StablePrompt : Automatic Prompt Tuning using Reinforcement Learning for Large Language Model

Anonymous ACL submission

Abstract

Finding appropriate prompts for the specific task has become an important issue as the usage of Large Language Models (LLM) has expanded. Generating prompts that appropriate the characteristics of the target LLM and tasks is manual and time-consuming work. Reinforcement Learning (RL) is promising for prompt tuning due to its ability to incrementally produce better results through interaction with the environment. However, its inherent training instability and environmental dependency make it difficult to use in practice. In this paper, we propose StablePrompt, which strikes a balance between training stability and search space, mitigating the instability of RL and producing high-performance prompts. We formulate prompt tuning as RL problem between the agent and target LLM, and introduce Adaptive Proximal Policy Optimization (APPO), a modified version of PPO for prompt tuning. APPO incorporates an anchor model that is adaptively updated based on the training trajectory. Using the anchor model for the KL-divergence term in PPO keeps the search space flexible and ensures training stability. StablePrompt achieves State-of-The-Art performance on various tasks, including text classification, question answering, and text generation. Furthermore, our methods are robust to the type and size of the model.

1 Introduction

From Semantics (Bréal, 1900) to GPT-4 (Achiam et al., 2023), language models have undergone a huge evolution. Recently, large language models (LLM) have been applied beyond traditional natural language processing tasks to more general and diverse fields. While appropriate prompts enable LLMs to perform on par with or surpass human performance in many areas, finding these prompts is a manual and time-consuming process.

Various automatic prompt tuning methods are developed to address this problem, such as directly optimizing soft prompts (Bailey et al., 2023), leveraging the generative abilities of LLMs (Zhou et al., 2022; Pryzant et al., 2023; Wang et al., 2023), and editing manual prompts (Zhang et al., 2022b; Dong et al., 2023; Hou et al., 2023). However, they struggle to find prompts that satisfy different formats of inputs and outputs for each task. Finding prompts that make the target LLM respond in the correct format is also difficult due to slight differences in LLM type and size.

Reinforcement Learning (RL) is a promising method for addressing this problem as it allows the prompt to be updated using the responses of target LLM as rewards. However, its inherent training instability and environmental dependency raise practical challenges.

In this paper, we propose StablePrompt, a noble RL-based method that keeps training stability while ensuring search space flexibility. We define automatic prompt tuning as an RL problem between an agent and a target LLM, and introduce Adaptive Proximal Policy Optimization(APPO) as an optimizer designed for prompt tuning.

APPO combines the advantages of the original PPO (Schulman et al., 2017), which is well used in RL tasks, and the RLHF-style PPO (Ouyang et al., 2022), which is used for LLM training. Original PPO can explore a relatively large search space but suffer from learning instability. RLHF-style PPO leverages the language generation ability of LLM by using a fixed initial model but often finds suboptimal prompts due to the narrow search space.

To take advantage of both methods, APPO employs an anchor model. The anchor model, a snapshot of the agent model in the training trajectory, is updated only when a significant increase in performance is observed We use this anchor model to modify the KL-divergence term of the PPO.

Additionally, we extend StablePrompt for input-

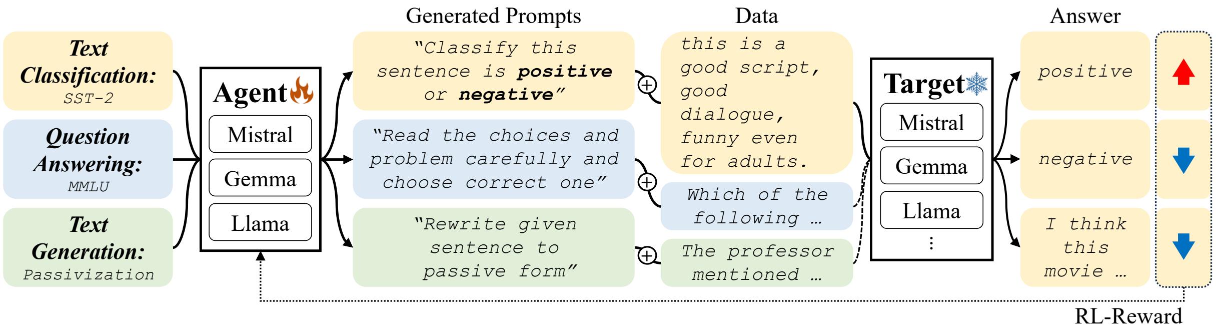

Text Classification: SST-2
Question Answering: MMLU
Text Generation: Passivization
Agent
Mistral
Gemma
Llama
Generated Prompts
"Classify this sentence is positive or negative"
"Read the choices and problem carefully and choose correct one"
"Rewrite given sentence to passive form"
Data
this is a good script, good dialogue, funny even for adults.
Which of the following …
The professor mentioned …
Target
Mistral
Gemma
Llama
Answer
positive
negative
I think this movie …
RL-Reward

Figure 1: Overview of StablePrompt. We construct an RL-framework using LLM as an agent and target, and train agent LLM to generate appropriate prompts for a given training dataset.

dependent prompt generating, which is named Test-Time Editing version StablePrompt (TTE-StablePrompt). TTE-StablePrompt is designed to handle tasks that are difficult to solve with a single prompt such as question answering. It can be easily extended by modifying the reward function while maintaining the framework of StablePrompt.

We validate StablePrompt on various tasks, including few-shot text classification, question answering, and text generation. To the best of our knowledge, our method is the first RL-based approach that works on agents LLM larger than 7B. We conduct experiments on diverse agents and target LLMs, including Mistral (Jiang et al., 2023), Gemma (Team et al., 2024), Llama (Touvron et al., 2023), and Falcon (Almazrouei et al., 2023). In our experiments, we show that StablePrompt generates appropriate prompts for a wide range of LLM sizes, types, and tasks. StablePrompt achieves State-of-The-Art (SoTA) performance across various tasks.

Our contributions are summarized as follows:

- We propose StablePrompt, which is RL-based prompt tuning method using APPO. APPO introduces an anchor model and modifies the KL-divergence term to keep training stable while ensuring the search space is flexible.
- StablePrompt achieves SoTA performance on various tasks, including text classification, question answering, and text generation. It also can be used with various types and sizes of agents and target LLMs.
- We extend StablePrompt to create an input-dependent prompt. It achieves high performance on tasks that are hard to solve with a single prompt.

2 Relatd Work

Automatic prompt tuning aims to find the appropriate prompts for a given dataset and target model. Soft prompt tuning (Bailey et al., 2023; Lester et al., 2021) uses direct gradient descent to search prompts. While it can find the optimal prompt, the generated prompt is often not readable and requires a substantial amount of data to converge.

By contrast, discrete prompt tuning aims to find prompts in the form of natural language. This approach often operates like black-box optimization, making it suitable for API-based LLMs. It can be broadly categorized into generation-based methods and RL-based methods.

Generation-based methods rely on the text generation abilities of LLMs to find prompts. Examples include APE (Zhou et al., 2022), which generates prompts by using example input-output pairs, ProTeGi (Pryzant et al., 2023), which improves prompts through iterative conversation, and PrompTAgent (Wang et al., 2023), which edits prompts based on a Monte Carlo tree search. Since these methods rely on the performance of a pre-trained LLM without additional tuning, they struggle with tasks that are outside the scope of pre-training.

RL-based methods use reinforcement learning to find prompts. Such as GrIPS (Prasad et al., 2022), BoostPrompt (Hou et al., 2023) and PACE (Dong et al., 2023), which edit the initial manual prompt using RL with an agent model. While these methods are relatively stable in training, they heavily depend on the quality of the manual prompt and the predefined action space for editing.

RLprompt (Deng et al., 2022) is a pioneering work that proposed a method for training agent LLMs using RL. RLprompt adds an MLP layer to the agent LLM for training stability and trains only on this layer. However, as the hidden size of the agent LLM increases, the size of the MLP layer increases, making it difficult to use. Also, these MLP layers are trained using only RL, which loses the initial model's language generation ability.

