, Maria Lomeli, Christoforos Nalmpantis, Ramakanth Pasunuru, Roberta Raileanu, Baptiste Roziere, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann

LeCun, and Thomas Scialom. 2023. Augmented language models: a survey. *Transactions on Machine Learning Research*. Survey Certification.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5316–5330, Dublin, Ireland. Association for Computational Linguistics.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions.

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. 2024. Gpt-4 technical report.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

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.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine

Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Mrigank Raman, Pratyush Maini, J Zico Kolter, Zachary Chase Lipton, and Danish Pruthi. 2023. Model-tuning via prompts makes NLP models adversarially robust. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale.

Abel Salinas and Fred Morstatter. 2024. The butterfly effect of altering prompts: How small changes and jailbreaks affect large language model performance.

Zeru Shi, Zhenting Wang, Yongye Su, Weidi Luo, Fan Yang, and Yongfeng Zhang. 2024. Robustness-aware automatic prompt optimization.

Yao Shu, Wenyang Hu, See-Kiong Ng, Bryan Kian Hsiang Low, and Fei Richard Yu. 2024. Ferret: Federated full-parameter tuning at scale for large language models. In *International Workshop on Federated Foundation Models in Conjunction with NeurIPS 2024*.

Chongjie Si, Zhiyi Shi, Shifan Zhang, Xiaokang Yang, Hanspeter Pfister, and Wei Shen. 2024. Unleashing the power of task-specific directions in parameter efficient fine-tuning.

Suvrit Sra, Sebastian Nowozin, and Stephen J Wright. 2011. *Optimization for machine learning*, page 351–368. Mit Press.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *Proceedings of ICML*.

Anton Voronov, Lena Wolf, and Max Ryabinin. 2024. Mind your format: Towards consistent evaluation of in-context learning improvements. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6287–6310, Bangkok, Thailand. Association for Computational Linguistics.

Chenxing Wei, Yao Shu, Ying Tiffany He, and Fei Richard Yu. 2024. Flexora: Flexible low-rank adaptation for large language models. In *NeurIPS 2024 Workshop on Fine-Tuning in Modern Machine Learning: Principles and Scalability*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. 2023. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence?

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2024. Instruction tuning for large language models: A survey.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2024. A survey of large language models.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Gong, and Xing Xie. 2024. Promptrobust: Towards evaluating the robustness of large language models on adversarial prompts. In *LAMPS@CCS*, pages 57–68.

A Experimental setting

In the main experiment, we compared PAFT with the baseline. The datasets and experimental parameters are as follows:

A.1 Dataset

In this section, we introduce the statistics of the dataset. The statistics of the dataset are shown in Table 4.

Table 4: Number of samples in the train, validation, and test datasets for various dateset.

Number of samples	train dataset	validation dataset	test dataset
Hellaswag	39900	10000	10000
PIQA	16000	2000	3000
Winogrande	40398	1267	1767
RACE	87866	4887	4934

A.2 Specific experimental parameters

Based on the LLaMA3-8B model configuration, several adjustments were made to optimize model performance. In the baseline model experiment, generation parameters were adjusted to ensure the correct output. In the LoRA experiment, adjustments to the generation parameters were retained, and LoRA-related parameters were adjusted. In the PAFT experiment, the size of the validation set was adjusted to control the time required to search for the optimal layer. For specific experimental parameters, see the table 5.

Table 5: Detailed experimental parameters. This table lists the specific parameters we used in the experiments for various methods. These parameters include the target module of LoRA (Lora Target), the maximum sequence length (Max Length), the number of samples for supervised fine-tuning (SFT Samples), the learning rate (LR), the number of training prompts (Training Prompts). Epoch(Epoch) represents the epoch of training. All other parameters not listed here remain consistent across all experiments.

Methods	LoRA Target	Max Length	SFT Samples	LR	Training Prompts	Epoch
LoRA	q & v Proj	1024	20000	0.0001	1	3
PAFT	q & v Proj	1024	20000	0.0001	400	3

B Training cost and inference time

PAFT Maintains Training Efficiency We now turn our attention to the training efficiency of PAFT. A critical consideration for any practical fine-tuning approach is its impact on training time. Introducing complex mechanisms or additional computational overhead can significantly hinder the training process, especially when dealing with large language models and extensive datasets. Therefore, it is essential to demonstrate that PAFT does not introduce such burdens.

