OFFLINE PROMPT EVALUATION AND OPTIMIZATION WITH INVERSE REINFORCEMENT LEARNING

Hao Sun [*]
DAMTP, University of Cambridge
hs789@cam.ac.uk

(Temporary) Anonymous Coauthors
Anonymous Affiliation
Anonymous email

ABSTRACT

The recent advances in the development of Large Language Models (LLMs) like ChatGPT have achieved remarkable performance by leveraging human expertise. Yet, fully eliciting LLMs' potential for complex tasks requires navigating the vast search space of natural language prompts. While prompt engineering has shown promise, the requisite human-crafted prompts in trial-and-error attempts and the associated costs pose significant challenges. Crucially, the efficiency of prompt optimization hinges on the costly procedure of prompt evaluation. This work introduces Prompt-OIRL, an approach rooted in offline inverse reinforcement learning that seeks to bridge the gap between effective prompt evaluation and affordability. Our method draws on offline datasets from expert evaluations, employing Inverse-RL to derive a reward model for offline, query-dependent prompt evaluations. The advantages of Prompt-OIRL are manifold: it predicts prompt performance, is cost-efficient, produces human-readable results, and efficiently navigates the prompt space. We validate our method across four LLMs and three arithmetic datasets, highlighting its potential as a robust and effective tool for offline prompt evaluation and optimization. Our code as well as the offline datasets are released, and we highlight the Prompt-OIRL can be reproduced within a few hours using a single laptop using CPU [1].

1 INTRODUCTION

Human Expertise Helps ML. Machine learning algorithms have achieved human-level performance or even surpassed it in various domains such as computer vision, natural language processing, and robotics by incorporating human expertise (Deng et al., 2009; Zhang et al., 2018; Silver et al., 2016; Vinyals et al., 2019). Recent advances in Large Language Models (LLMs), such as ChatGPT (Ouyang et al., 2022), underscore the importance of injecting human knowledge through learning from human feedback (Christiano et al., 2017; Ouyang et al., 2022; OpenAI, 2023), which not only reduces potential harm but also enhances the model's capacity to follow instructions.

Prompt Optimization is Essential yet Challenging. In tasks as intricate as arithmetic reasoning, even state-of-the-art, general-purpose LLMs such as GPT-4 are prone to shortcomings. Unlocking their full potential for those complex tasks remains to be a formidable challenge. Out of the many attempts, prompt engineering stands out as a promising solution for eliciting the capabilities of LLMs without necessitating optimization. Although the efficacy and elegantly of zero-shot prompting (Kojima et al., 2022) reveals the potential of prompting, the reliance on manually crafted prompts and the vast search space over natural language prompts compound the challenge of effective prompts discovery (Deng et al., 2022). While several automated generators have been proposed to improve the efficiency in navigating the prompt space (Hao et al., 2022; Pryzant et al., 2023), how to *effectively generate human-readable prompts according to the context without access to the LLMs* remains an open question. Essentially, such an objective poses 4 challenges: **(1) effective:** the prompt should reinforce LLMs; **(2) human-readable:** so that the reinforcement can be more transparent; **(3) cost(interaction)-efficient:** to make the prompting optimization more affordable and practically

[*]Preliminary Work. Hao Sun is one of the main contributors.

[1]with our implementation, conducting OIRL for the GSM8k takes 50 minutes on a MacBookAir with M2 chip, and takes only 5 minutes on a server with 16(out of 64)-core AMD 3995WX CPUs.

feasible; **(4) context-awareness:** the best prompting practice should be query-dependent. While the first two challenges are relatively self-explanatory, the final two warrant further elaboration:

Challenge of Cost-Efficiency: Online Prompt Evaluation is Expensive. One of the critical prerequisites for prompt optimization is prompt *evaluation*. This step presents a significant challenge in the optimization process due to its high cost. For instance, consider evaluations using the GPT-3.5-turbo API (OpenAI, 2023): Evaluating a single prompt on a medium-sized ***training*** dataset with 10k query-answer pairs costs around \$1, yet reinforcement learning often requires millions of interactions with the environment. Such a prohibitive cost restricts the iterative process of learning from trial and error in the vast space of natural language, as each iteration incurs such a considerable expense (Zhang et al., 2022a; Deng et al., 2022).