TEMPERA (Zhang et al., 2022b) is another pioneering study that used RL to explore input-dependent prompting. It adopts an agent model that shares a stem of the target LLM to generate input-dependent prompts. However, like GrIPS, it limited the action space to predefined prompts, suffering from a similar problem. It also struggles with scalability when the hidden size of the agent LLM becomes large. In this study, we extend the pioneering frameworks, to develop a stable and scalable RL-based method.

3 Method

3.1 RL Formulation

Problem Definition Our method follows RL-prompt (Deng et al., 2022) in terms of RL formulation. We formulate the discrete prompt tuning as the problem of finding the optimal discrete prompt $\mathbf{z}^{\star}$ for a given target model M_T and a dataset D. $\mathbf{z}$ is defined in the target model's vocabulary V^L and satisfy the following equation.

$$max_{\mathbf{z}\in V^L} R(M_T(\mathbf{z},x),y) \tag{1}$$

where R is pre-defined reward function, L is length of prompt, and $x, y \in D$.

We introduce an agent model M_a as LLM that generates prompts autoregressively from random input-output pair $(x_r, y_r) \in D$ and task-agnostic meta prompt. We define this set of inputs as state s. Detailed meta-prompt can be found in Figure 6. Agent model generates prompts up to the length l according to the $M_a(z_l|s,\mathbf{z}_{<l})$. After $\mathbf{z}$ is created, it receives a reward from the $R(M_T(\mathbf{z},x),y)$. The full training objective function is below:

$$max_{M_a} R(M_T(\mathbf{z},x),y), \mathbf{z} \sim \Pi_{l=1}^{L} M_a(z_l|s,\mathbf{z}_{<l}) \tag{2}$$

Original PPO As a method for training LLM agents with RL, we adopt Proximal Policy Optimization (PPO). We add a value head to the last layer of the LLM agent, which is trained using MSE loss to predict reward values for inputs.

$$L_v = (v_{preds} - reward)^2 \tag{3}$$

The value expected from value head is used with reward to compute advantage A, which uses Generalized Advantage Estimation (GAE) and clipped.

$$A = GAE(v_{preds}, reward) \tag{4}$$

$$ratio = \frac{\theta_t(\mathbf{z}|s)}{\theta_{t-1}(\mathbf{z}|s)} \tag{5}$$

$$A_{clipped} = clip(ratio, 1-\epsilon, 1+\epsilon) * A \tag{6}$$

where θ is parameter of agent model and t is timestep.

Then calculate the penalty P which is the KL-divergence between the previous version of the agent model and the current version. The full agent loss is the following :

$$P = KL(\theta_t(\mathbf{z}|s)||\theta_{t-1}(\mathbf{z}|s)) \tag{7}$$

$$L_{agent} = A_{clipped} + P \tag{8}$$

The final PPO objective is the following :

$$L_{PPO} = L_v + L_{agent} \tag{9}$$

In practice, we perform parameter-efficient training using LoRA (Hu et al., 2021) and update only value head and LoRA adaptor.

3.2 StablePrompt

Anchor Model We introduce an anchor model, which is a copy of the agent model with validated performance improvements in the training trajectory. The anchor model starts as a copy of the initial agent and is carefully updated at a predefined update period u_t. If the performance of the current agent model is higher than an update threshold compared to the anchor model, the anchor model is updated to the copy of the current agent model. Conversely, if the agent model underperforms the anchor model by less than a rollback threshold, the agent model is rolled back to the anchor model.

This allows the anchor model to adaptively update based on the characteristics of the task. If the reward signal is stable or requires many updates to find the optimal prompt, the anchor model is updated accordingly. On the other hand, if the reward signal is unstable or does not require many updates to find the optimal prompt, the anchor model is updated in a few steps or not. The suggested anchor model reduces the environmental dependence of RL by ensuring a performance-validated model for the KL-divergence penalty.

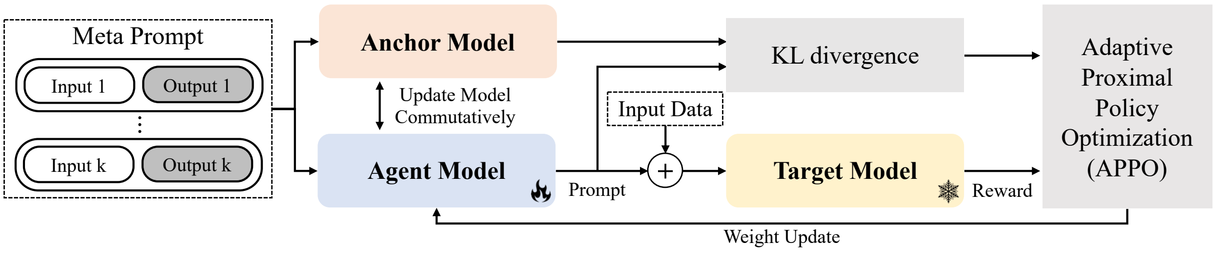


Figure 2: Training framework of StablePrompt. Generate prompts using the Task agnostic meta-prompt, and calculate the reward of the generated prompts with training data.

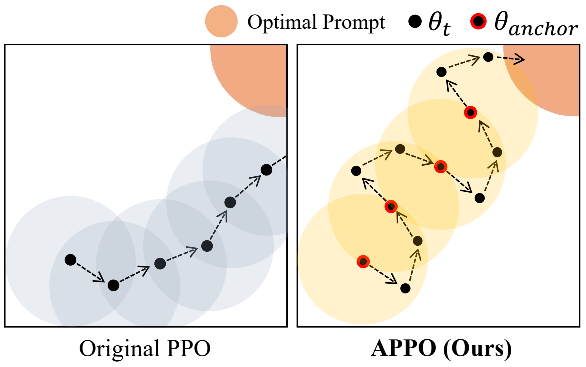


Figure 3: Illustration comparing APPO to the original PPO. The circle represents the constraint of KL-divergence, and each dot represents the parameter of the agent model at each time step. APPO is robust to incorrect rewards because it maintains an anchor model, while PPO deviates from the optimal prompt as incorrect rewards accumulate.

Adaptive PPO The KL-divergence penalty term (7) uses the parameters of the previous model to prevent the current model from changing too much. But as the steps get longer, the model can gradually diverge from the initial. When unstable reward signals accumulate, this can lead the model into a local minima.

In RLHF-style PPO (Ouyang et al., 2022), the penalty term (7) is modified by $KL(\theta_t(\mathbf{z}|s)||\theta_0(\mathbf{z}|s))$ to prevent the agent model from deviating too far from the initial version. This is appropriate for a task like RLHF that needs to answer a wide variety of questions while not losing the initial language generation capability. However, in prompt tuning, RLHF-style PPO limits the agent's search space to the initial agent, which leads to suboptimal prompts.

We propose Adaptive PPO (APPO), which combines the advantages of RLHF-style and original PPO, achieving both training stability and an extensive search space. We leverage the anchor model to modify Equation (7) as follows :

$$P_{APPO} = KL(\theta_t(\mathbf{z}|s)||\theta_{anchor}(\mathbf{z}|s)) \quad (10)$$

This term restricts the agent model from diverging too far from an anchor model. This approach allows for more conservative agent updates compared to the original PPO while ensuring a larger search space compared to RLHF-style PPO. The full objective of APPO is below:

$$L_{agent}^{APPO} = A_{clipped} + P_{APPO} \quad (11)$$

$$L_{APPO} = L_v + L_{agent}^{APPO} \quad (12)$$

Reward Function We design reward functions for RL. For text classification, we use accuracy and softmax difference. While accuracy is a good reward function, it has discrete values, which can lead to many prompts having the same accuracy. This problem is often encountered in scenarios with limited training data, such as few-shot text classification. To mitigate this, we introduce the softmax difference, which subtracts the highest value among the incorrect options from the value of the correct answer from the softmax output.

$$R(\mathbf{z}, x, y) = c_a Acc(\mathbf{z}, x, y) + c_s D(\mathbf{z}, x, y) \quad (13)$$

$$D = M_T(\mathbf{z}, x)_{i=y} - \max[M_T(\mathbf{z}, x)_{i \neq y}] \quad (14)$$

This metric is used to rank prompts when they have the same accuracy. The softmax difference is also widely used in other RL frameworks for classification (Han et al., 2023).