To rigorously evaluate the training time implications of PAFT, we conducted a series of experiments, using Low-Rank Adaptation (LoRA) (Hu et al., 2022) as a representative example of a parameter-efficient fine-tuning method. LoRA has gained popularity due to its ability to adapt pre-trained models with minimal computational cost, making it a suitable baseline for our analysis. Our experiments, the results of which are presented in Table 3, directly compare the training time required for traditional LoRA fine-tuning with the training time required for PAFT integrated with LoRA.

The key finding from our analysis is that PAFT does not introduce any noticeable increase in training time. The data in Table 6 clearly demonstrates that the training duration remains virtually identical whether we employ standard LoRA or incorporate PAFT's dynamic prompt selection mechanism. This

Table 6: Training Time Comparison of Different Fine-tuning Methods on the Test Prompt Sets Across Various Reasoning and Reading Comprehension Tasks Using the LLaMA3-8B(Meta, 2024) Model with LoRA Rank 8. Experiments were conducted on an NVIDIA RTX 4090 GPU. Results are reported as training time in hours. **LoRA + TopAccuracy prompt** prompt refers to the prompt with the highest accuracy in the training set, **LoRA + user-specified prompt** (Wei et al., 2024) refers to fine-tuning with human-designed prompts, **LoRA + BATprompt** (Shi et al., 2024) uses the most robust prompt generated by BATprompt, and **LoRA + ZOPO prompt** (Hu et al., 2024) employs the optimal prompt selected by ZOPO from the training prompt set.

Training time/h	**Hellaswag**	**PIQA**	**Winogrande**	**RACE**	**Average**
LoRA + user-specified prompt	3.01	2.35	3.27	3.95	3.15
LoRA + TopAccuracy prompt	3.00	2.29	2.98	3.93	3.05
LoRA + BATprompt	3.02	**2.23**	3	3.93	3.05
LoRA + ZOPO prompt	**2.97**	2.3	**2.97**	3.83	**3.02**
PAFT	2.98	2.32	3.38	**3.81**	3.12

crucial observation underscores the efficiency of PAFT. The dynamic prompt selection process, which is central to PAFT's ability to enhance prompt robustness, is implemented in a way that does not add significant computational overhead. This is because the selection process is lightweight and seamlessly integrated into the existing training loop. Rather than requiring complex computations or extensive data manipulations, PAFT efficiently chooses from a diverse set of prompts, allowing the model to experience a wider range of input formulations without incurring a substantial time penalty. This efficient dynamic prompt selection is critical for the practical applicability of PAFT, ensuring that it can be readily deployed without compromising training efficiency. Furthermore, this efficiency allows for more extensive experimentation and exploration of different prompt variations, ultimately leading to more robust and generalizable models.

Efficient Candidate Prompt Generation A key aspect of PAFT's effectiveness lies in its ability to generate a diverse and high-quality set of candidate prompts efficiently. The process of constructing these candidate prompts involves leveraging the capabilities of external large language models (LLMs), which naturally raises the question of associated costs. Specifically, we sought to quantify the token usage required for candidate prompt generation, as this directly translates to the expense incurred when interacting with commercial LLM APIs.

To address this, we conducted a detailed analysis of the token consumption during the candidate prompt generation phase of PAFT. Our investigation, the results of which are summarized in Table 1, focuses on the number of tokens required to produce a sufficient variety of prompts suitable for subsequent selection and fine-tuning. We meticulously tracked the token usage across various prompts generated for different tasks, considering factors such as prompt length, complexity, and diversity.