For text generation, we use the F1 score directly as the reward function.

3.3 Test Time Editing StablePrompt

For tasks that are difficult to solve with a single prompt, we expand StablePrompt to generate prompts that depend on the input query. We call this extended version as Test-Time Editing StablePrompts (TTE-StablePrompt).

In TTE-StablePrompt, input state s is defined by the set of meta-prompt, example input-output pairs and current input. The agent generates a prompt for the current input. The generated prompt and current input are fed into the target model to calculate rewards. Detailed meta prompt can be found in Figure 6. We keep the same settings for the other parts of the method.

This approach is different from StablePrompt, which uses the average value of the training batch as its reward. The reward of TTE-StablePrompt is calculated using only the current input. This instance reward signals in TTE-StablePrompt train the agent model to generate prompts that are effective for specific inputs, rather than prompts that are effective across the entire dataset.

4 Experiment

4.1 Few Shot Text Classification

Datasets Few-shot text classification is a task that has been used in many previous prompt tuning studies, including (Deng et al., 2022; Zhang et al., 2022b). We use datasets that are subsets of GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019), including sentiment analysis datasets (SST-2) and natural language inference datasets (MRPC, MNLI, QNLI, SNLI, and RTE).

For inference, similar to previous studies, we used a verbalizer with predefined class label tokens. When determining the model's predictions, we selected candidates only from the set of verbalizers. Detailed dataset statistics and verbalizer settings can be found in the Table 7.

Baselines As a baseline, we compare the various methods. Our baselines include supervised fine-tuning methods such as LoRA fine-tuning and soft prompt tuning (Bailey et al., 2023). We also use fixed prompts including hand-crafted manual prompts, few-shot prompts, and zero-shot chain of thought (Zero-Shot CoT) (Wei et al., 2022).

For direct comparison with StablePrompt, we use various discrete prompt tuning methods. These include generation-based methods such as APE (Zhou et al., 2022) and ProTeGi (Pryzant et al., 2023), and RL-based method such as GrIPS (Prasad et al., 2022), PromptBoosting (Hou et al., 2023) and RLprompt (Deng et al., 2022), which is directly comparable to ours. In the case of TEMPERA (Zhang et al., 2022b), the agent model grows proportionally to the size of the target model, making it difficult to experiment with 7B models. Therefore, we conducted experiments on the 330M RoBERTa-large (Liu et al., 2019) model and included the results in the Table 8.

Implementation Detail We perform two experiments on Few-Shot Text Classification. One is an experiment with both the target and the agent model fixed to gemma-1.1-7B-it (Gemma-7B) (Team et al., 2024) for comparison with the baselines. For RLprompt, due to its MLP layer size overhead, we use GPT2-XL (Radford et al., 2019) as the agent.

The other experiment runs StablePrompt on five target models: gemma-1.1-2B-it (Gemma-2B), Gemma-7B, Mistral-7B-it-v2.0 (Mistral-7B) (Jiang et al., 2023), llama3-8B-it (Llama3-8B) (Touvron et al., 2023), and falcon-11B (Falcon-11B) (Almazrouei et al., 2023), and four agent models: Gemma-2B, Gemma-7B, Mistral-7B, Llama3-8B. We report the average accuracy of 6 datasets.

All experiments were performed with three distinct random seeds. For the generated prompts, we used the template *"[prompt] Input : [input] Output :"* for prediction. We use only 16 samples per label for training. The generated prompts of each step are queued in pairs with rewards. At the test time, the top 5 prompts in order of reward are selected for testing and report the highest performance. This is the same method as RLprompt. Detailed numbers are shown in the Table 6.

Result Table 1 shows the performance of various baselines and StablePrompt. StablePrompt achieves State-of-The-Art (SoTA) performance on all tasks except QNLI. In QNLI, StablePrompt also achieves the best performance among the discrete prompt tuning methods. The average score also outperforms APE and achieves SoTA. We present the full version of the generated prompt in the Appendix C.1.

Figure 4 illustrates the performance of StablePrompt across various Agent-Target pairs. The values in the heatmap are the averages of six datasets. StablePrompt outperforms manual prompts across all pairs except (Gemma-2B, Llama3-8B) pair. These results demonstrate that our method is robust to model sizes, such as a small agent model of 2B and a large target model of 11B.

Specifically, when comparing Mistral-7B and Falcon-11B, the manual prompt performance is higher with falcon-11B, but with appropriate prompting from StablePrompt, Mistral-7B outperforms falcon-11B. This shows that an appropriate prompt can enable a small model to easily understand a given task and achieve performance comparable to that of a large model.

	Method	SST-2	MRPC	RTE	QNLI	MNLI	SNLI	Average
Fine-Tuning	Fine-Tuning	71.9	59.6	55.7	63.1	41.1	64.8	59.3
	Soft prompt tuning	78.3	57.1	51.6	**89.0**	34.9	55.8	61.1
Fixed prompt	Manual prompt	89.1	51.0	64.0	73.0	67.0	47.0	65.2
	Zero-shot CoT	57.9	38.4	81.6	75.2	71.1	66.3	65.1
	Few-shot prompt	55.0	49.0	76.0	82.0	58.0	52.2	62.0
Discret prompt tuning	GrIPS	84.7(±4.6)	55.6(±2.6)	60.9(±3.5)	28.9(±1.2)	44.4(±1.1)	63.5(±2.3)	59.4
	PromptBoosting	65.4(±1.0)	52.7(±1.1)	71.6(±0.9)	71.6(±1.1)	35.5(±1.4)	52.6(±1.8)	58.2
	APE	83.2(±7.7)	55.3(±4.9)	78.6(±1.3)	75.0(±2.2)	54.6(±7.9)	72.3 (±4.8)	70.1
	ProTeGi	69.2(±8.4)	48.8(±1.3)	73.2(±6.3)	74.2(±7.7)	56.6(±10.9)	61.3(±12.3)	64.0
	RLprompt	70.8(±6.5)	56.0 (±1.5)	67.3(±2.5)	62.6(±1.3)	54.6(±1.9)	56.6(±1.3)	61.3
	StablePrompt (Ours)	**92.5**(±1.3)	**71.3**(±3.4)	**81.5**(±2.8)	75.9 (±1.4)	**63.3**(±1.2)	**74.1**(±1.4)	**76.4**

Table 1: Result of 6 few-shot text classification datasets. Generated prompt can found in Appendix C.1

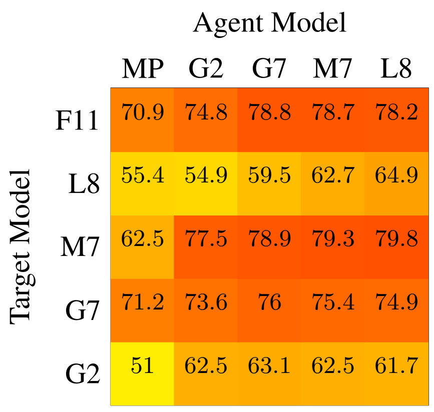


Figure 4: Heatmap of few-shot text classification tasks on diverse target-agent pairs. Reported numbers are an average of 6 datasets. *MP : Manual prompt, G2: Gemma-2B, G7: Gemma-7B, M7: Mistral-7B, L8: Llama-3-8B, F11: Falcon-11B.*

	BBII		II
	Text Classification	Text Generation	Instruction Induction
Manual Prompt	51.57	37.61	33.70
PromptAgent	28.50	-	-
APE	56.46	49.59	51.94
ProTeGi	56.58	55.61	51.60
StablePrompt (Ours)	**57.75**	**61.36**	**65.80**

Table 2: Result of BigBench-Hard Instruction Induction (BBII) and Instruction Induction (II) datasets. For BBII, we divided it into two parts based on the type of task. Full results can be found in Table 10 and Table 11.

4.2 Induction Task

Datasets We experimented with an induction task in which the agent has to provide a rule for an input-output pair as a prompt. We used the Instruction Induction dataset (II) (Mishra et al., 2022) and BigBench-Instruction Induction dataset (BBII) (Zhou et al., 2022), a subset of BiG-Bench (Ghazal et al., 2013). These include tasks such as editing the input sentence or finding answers according to rules. Each task requires prompts in the form of instructions designed to help the target model induce the correct answer.

The tasks consist of text classification and text generation, requiring an understanding of various fields such as spelling, morphosyntax, and phonetics. We conducted experiments on BBII, which has 20 subsets, and Instruction Induction, which has 23 subsets. The dataset details can be found in Appendix A.3.

Implementation Detail We performed experiments with two different target models. One used the Gemma-7B and the other performed experiments on InstructGPT3.5. For the first experiments, due to the large number of datasets, we used APE and ProTeGi as baselines, and we included PromptAgent (Wang et al., 2023) which is a Monte Carlo tree search-based generation method designed for BigBench text classification tasks.

In experiments using InstructGPT3.5, due to cost issues, we used APE, which requires fewer steps. For APE, we used various agent models such as (Zhang et al., 2022a; Zeng et al., 2022). We use PACE (Dong et al., 2023), an RL-based editing method designed for induction tasks, and a human prompt from the same paper as the baseline.

For text classification, we used the same reward function as Section 4.1. For text generation, we use f1 score as reward function. We use the same template as Section 4.1 for both BBII and II.

Results Experiments on Gemma-7B target model are presented in Table 2. Our method achieves SoTA on both BBII and II. In particular, it outper-

Method	Human prompt	Human prompt + PACE	APE			StablePrompt
Agetn Model	-	-	GLM	OPT	InstructGPT3.5	Mistral
Parameters	-	-	130B	175B	unknown	7B
Antonyms	85.0	**87.0**	78.0(±0.5)	82.7(±0.7)	81.0(±0.7)	83.7(±0.9)
Cause selection	84.0	85.0	53.3(±0.1)	65.3(±1.0)	72.0(±1.0)	**88.7**(±1.0)
Passivization	100.0	100.0	7.3(±0.0)	100.0(±0.0)	100.0(±0.0)	100.0(±0.0)
Second Letter	99.0	100.0	3.3(±0.9)	100.0(±0.0)	100.0(±0.0)	100.0(±0.0)
Sentiment	91.0	**92.0**	87.7(±0.8)	82.7(±0.9)	88.3(±0.8)	90.7(±0.9)
Translation en-fr	89.0	88.0	79.7(±0.8)	85.3(±0.8)	84.3(±0.8)	**90.3**(±1.0)
Average on 6 tasks	91.3	92.0	51.8	68.6	89.3	**92.8**
Average on 24 tasks	79.8	80.3	-	-	77.5	**81.5**

Table 3: Result of 6 selected tasks and an average of all 24 tasks in the Instruction induction dataset with InstructGPT3.5 as the target model. Full results can be found in Table 12.

	STEM	Social Sciences	Humanities	Other	Average
manual prompt + fewshot	47.1	61.6	55.4	54.5	53.9
Zero-Shot CoT	49.2	59.6	54.5	56.0	54.2
APE	45.0	59.3	56.4	51.1	52.1
ProTeGi	45.7	59.7	56.0	55.3	53.3
RLprompt	46.5	55.1	56.6	55.7	52.8
StablePrompt (Ours)	47.8	63.6	58.6	59.0	56.3
TTE-StablePrompt (Ours)	**49.6**	**65.7**	**59.6**	**58.8**	**57.5**

Table 4: Result of MMLU QA datasets with Gemma-7B as target model. Full results can be found in Table 13

forms on the text generation tasks II and BBII. This shows the effectiveness of the RL framework on the text generation tasks, where the format of the output is important.

Table 3 shows the experiments conducted using InstructGPT3.5. StablePrompt shows strong performance even with the large black-box model InstructGPT3.5 as a target model. This highlights the benefits of RL-based method, which works well when the target model is not publicly accessible.

Note that our method outperforms APE, which uses models larger than 100B as the agent. In particular, the 7B model trained by StablePrompt produces better prompts than the commercial black box model InstructGPT3.5. This shows that our method does not rely on the ability of the agent model and is cost-efficient by using a small model.

4.3 Question Answering

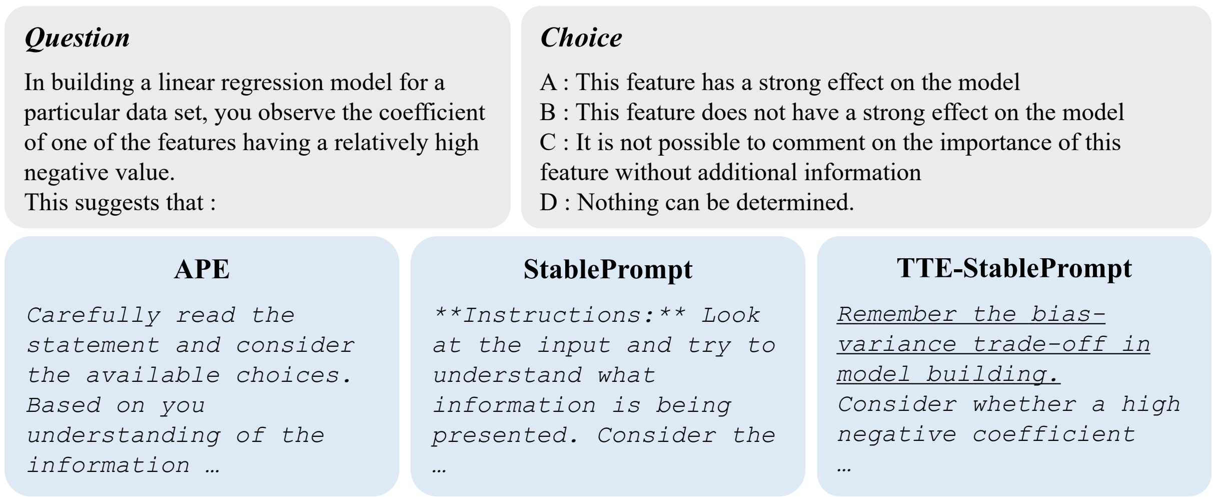
Datasets We conducted an experiment on a Question Answering (QA) task. In this paper, we use the MMLU (Hendrycks et al., 2020) dataset, which requires users to answer questions from various fields. We report the performance of 57 question topics from MMLU, categorized into STEM, Humanity, Social Science, and Others. The verbalizer is used in the same way as for text classification. We present 4 options (A,B,C,D) in a question and use the alphabet corresponding to each option as a verbalizer. The reward function is the same as Section 4.1. Detailed numbers of datasets can be found in Appendix A.3.

Implementation Detail The target and agent model are both fixed with Gemma-7B. For the prompt, we used the template *"[Prompt] Question : [Question] Choice : [Choice] Output :"*. We trained the model using 20 question-answer pairs from the validation dataset for each topic.

Result Table 4 shows the performance of various baselines. StablePrompt achieves the highest performance among the baselines. In particular, StablePrompt outperformed in all fields except STEM.

Note that TTE-StablePrompt outperforms StablePrompt. There are many different questions on the same topic that are difficult to solve with a single prompt. TTE-StablePrompt, which gives different instructions depending on the input within the same subject, is more effective than StablePrompt, which only uses a single prompt. TTE-StablePrompt also performs better than Zero-Shot CoT, which uses the same multi-step reasoning and is known to perform well on maths/science tasks.