The findings presented in Table 7 demonstrate that PAFT requires remarkably few tokens to generate a substantial pool of candidate prompts. This efficiency stems from PAFT's strategic approach to prompt engineering. Rather than relying on brute-force generation or computationally intensive search methods, PAFT employs a carefully designed prompting strategy that encourages the external LLMs to produce a wide range of prompt formulations with minimal token consumption. This is achieved through techniques such as few-shot prompting with carefully chosen examples, targeted instructions that guide the LLM towards desired prompt characteristics, and potentially iterative refinement of prompts based on preliminary evaluation. The low token count is crucial for practical applications, as it minimizes the cost associated with using commercial LLM APIs. Moreover, this efficiency enables the exploration of a broader range of potential prompts within a fixed budget, increasing the likelihood of discovering highly effective prompts that contribute to improved model robustness. This efficient prompt generation process is a significant advantage of PAFT, enabling it to achieve superior performance without incurring prohibitive costs.

Table 7: Token Usage for Candidate Prompt Generation. This table shows the number of tokens used to generate approximately 400 candidate prompts for each task. The average token usage is 11.75k. The number of generated prompts can be adjusted based on the scaling law observed in Figure 5 to control costs.

Tokens	Hellaswag	PIQA	Winogrande	RACE	Average
Total Tokens	11.7k	12.1k	10.9k	12.3k	11.75k

C Prompt

In this section, we present a selection of training and test prompts to illustrate the efficacy of our prompt construction algorithm and to provide a clearer understanding of operational process of PAFT. Due to space constraints, we only list 10 prompts as examples. Section C.1 showcases examples of training prompts, Section C.2 highlights test prompts, and Section C.3 outlines the prompts utilized by the baseline method.

C.1 Train prompt

In this section, we present the prompts generated using the method outlined in Section 4.1 across various datasets. All prompts listed here are utilized for training purposes.

Train Prompt of Hellaswag

```
1. Based on the given context {ctx}, which of the following options correctly predicts the outcome?
Choose the correct letter option.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
2. Considering the scenario described in {ctx}, identify the most accurate prediction of the
final result:Select the correct letter.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
3. Given the information in {ctx}, which option best forecasts the correct ending?Provide the
correct letter choice.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
4. From the context {ctx}, which of the following options accurately predicts the conclusion?Write
down the correct letter.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
5. Using the details provided in {ctx}, select the option that correctly predicts the final outcome:
Enter the correct letter.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
6. Based on the context {ctx}, which option is the most accurate prediction of the ending?Choose the
correct letter option.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
7. Given the scenario in {ctx}, identify the option that correctly forecasts the outcome:Select the
correct letter.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
8. Considering the details in {ctx}, which option best predicts the correct conclusion?Provide the
correct letter choice.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
9.Analyze the context {ctx} and determine the correct prediction of the outcome:Indicate the
correct letter.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
10. Analyze the given context {ctx} and determine the most accurate prediction of the final result:
Indicate the correct letter.\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
```

Train Prompt of PIQA

```
1.In order to {goal}, which of the following options is the most logical choice based on common
knowledge?\nA. {sol1}\nB. {sol2}\nAnswer:
2.Consider the scenario where you need to {goal}. Which option would be the most appropriate
according to general understanding?\nA. {sol1}\nB. {sol2}\nAnswer:
3.When trying to {goal}, which of the following would be the best course of action based on everyday
reasoning?\nA. {sol1}\nB. {sol2}\nAnswer:
4.To achieve {goal}, which option aligns best with common sense?\nA. {sol1}\nB. {sol2}\nAnswer:
5.Based on typical knowledge, which of the following is the correct choice to {goal}?
\nA. {sol1}\nB. {sol2}\nAnswer:
6.If you want to {goal}, which of these options would be the most sensible according to common
reasoning?\nA. {sol1}\nB. {sol2}\nAnswer:
7.Using general knowledge, determine the best option to {goal}.\nA. {sol1}\nB. {sol2}\nAnswer:
8.To {goal}, which of the following choices is the most reasonable based on common sense?
\nA. {sol1}\nB. {sol2}\nAnswer:
9.When considering how to {goal}, which option would be the most logical based on everyday knowledge?
\nA. {sol1}\nB. {sol2}\nAnswer:
10.According to common reasoning, which of the following is the best way to {goal}?
\nA. {sol1}\nB. {sol2}\nAnswer:
```