Figure 5 shows question-choice pairs from the machine learning dataset in MMLU, along with the prompts generated by APE, StablePrompt, and TTE-StablePrompt. APE and StablePrompt generate almost semantically similar prompts which can be generally used for all questions in subject. However, TTE-StablePrompt generates prompts appropriate to the given question (emphasized with



Question

In building a linear regression model for a particular data set, you observe the coefficient of one of the features having a relatively high negative value.
This suggests that :

Choice

A : This feature has a strong effect on the model
B : This feature does not have a strong effect on the model
C : It is not possible to comment on the importance of this feature without additional information
D : Nothing can be determined.

APE

Carefully read the statement and consider the available choices. Based on you understanding of the information …

StablePrompt

***Instructions:** Look at the input and try to understand what information is being presented. Consider the …*

TTE-StablePrompt

Remember the bias-variance trade-off in model building. Consider whether a high negative coefficient …

Figure 5: Generated prompts and input in machine learning subset of MMLU dataset. We truncated the latter part of the generated prompt for readability. Full prompt can find in Appendix C.5

	Original PPO	RLHF-style PPO	APPO (Ours)
SST2	91.5(±0.7)	91.1(±1.0)	**92.5**(±1.3)
MRPC	65.9(±9.0)	70.6(±2.2)	**71.3**(±3.4)
RTE	80.2(±2.1)	80.3(±2.0)	**81.5**(±2.8)
QNLI	70.2(±2.1)	**76.7**(±1.6)	75.9(±1.4)
MNLI	**66.2**(±2.5)	61.0(±1.2)	63.3(±1.2)
SNLI	69.5(±1.9)	70.4(±3.3)	**74.1**(±1.4)
Average	73.3	74.2	**76.4**

Table 5: Result of ablation study on few-shot text classification tasks. We report the average and standard deviation of experiments from 5 distinct random seeds.

underlining). This shows that a simple TTE extension effectively creates an input-dependent prompt.

4.4 Ablation Study

Experiment Settings We conduct an ablation study for APPO. We use same settings as few-shot text classification. We fix the agent and the target model to Gemma-7B.

Results Table 5 shows the performance of PPO variants. APPO performs well on average across all tasks by leveraging the strengths of both the original PPO and RLHF-style PPO through adaptive anchor model updates. APPO can either behave like the RLHF-style PPO, with no updates, or like the original PPO, with updates in each every update period. In particular, when the performance gap between the original PPO and RLHF-style PPO is significant, APPO adapts to the better performing model. This pattern is observed in tasks like MRPC, QNLI, and MNLI. Additionally, in tasks such as SNLI, APPO can identify more appropriate prompts than either the original or RLHF-style PPO alone.

5 Conclusion

In this paper, we propose a novel RL-based prompt tuning method, StablePrompt. We define prompt tuning as an RL problem and introduce APPO, which is an improved version of PPO. APPO has the advantages of both RLHF-style and the original PPO. It introduces an anchor model that is updated adaptively to reduce task dependency and increase training stability. We demonstrate its performance through experiments on a variety of tasks, model sizes, and model types. To the best of our knowledge, this is the first RL-based tuning approach that works effectively on agent models larger than 7B.

6 Limitation

The limitations of this study can be summarized as follows: (1) This paper does not cover experiments that are significantly beyond the scope of prior learning, such as medical and legal domains; however, since it is a training-based method, it is expected to be scalable in future work. (2) This paper can be used to abuse LLM for specific purposes. This is a particular threat to commercial LLMs in the API format because they are based on black-box optimization.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic, et al. 2023. Falcon-40b: an open large language model with state-of-the-art performance. *Findings of the Association for Computational Linguistics: ACL*, 2023:10755–10773.

Luke Bailey, Gustaf Ahdritz, Anat Kleiman, Siddharth Swaroop, Finale Doshi-Velez, and Weiwei Pan. 2023. Soft prompting might be a bug, not a feature.

Michel Bréal. 1900. *Semantics: Studies in the science of meaning*. W. Heinemann.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint arXiv:2205.12548*.

Yihong Dong, Kangcheng Luo, Xue Jiang, Zhi Jin, and Ge Li. 2023. Pace: Improving prompt with actor-critic editing for large language model. *arXiv preprint arXiv:2308.10088*.

Ahmad Ghazal, Tilmann Rabl, Minqing Hu, Francois Raab, Meikel Poess, Alain Crolotte, and Hans-Arno Jacobsen. 2013. Bigbench: Towards an industry standard benchmark for big data analytics. In *Proceedings of the 2013 ACM SIGMOD international conference on Management of data*, pages 1197–1208.

Gyojin Han, Jaehyun Choi, Haeil Lee, and Junmo Kim. 2023. Reinforcement learning-based black-box model inversion attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20504–20513.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Or Honovich, Uri Shaham, Samuel R Bowman, and Omer Levy. 2022. Instruction induction: From few examples to natural language task descriptions. *arXiv preprint arXiv:2205.10782*.

Bairu Hou, Joe O'connor, Jacob Andreas, Shiyu Chang, and Yang Zhang. 2023. Promptboosting: Black-box text classification with ten forward passes. In *International Conference on Machine Learning*, pages 13309–13324. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2022. Grips: Gradient-free, edit-based instruction search for prompting large language models. *arXiv 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. *arXiv preprint arXiv:2305.03495*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P Xing, and Zhiting Hu. 2023. Promptagent: Strategic planning with language models enables expert-level prompt optimization. *arXiv preprint arXiv:2310.16427*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022a. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Tianjun Zhang, Xuezhi Wang, Denny Zhou, Dale Schuurmans, and Joseph E Gonzalez. 2022b. Tempera: Test-time prompting via reinforcement learning. *arXiv preprint arXiv:2211.11890*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

A Experiment Details

A.1 RL parameters

We summarize the RL-related hyperparameters used in our experiments in the Table 6. We used the same hyperparameters for all tasks.

HyperParameters	Stableprompt
Leraning Rate	1.00E-05
Value loss Coefficient	0.1
Gamma	1
GAE Lambda	0.95
cliprange	0.2
u_t	5
Update Threshold(%)	0.05
Rollback Threshold(%)	0.1
Prompt per Batch	4
Maximum Prompt Length	150
c_a	10
c_s	0.1

Table 6: Detail parameters used in StablePrompt.

A.2 Meta Prompt

We show the meta-prompt used as input to StablePrompt in Figure 6.

A.3 Dataset Details

Few-shot Text Classification Detailed number and verbalizer settings can be found in Table 7.

Dataset	Type	|C|	|Train|=|Dev|	|Test|	Verbalizer
SST2	sentiment	2	32	1.8k	[yes,no]
MRPC	NLI	2	32	1.7k	[yes,no]
RTE	NLI	2	32	0.3k	[yes,no]
QNLI	NLI	2	32	9.8k	[yes,no]
MNLI	NLI	3	48	10k	[yes,maybe,no]
SNLI	NLI	3	48	9.8k	[yes,maybe,no]
MMLU	QA	4	-	-	[A,B,C,D]

Table 7: Details of the datasets for few-shot classification.

Induction Task BIG-Bench Instruction Induction (BBII) is a subset of 21 tasks with clear and human-written instructions that can be applied to all examples in the dataset (Zhou et al., 2022). The detailed type and metric for each dataset can be found in Table 10.

Instruction Induction is conducted with 24 induction tasks proposed in (Honovich et al., 2022). The tasks span many features of language understanding, from simple phrase structure to similarity and causality identification. The detailed metric for each dataset can be found in Table 11.

Question Answering The MMLU QA dataset consists of 15,908 questions. The dataset is divided into subsets according to 57 subjects. We used the validation set of all subsets as the training set. The total number of validation sets is 1,540. Each subset has a minimum of 100 test samples, with a total of 14,079 test questions.

Meta prompt Template

I gave a friend an instruction and five inputs.
The friend read the instruction and wrote an output for every one of the inputs.
Here are the input-output pairs:

Input : $[Q_1]$	Output : $[A_1]$
Input : $[Q_2]$	Output : $[A_2]$

…..

The instruction was :

Meta prompt Template - TTE

Look at the instruction and current input,
rewrite instruction for current input.
Here are the input-output pairs:

Input : $[Q_1]$	Output : $[A_1]$
Input : $[Q_2]$	Output : $[A_2]$

…..

Current Input Here : $[Q_t]$
The Rewritten instruction was :

Figure 6: Detail template of meta prompt used in StablePrompt and TTE-StablePrompt

A.4 Baseline Details

APE For a fair comparison, we scale the number of prompts generated by APE to be the same as the number that StablePrompt generates during training. Also, unlike the original APE, we use the entire validation set to determine the final prompt. This setting is more favorable than the original APE and improves performance.

ProTeGi We used additional settings same as APE and limited the number of consecutive conversations to two.

RLprompt For RLprompt, as the hidden size of the agent model increases, the size of the MLP layer increases as well, making it difficult to train the model. Therefore, we used GPT2-XL (Radford et al., 2019) 1.5B, which is the largest model in the official implementation.

PromptAgent We utilized the official repository and only used it for the text classification problem as no evaluation metric was specified for text generation. PromptAgent is known to work well on high-performance LLMs such as GPT-4. However, in our experiments, we found that using small 7B-level models as agents significantly degrades performance.

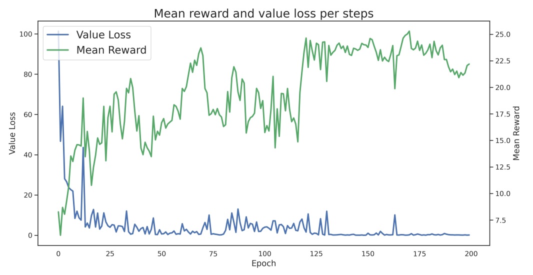


Figure 7: Training curve of mean reward and value loss by steps.

A.5 Training Details

We experiment on a single A100 GPU. For text classification, we use 100 epochs and need 2-3 GPU hours per task. For question answering and induction tasks, we use 30 epochs and need 1-2 GPU hours per task. Training time can be changed by the average length of inputs.

B Additional Experiments

B.1 Text Classification in Small Target Model

Implementation Details To compare the performance of our methods with traditional prompt tuning baselines, we perform text classification again on a relatively smaller target model. The target model is fixed as RoBERTa-Large (330M). We replace the MRPC dataset with the MR dataset from glue. Note that the MR dataset is a sentiment classification task, not NLI. For RLprompt, the agent model is GPT-2, as specified in the original paper. For StablePrompt, the agent model used is Mistral-7B.

Results The experimental results are shown in Table 8. StablePrompt demonstrates the highest performance across all datasets except MR. Even in MR, it shows comparable performance to TEMPERA, which uses Test-Time Editing for multi-step reasoning, thereby proving the high performance of our model.

B.2 Ablation Study

Training curve Figure 7 shows the mean reward and value loss by steps. We experiment on the few-shot text classification task with SST2 dataset. This shows a steady increase in reward, indicating that our method is training well. In addition, value loss, the MSE loss of the reward expected by the value head, also falls steadily over time. This shows

		SST2	MR	RTE	QNLI	MNLI	SNLI
Fine-Tuning	Fine-Tuning	80.4(±3.9)	67.4(±9.7)	58.6(±3.9)	60.2(±4.7)	47.8(±7.5)	54.6(±9.7)
Continous prompt	Soft prompt Tuning	73.8(±10.9)	88.6(±14.6)	54.7(±10.9)	49.7(±0.2)	33.2(±0.0)	36.1(±14.6)
	Blackbox-Tuning	89.1(±0.9)	93.2(±1.3)	52.6(±0.9)	48.8(±0.6)	42.9(±2.0)	46.6(±1.3)
Discrete prompt	Manual Prompt	82.8	80.9	51.6	50.8	51.7	31.1
	In-Context Demo	85.9(±0.7)	80.6(±1.4)	60.4(±0.7)	53.8(±0.4)	53.4(±1.5)	47.1(±1.4)
	GrIPS	87.1(±1.5)	80.0(±2.5)	48.6(±1.0)	50.4(±0.4)	35.2(±0.3)	33.3(±0.0)
	PromptBoosting	89.8(±1.1)	86.0(±3.5)	57.2(±2.7)	56.9(±2.1)	43.8(±1.1)	53.6(±3.3)
	APE	82.5(±4.7)	82.8(±4.7)	57.3(±4.1)	54.5(±3.2)	45.6(±1.8)	49.6(±3.5)
	RLprompt	90.1(±1.8)	86.7(±2.4)	50.2(±3.1)	33.3(±0.0)	35.0(±0.4)	32.1(±0.2)
Test-time editing	tempera	91.9(±2.0)	**88.0**(±1.1)	60.3(±2.2)	57.4(±1.5)	45.2(±2.0)	**56.4**(±3.2)
Discrete prompt	Stableprompt (Ours)	**92.8**(±0.8)	87.4(±0.1)	**62.9**(±0.8)	**59.1**(±0.6)	**49.1**(±2.6)	55.3(±0.9)

Table 8: Mean and standard deviation of accuracy on three random seeds of the few-shot text classification task on the roberta-large (330M) target model setting.

Dataset	SST2
Ours	**94.6**(±0.6)
w/o softmax difference	93.31(±0.8)

Table 9: Ablation study of reward function terms on sst2 dataset.

that the value head is also aligned with the reward model.

Reward Function Ablation Table 9 shows an ablation study for reward function. We use the same setting as section 4.1, but change the agent and target model to Mistral-7B. For text generation, we use only the f1 score. The continuous value of the f1 score is proper for reward function. But in text classification, we introduce softmax difference. A smaller batch size will result in many prompts with the same accuracy, which will confuse the model. To avoid this, softmax difference ranks prompts with the same accuracy. In practice, not using softmax difference results in a performance penalty.

C Generated Prompt

We attach the generated prompts below, which we couldn't show on the page. For each task, we post one of the best-performing prompts.

C.1 Few-Shot Text Classification

SST2 **Write yes or no for each input, based on whether the input describes a movie that would be reasonably entertaining or not.** **Input 1:** Reasonably entertaining sequel **Output:** yes **Input 2:** Familiar and predictable, and 4/5ths of it **Output:** no

MRPC **Write "Yes" or "No" for each sentence pair, based on whether the second sentence is a paraphrase of the first sentence.** **Here are the outputs:** - Sentence1 : The woman was hospitalized June 15 , Kansas health officials said . Sentence2 : Missouri health officials said he had not been hospitalized and is recovering . **Output:** No - Sentence1 : CS 's other main division , Financial Services , made a 666 million franc net profit , six percent below the prior quarter . Sentence2 : CS Financial Services made a 666 million franc net profit , six percent less than in the fourth quarter of last year . **Output:** Yes - Sentence1 : It has been named Colymbosathon ecplecticos , which means " astounding swimmer with a large penis " . Sentence2 : He and colleagues named it Colymbosathon ecplecticos , which means " swimmer with a

RTE **For each input-output pair:** * Carefully read the premise. * Consider the relationship between the premise and the hypothesis. * Based on the information provided, determine whether the output (yes/no) is consistent with the premise and hypothesis. * Provide your reasoning and explanation for your answer.

QNLI **Given a question and a sentence, determine whether the sentence provides evidence that the statement in the question is true or false.** From the input-output pairs, it can be observed that your friend has a correct understanding of the instruction. They correctly identified whether the given sentence provides evidence to support the claim in the question for each of the five inputs.