Train Prompt of Winogrande

```
1.Choose the correct answer to complete the sentence.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
2.elect the appropriate option to fill in the blank.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
3.Fill in the blank with the correct answer.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
4.Identify the correct choice to complete the statement.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
5.Choose the right answer to fill in the gap .{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
6.Select the correct option to complete the sentence.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
7.Fill in the blank with the correct answer.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
8.Identify the correct choice to complete the sentence.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
9.Choose the right answer to fill in the blank. {ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
10.Select the appropriate option to complete the statement.{ctx}
\nA. {only_option1}\nB. {only_option2}\nAnswer:
```

Train Prompt of RACE

```
1.Carefully read the following article and answer the question by selecting the correct option.
Respond with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
2.Read the passage below and choose the best answer to the question.
Reply with the letter A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
3.After reading the article, answer the following question by selecting the correct option.
Please respond with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
4.Examine the article provided and answer the question by choosing the most appropriate option.
Reply with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
5.Read the following text and answer the question by selecting the correct letter.
Respond with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
6.Carefully read the article and choose the best answer to the question.
Reply with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
7.Read the passage and answer the question by selecting the correct option.
Respond with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
8.After reading the article, choose the correct answer to the question.
Reply with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
9.Read the provided text and answer the question by selecting the best option.
Respond with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
10.Examine the article and answer the question by choosing the correct letter.
zReply with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
```

C.2 Test prompt

In this section, we present the prompts generated using the method outlined in Section 4.1 across various datasets. All prompts listed here are utilized for testing purposes, and they are not visible during training.

Test Prompt of Hellaswag

```
1.Based on the information provided, please select the most probable conclusion: {ctx}
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n
Remember to consider the implications of each option. Answer:
2.In the scenario described by {ctx}, there is only one correct way the story or situation could end.
When predicting the right ending, consider the cause-and-effect relationships established within
the context.An option that logically follows from the preceding events is likely the correct one.
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n You may choose from ’A’, ’B’, ’C’, ’D’.\n Answer:
3.Based on the given context {ctx}, which of the following options correctly predicts the outcome?
Choose the correct letter option.
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
4.To solve this problem based on {ctx}, weigh the significance of each potential ending:
A. {A}\nB. {B}\nC. {C}\nD. {D}\n You may choose from ’A’, ’B’, ’C’, ’D’.\n Answer:
5.Analyzing the context of {ctx}, think about the relationships and conflicts presented.
Which option is most likely to resolve these issues and lead to a satisfying ending?
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
6.{ctx}\nQuestion: Taking into account the context, which outcome is the most expected?
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
7.From the detailed description provided, choose the option that best completes the scenario:{ctx}\
n A. {A}\nB. {B}\nC. {C}\nD. {D}\n
Consider all aspects of the scenario to make an informed decision on the correct ending.\n Answer:
8.Given the scenario described in {ctx}, which of the following conclusions seems most plausible?
Consider all the details and clues provided to make an informed guess.
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
9.To unlock the hidden treasure in {ctx}, you need to choose the correct key.
Which option will open the treasure chest?
A. {A} B. {B} C. {C} D. {D}\n You may choose from ’A’, ’B’, ’C’, ’D’.\n Answer:
10.{ctx}\nQuestion: Reflecting on the emotional stakes and the structure of the narrative,
which conclusion feels the most genuine?
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n Answer:
```

Test Prompt of PIQA

```
1.Solve the following single-choice question by using your common sense reasoning skills.
Choose the correct option and reply with the corresponding letter.
\nQuestion: {goal}\nA. {sol1}\nB. {sol2}\nAnswer:
2.For the situation described by {goal}, consider which solution aligns more closely with how things
usually work in real life: A. {sol1}\nB. {sol2}. Use logical reasoning to guide your choice. Answer:
3.Given the context of the question, choose the answer that demonstrates the best common
sense reasoning: {goal}\nA. {sol1}\nB. {sol2}\n Answer format: A/B \nAnswer:
4.In considering the aim set forth in {goal}, visualize the potential consequences of each action
as if you were directly involved. This visualization can help you identify the better choice:\n
Question: {goal}\nA. {sol1}\nB. {sol2}\nAnswer:
5.Which solution fits the goal based on common sense?
{goal}\n A. {sol1}\nB. {sol2}\n Answer format: A/B \nAnswer:
6.Analyze the following scenario and select the answer that reflects logical reasoning: {goal}
\nA. {sol1}\nB. {sol2}\n Answer format: A/B \nAnswer:
7.Identify the most logical outcome for the situation described: {goal} A. {sol1} B. {sol2}
Answer format: A/B Remember, the trick is to apply your general knowledge to the scenario. Answer:
8.According to common reasoning, which of the following is the best way to {goal}?
\nA. {sol1}\nB. {sol2}\nAnswer:
9.Which solution best fits the goal based on your general knowledge? {goal}
\n A. {sol1}\nB. {sol2}\n Answer format: A/B \nAnswer:
10.You are about to answer a question that relies on your understanding of basic logic.
Please respond with A or B to indicate your choice.
\nQuestion: {goal}\nA. {sol1}\nB. {sol2}\nAnswer:
```

Test Prompt of Winogrande

```
1.In the context of {prompt}, which word best completes the sentence?
Choose: A. {only_option1}. B. {only_option2}.\nAnswer:.
2.When analyzing {prompt}, think about the overall theme. What fits best?
A. {only_option1}. B. {only_option2}.\nAnswer:.
3.For {prompt}, consider the emotional tone. Which option resonates more?
A. {only_option1}. B. {only_option2}.\nAnswer:.
4.Reflect on {prompt}. Which word logically fills the gap?
A. {only_option1}. B. {only_option2}.\nAnswer:.
5.In {prompt}, which choice aligns with the preceding ideas?
A. {only_option1}. B. {only_option2}.\nAnswer:.
6.When faced with {prompt}, think about the context. What completes it best?
A. {only_option1}. B. {only_option2}.\nAnswer:.
7.For {prompt}, identify the word that maintains the flow of the sentence.
Choose: A. {only_option1}. B. {only_option2}.\nAnswer:.
8.In the case of {prompt}, which option best conveys the intended meaning?
A. {only_option1}. B. {only_option2}.\nAnswer:.
9.Analyze {prompt} for clues. Which word fits the context?
A. {only_option1}. B. {only_option2}.\nAnswer:.
10.When considering {prompt}, which option enhances the clarity of the statement?
A. {only_option1}. B. {only_option2}.\nAnswer:.
```

Test Prompt of RACE

```
1.After reading the article, analyze the question and choose the best answer
based on the details and themes discussed. Look for clues within the text that
align with one of the options.\nArticle:\n{article}\n\nQuestion:
{question}\nOptions: \nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
2.Article:\n{article}\nAfter reading the passage, please answer the following question:
\n{question}\nA. {A}\nB. {B}\nC. {C}\nD. {D} \nAnswer:
3.Carefully read the following article and answer the question by selecting the correct option.
Respond with A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
4.Read the text carefully and answer the question by choosing the most appropriate option.
Evaluate the relevance of each choice to the main points discussed.
\nArticle:\n{article}\n\nQuestion: {question}\nOptions: \nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
5.Describe the setting of the article.
{question}\n{article}\nA. {A}\nB. {B}\nC. {C}\nD. {D} \nAnswer:
6.While reading the {article}, highlight or make mental notes of significant details.
The {question} is asking [describe the specific query].
Now evaluate the options:\nA. {A}\nB. {B}\nC. {C}\nD. {D} \nAnswer:
7.After carefully analyzing {article}, determine which of the following options best
answers the question:
{question}. A. {A}\nB. {B}\nC. {C}\nD. {D} \nAnswer:
8.Read {article} with a focus on answering {question}. Choose the most suitable option.
Article: {article} Question:{question} Options: A. {A} B. {B} C. {C} D. {D}
Trick: Be cautious of answer choices that seem too extreme. Your answer is just one letter. Answer:
9.Article:\n{article}\nFrom the information in the article, identify the correct
answer to the following question: \n{question}\nA. {A}\nB. {B}\nC. {C}\nD. {D} \nAnswer:
10.When {article} mentions {question}, which option best describes the author's attitude?
\nA. {A}\nB. {B}\nC. {C}\nD. {D} \n// Pay attention to the tone of the author.
Look for words that convey emotions or opinion to determine the attitude.Answer:
```

C.3 Baseline prompt

In this section, we present the best prompts generated or filtered using the baseline for training.