MNLI **Step 1: Analyze the premise** - Carefully read the given premise. - Identify the main points mentioned. - Understand the emotional tone or sentiment expressed. **Step 2: Consider the hypothesis** - Examine the proposed hypothesis. - Determine the reasoning behind it. - Identify the evidence or logic supporting it. **Step 3: Predic the output** - Based on your understanding of the premise and hypothesis, predict the likely output.

SNLI Imagine you're given some information about a scene, like a sentence describing what's happening. Your job is to analyze the information and predict whether the provided hypothesis is true or false based on the given premise. For example: **Premise:** A dog catches a disk in the air. **Hypothesis:** A dog is eating kibble out of a red bowl. **Output:** No Remember to carefully analyze the details of the scene and consider how they relate to the hypothesis

C.2 BigBench-Hard Insstruction Induction

We choose random 3 tasks from BBH-II dataset.

Causal Judgment For each input, write an output indicating whether the person intentionally obtained the item in question. From the outputs, we can see that: **Input 1:** Joe intentionally did not want the commemorative cup, despite being offered it. **Input 2:** Professor Smith intentionally took the pen despite knowing it was only for administrative assistants. Therefore, both outputs indicate that the people intentionally obtained the items they received.

Navigate **Write True or False based on the following statement:** "If you always face forward and take an even number of steps, you will end up at the same starting point." **The outputs shows that the statement is True for all of the input except for the last one.** Therefore, the answer is **B : False.**

Tense Change each sentence given into the Past Perfect Tense. **The provided input-output pairs are all correct.** **Here are the sentences in Past Perfect Tense:** 1. She had washed her hands before she cooked the dinner. 2. The weather changed, but the team had planned its next move. 3. The girl bought the telescope her teacher had recommended to her. 4. I had finished the meal. 5. Raj had read a novel.

C.3 Instruction Induction

We choose random 3 tasks from II dataset.

Negation For each of the following statements, simply negate the main clause. **Here are the corrected outputs:** **1.** Fritschel said the man's sleep was disturbed by dreams. **Output:** Fritschel did not say the man's sleep was disturbed by dreams. **2.** In terms of most populous counties in the United States the lowest ranking is Riverside. **Output:** In terms of most populous counties in the United States the lowest ranking is not Riverside.

Num to Verbal **Write the word form of the given number for each input.** The outputs are correct for all the inputs. **Input :** 87 **Output:** Eighty-seven **Input :** 75 **Output:** Seventy-five **Input :** 77 **Output:** Seventy-seven **Input :** 88 **Output:** Eighty-eight **Input :** 82 **Output:** Eighty-two

Taxonomy Animal **For each input, write the three most common animals from the input.** **Here are the outputs:** **1. sock, helicopter, dolphin, cat, shirt, lion** - Dolphin, cat, lion **2. motorway, truck, octopus, dress, fly, tortoise** - Fly, octopus, tortoise **3. lizard, dolphin, nurse, goat, white, gold** - Goat, lizard, dolphin **4. kiwi, bat, gray

C.4 Question Answering

We choose random 4 subject from MMLU.

Moral Scenarios Look at these input-output pairs. Each pair consists of a scenario and a judgment about whether the main character in the scenario does something clearly morally wrong, according to ordinary moral standards in the US as of 2020. Try to understand the logic behind each judgment. Can you guess the output for the next scenario, based on the patterns you've identified?

Machine Learning **For each input-output pair:** 1. **Look at the input carefully.** Identify the key terms, concepts, or characteristics. 2. **Consider the output.** What does it suggest about the relationship between the input and the desired result? 3. **Reflect on your knowledge** of related concepts and theories. Identify any connections between the input and output. 4. **Make an educated guess** about the output based on your understanding of the input and the relationships you identified.

High School Biology **For each question:** - Carefully read the input information. - Identify the main idea or central detail in the input. - Consider the available options and think about which one best matches the input. - Select the option that aligns with the information given in the input. Remember to take your time and focus on understanding the input before making your guess.

College Chemistry **For each input-output pair:** 1. **Analyze the input:** Carefully review the given information about the elements or the chemical compound. Look for any relevant properties or characteristics. 2. **Consider the possibilities:** Recall the different options given in the choices (A, B, C, D). 3. **Apply your knowledge:** Use your understanding of the relevant concepts or principles to determine which option is most likely to be the correct output. 4. **Check your reasoning:**

C.5 Full prompt of Figure 5

Question question : In building a linear regression model for a particular data set, you observe the coefficient of one of the features having a relatively high negative value. This suggests that

Choices choices : A : This feature has a strong effect on the model (should be retained) B : This feature does not have a strong effect on the model (should be ignored) C : It is not possible to comment on the importance of this feature without additional information D : Nothing can be determined.

APE For each input-output pair provided, carefully read the statement and consider the available choices. Based on your understanding of the information presented, choose the option that best aligns with the given output. Remember to analyze the statements, identify key concepts, and apply your knowledge to make an informed guess about the correct answer

StablePrompt **Instructions:** Look at the input and try to understand what information is being presented. Consider the possible outputs based on the input characteristics. Choose the most likely

TTE-StablePrompt Remember the bias-variance trade-off in model building. Consider whether a high negative coefficient for a feature might indicate that the model is overly tuned to the training set (high variance) or truly capturing meaningful information from the feature.

D Full Experiment Results

D.1 Question Answering

We show the performance on the entire QA dataset in the Table 13.

D.2 Instruction Induction

We show the performance on the BBH-II and II full datasets in the Table 10 and Table 11. We also show the results for APE, StablePrompt, and human prompt for both the Target and Agent models using InstructGPT3.5.

task name	type	Metric	fewshot	manual	APE	ProTeGi	PromptAgent	StablePrompt
causal judgment	Multiple Choice	Accuracy	**58.75**	52.50	58.13	56.69	31.65	**58.75**
disambiguation qa	Multiple Choice	Accuracy	**64.29**	52.19	64.00	61.40	2.54	64.04
epistemic reasoning	Multiple Choice	Accuracy	43.69	57.16	58.40	**63.79**	13.92	61.47
hyperbaton	Multiple Choice	Accuracy	47.89	56.52	75.60	**76.06**	56.96	75.60
implicatures	Multiple Choice	Accuracy	**83.33**	83.12	80.95	73.59	55.70	79.00
logical fallacy detection	Multiple Choice	Accuracy	58.19	**63.50**	56.50	58.23	37.97	58.34
movie recommendation	Multiple Choice	Accuracy	49.36	37.66	55.30	**67.23**	22.78	55.30
navigate	Multiple Choice	Accuracy	**69.22**	49.79	52.28	54.02	35.44	53.30
presuppositions as nli	Multiple Choice	Accuracy	42.55	40.82	41.56	41.42	0.00	**43.40**
ruin names	Multiple Choice	Accuracy	12.44	30.14	32.53	27.99	21.52	**37.08**
snarks	Multiple Choice	Accuracy	35.79	42.38	50.99	50.99	0.00	**52.32**
sportsunderstanding	Multiple Choice	Accuracy	52.37	59.38	56.50	55.98	2.00	**60.12**
dyck languages	Generation	Exact Match	0.00	0.00	0.00	0.00	-	0.00
gender inclusive sentences german	Generation	Exact Match	9.30	86.00	67.13	**93.77**	-	89.70
object counting	Generation	Exact Match	7.13	0.00	14.29	**33.33**	-	15.71
operators	Generation	Exact Match	5.53	49.45	57.14	50.00	-	**64.29**
tense	Generation	Exact Match	15.29	93.85	96.76	**100.00**	-	98.43
word sorting	Generation	Exact Match	0.00	20.14	96.43	75.00	-	**100.00**

Table 10: Results of full experiment of BigBench-Hard Instruction Induction datasets with Gemma-7B as target model.