Test Prompt of Hellaswag

```
TopAccuracy prompt:
Given the context {ctx}, predict the correct ending by choosing the most logical option.
\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n You may choose from 'A', 'B', 'C', 'D'.\n Answer:

User-specified prompt:
{ctx}\n Question: {Question}\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n
You may choose from 'A', 'B', 'C', 'D'.\n Answer:

BATprompt :
Given the context below, predict the most logical ending by choosing the correct option
from the provided choices. Ensure your choice aligns with the context and is the most coherent
conclusion. \n Context: {ctx}\n
Question: Which ending makes the most sense?\n A. {A}\nB. {B}\nC. {C}\nD. {D}\n
You may choose from 'A', 'B', 'C', 'D'.\n Answer:

ZOPO prompt:
Based on {ctx}, which option is the most likely correct ending?
Consider the overall context, character motivations, and any foreshadowing.
Trick: Analyze the consistency of each option with the established details.
A. {A}\nB. {B}\nC. {C}\nD. {D}\n You may choose from 'A', 'B', 'C', 'D'.\n Answer:
```

Test Prompt of PIQA

```
TopAccuracy prompt:
Use both common sense and logical reasoning to determine the correct solution for the goal:
{goal}\n A. {sol1}\nB. {sol2}\n Answer format: A/B \nAnswer:

User-specified prompt:
There is a single choice question. Answer the question by replying A or B.'\n
Question: {goal}\nA. {sol1}\nB. {sol2}\nAnswer:

BATprompt :
You should use both common sense and logical reasoning to determine the most appropriate
solution for the following goal. Carefully evaluate the provided options and choose the
one that best aligns with the goal. Goal: {goal}\nA. {sol1}\nB. {sol2}\nAnswer:

ZOPO prompt:
To solve this common sense reasoning question, consider which of the two options seems
more plausible based on everyday knowledge and logic.
\nQuestion: {goal}\nA. {sol1}\nB. {sol2}\n
Think about the practical implications of each choice to determine the correct answer.\nAnswer:
```

Test Prompt of Winogrande

```
TopAccuracy prompt:
Question: {prompt}\nA. {only_option1}\nB. {only_option2}\nAnswer:

User-specified prompt:
There is a single choice question,  you need to choose the correct option to fill in the blank.
Answer the question by replying A or B.\n
Question:{prompt}\nA. {only_option1}\nB. {only_option2}\nAnswer:

BATprompt :
Complete the following sentence by selecting the most contextually appropriate option.
Carefully consider the meaning and context of the sentence to make your choice.
Question: {prompt}\nA. {only_option1}\nB. {only_option2}\nAnswer:

ZOPO prompt:
Question: Choose the correct modal verb: {prompt}\nA. {only_option1}\nB. {only_option2}\nAnswer:.
```

Test Prompt of RACE

```
TopAccuracy prompt:
Read the following article carefully: {article}. After reading, answer the question: {question}.
Choose the correct option from the choices provided:
\nA. {A}\nB. {B}\nC. {C}\nD. {D} \n
Trick: Focus on the main idea and supporting details in the article.
Output: Only the letter of the correct answer.\nAnswer:

User-specified prompt:
Article:\n{article}\nQuestion:\n{question}\nA. {A}\nB. {B}\nC. {C}\nD. {D} \nAnswer:

BATprompt :
Please read the passage carefully, focusing on the main ideas and supporting details.
Answer the question that follows by choosing the best option from the choices provided.
Ensure your response is based solely on the information in the passage. Output only the
letter of the correct answer. Article:\n{article}
\nQuestion:\n{question}\nA. {A}\nB. {B}\nC. {C}\nD. {D} \nAnswer:

ZOPO prompt:
A reading comprehension question is before you. Read the article and answer the question
by selecting A, B, C, or D.\n\nArticle:\n{article}\n\n
Q: {question}\n\nA. {A}\nB. {B}\nC. {C}\nD. {D}\nAnswer:
```