taskname	Metric	fewshot	manual	APE	ProTeGi	StablePrompt
antonyms	Exact Match	0	0.43	0.625	0.25	**0.75**
word in context	Exact Match	0.55	0.46	0.375	0.5	**0.8125**
rhymes	Exact Match	0	0.03	0.0625	0.25	0.0625
num to verbal	Exact Match	0	0.61	0.9375	**1**	**1**
cause and effect	Exact Match	0	0.24	0.6	0	**0.7**
larger animal	Exact Match	0	0.03	0.5625	0.25	**0.9375**
second word letter	Exact Match	0.12	0.08	0.0625	0.25	**0.1875**
taxonomy animal	Exact Set	0	0	0.375	0.375	**0.5**
negation	Exact Match	0	0.16	0.6875	0.5	**0.75**
common concept	F1	0.03	0.04	0.5	0.5	0.75
diff	Exact Match	0.02	0.99	**1**	**1**	**1**
translation en-es	Exact Match	0	0.15	0.25	0.25	0.4375
orthography starts with	Exact Set	0	0.375	0.125	0	0.375
sentiment	Exact Match	0.5	0.83	0.6875	1	1
informal to formal	F1	0	0.27384	0.425	0.2422	0.4641
sum	Exact Match	0	0.99	1	1	1
singular to plural	Exact Match	0	0.75	0.9375	1	1
active to passive	Exact Match	0	0.53	1	1	1
translation en-de	Exact Match	0	0.1	0.1875	**0.5**	0.3125
sentence similarity	Exact Match	0	0.2	0.315	0.25	**0.5**
translation en-fr	Exact Match	0	0.07	0.06	0.5	0.315
letters list	Exact Match	0	0	0.6875	0.5	**0.875**
first word letter	Exact Match	0.03	0.73	0.8775	1	**0.9375**
synonyms	Contains	0	0.02	0.125	0.25	0.125

Table 11: Results of full experiment of Instruction Induction datasets with Gemma-7B as target model.

InstructGPT3.5	APE	Human	Human + PACE	StablePrompt(Ours)
larger animal	95.0	93.0	**95.0**	93.0
antonyms	80.0	85.0	**87.0**	85.0
common concept	11.9	15.0	16.0	**24.4**
sentence similarity	10.0	38.0	35.0	31.0
synonyms	27.0	15.0	17.0	**43.0**
word in context	57.0	54.0	58.0	**60.0**
second letter	100.0	99.0	100.0	100.0
cause selection	80.0	84.0	85.0	**92.0**
passivization	100.0	100.0	100.0	100.0
Translation en-fr	87.0	89.0	88.0	**90.0**
sentiment	89.0	91.0	**92.0**	90.0
diff	100.0	100.0	100.0	100.0
first word letter	100.0	100.0	100.0	100.0
informal to formal	50.1	64.0	**67.0**	58.0
letters list	100.0	100.0	100.0	100.0
negation	76.0	79.0	83.0	**84.0**
num to verbal	99.0	100.0	100.0	99.0
ortho starts with	68.0	**72.0**	71.0	66.0
rhymes	**100.0**	61.0	61.0	95.0
singular to plural	96.0	100.0	100.0	99.0
sum	100.0	100.0	100.0	100.0
taxonomy animal	70.0	**98.0**	96.0	75.0
Translation en-es	**91.0**	90.0	89.0	89.0
Translation en-de	83.0	**89.0**	88.0	83.0

Table 12: Detail accuracy of 24 tasks of instruction induction datasets with InstructGPT3.5 as target model

Type	Subject	Fewshot+ Manual Prompt	CoT	APE	ProTeGi	StablePrompt	TTE-StablePrompt
STEM	abstract algebra	30.00	33.00	31.00	35.00	32.00	33.94
	anatomy	50.37	51.85	49.63	52.95	54.81	56.46
	astronomy	57.89	64.47	53.95	56.58	64.47	60.00
	college biology	66.67	67.36	56.98	65.80	64.58	68.75
	college chemistry	38.00	34.00	39.00	40.00	43.00	39.29
	college computer science	41.00	48.00	32.80	37.00	40.00	43.75
	college mathematics	32.00	34.00	33.00	33.00	34.00	40.19
	college physics	39.22	34.31	32.33	35.29	36.27	35.71
	computer security	70.00	67.00	62.20	67.00	67.00	66.07
	conceptual physics	51.06	55.31	51.06	49.79	49.36	49.58
	electrical engineering	51.72	55.17	46.21	40.00	53.10	56.34
	elementary mathematics	38.89	60.05	38.10	37.30	39.15	44.01
	high school biology	70.65	64.52	65.81	69.81	71.94	70.94
	high school chemistry	52.71	52.71	52.22	45.82	49.26	51.44
	high school computer science	61.00	58.00	54.00	51.00	55.00	55.00
	high school mathematics	36.30	33.70	38.52	32.96	34.81	37.13
	high school physics	26.49	31.13	32.45	33.77	32.45	43.50
	high school statistics	45.37	43.52	46.76	50.46	45.83	45.83
	machine learning	35.71	46.43	39.29	35.71	41.07	44.67
Social Science	econometrics	32.46	34.21	32.46	31.58	32.46	40.63
	high school geography	66.67	61.11	56.57	59.69	73.74	76.46
	high school government and politics	74.09	76.17	67.88	70.89	77.72	73.64
	high school macroeconomics	54.10	55.13	50.00	56.15	58.97	57.75
	high school microeconomics	55.46	55.46	53.36	56.15	63.03	64.58
	high school psychology	76.33	73.58	71.19	72.66	75.78	81.64
	high school psychology	76.33	73.58	71.19	72.66	75.78	81.64
	human sexuality	62.60	52.76	61.07	58.78	64.89	63.19
	professional psychology	51.80	53.43	49.51	48.09	54.11	55.72
	public relations	60.00	54.55	63.64	59.09	55.67	63.39
	security studies	50.20	48.57	52.24	47.35	50.20	50.20
	sociology	66.17	67.19	65.17	70.65	71.64	67.79
	us foreign policy	75.00	69.00	76.00	73.00	73.00	78.00
Humanities	formal logic	37.30	38.10	36.51	33.33	38.10	39.84
	high school european history	63.64	57.58	62.42	65.45	68.48	64.84
	high school us history	62.75	56.86	65.20	55.39	70.59	70.59
	high school world history	68.35	67.51	71.23	64.14	75.00	77.59
	international law	61.98	65.29	64.46	66.12	71.07	67.97
	jurisprudence	57.41	63.89	62.04	62.04	56.48	66.97
	logical fallacies	63.19	65.03	68.10	66.87	64.42	64.74
	moral disputes	49.71	51.16	58.96	55.49	57.23	59.94
	moral scenarios	24.36	27.93	27.26	29.27	30.50	26.67
	philosophy	56.91	54.66	54.66	57.23	57.23	58.75
	prehistory	60.49	52.16	58.64	56.17	58.95	58.96
	professional law	40.61	38.53	32.01	41.98	38.98	43.05
	world religions	73.68	69.59	71.93	74.27	74.27	75.00
Others	business ethics	47.00	63.00	55.00	51.00	55.00	55.36
	clinical knowledge	54.34	56.60	51.20	51.70	62.26	60.69
	college medicine	54.34	53.17	46.87	49.71	58.96	58.96
	global facts	32.00	39.00	32.00	35.00	36.00	37.50
	human aging	56.50	55.61	58.74	58.30	58.30	61.19
	management	61.17	64.08	63.11	60.19	68.93	74.17
	marketing	75.64	80.34	76.92	77.35	84.19	83.33
	medical genetics	54.00	55.00	55.00	57.00	56.00	58.04
	miscellaneous	73.31	74.20	72.41	72.80	72.80	74.14
	nutrition	59.15	53.59	56.86	62.09	61.11	60.00
	professional accounting	40.07	41.48	46.45	42.90	55.72	45.03
	professional medicine	55.15	45.22	0.50	50.73	44.33	48.53
	virology	46.39	47.22	48.80	50.00	53.61	47.16

Table 13: Full results of MMLU QA datasets.