Challenge of Context-Awareness: the Significance of Query-Dependent Prompt Evaluation. Beyond the expenses, an optimal evaluation framework should have the ability to *predict* the inference-time performance of a prompt for a specific query. For instance, when presented with an arithmetic question, a desired prompt evaluator should predict whether zero-shot CoT prompting (Kojima et al., 2022) might increase the likelihood of obtaining a correct answer, or if alternative prompting techniques such as those proposed by Zhou et al. (2022b) would be more effective. Ideally, this query-dependent evaluation should be offline (i.e., no interaction with LLMs). This is because the ground-truth answer during inference time remains unknown, making real-time interactions with LLMs potentially ineffective in selecting prompts.

Solution: Offline Prompt Evaluation and Optimization with Inverse Reinforcement Learning. In this work, we are inspired by the significant advancements achieved in prompt engineering, including both expert-crafted and algorithm-discovered prompts (Zhou et al., 2022b; Pryzant et al., 2023; Kojima et al., 2022), and motivated to seek an answer to the question of

How to <u>effectively</u> evaluate and optimize prompt with human expertise?

However, integrating human expertise can be financially expensive. As a consequence, to develop a practical prompting algorithm, we simultaneously grapple with:

How to <u>efficiently</u> inject human expertise without an increased expense?

To address the efficacy question, we formulate the **problem of prompting as policy learning**. Consequently, the inverse-RL can be applied to integrate existing human expertise in prompting; To address the efficiency question, we employ offline inverse-RL (OIRL) for prompt evaluation and optimization. We introduce Prompt-OIRL, which achieves all the desired properties discussed above in an efficient offline approach. Specifically, Prompt-OIRL utilizes the readily available **offline** datasets generated when experts evaluate their crafted prompts. We apply inverse-RL to first learn a proxy reward model capable of conducting query-dependent offline prompt evaluations. Subsequently, we use the learned reward model as an offline prompt evaluator to further conduct query-dependent prompt optimization. Prompt-OIRL offers multiple advantages:

- **Capable of Query-Dependant Offline Prompt Evaluation:** Given the learned dense reward function in IRL, we have an efficient and effective proxy that estimates the performance using different prompts — for the first time, we show the performance of prompting is predictable;
- **Low-Cost:** By learning with offline prompt-evaluation interaction logs, we circumvent the excessive demand of querying the costly LLMs;
- **Human-Readable:** By learning from expert prompts, we leverage the existing human expertise in prompt engineering and find new prompting strategies that are still in a human-readable form;
- **Efficient:** By learning from LLM feedback contained in the offline dataset, we avoid the need for random exploration of the vast prompt space yet still extrapolate beyond existing expert-crafted prompts to find better-performing prompting strategies.

Through experiments with 4 distinct LLMs, namely GPT-3.5, GPT-4, LLaMA-2-7B-Chat, and TigerBot-13B, across three arithmetic datasets: GSM8K (Cobbe et al., 2021), MAWPS (Roy & Roth, 2016), and SVAMP(Patel et al., 2021), we validate the *efficacy* and *efficiency* of Prompt-OIRL in offline prompt evaluation and optimization.

2 Prompting Problem as Policy Learning

Queries & Answers We consider the task of answering queries $x \in \mathcal{X} = \mathcal{V}^\infty$ expressed in a natural language with vocabulary $\mathcal{V}$. We assume that each query x has an expected answer $y^* \in \mathcal{Y}$. For example, x can be an arithmetic question and y^* the mathematically-correct answer to the question, or x can be the review of a movie and y^* a sentiment label for the movie.

Language Model Such tasks can be performed using a language model $\ell : \mathcal{X} \to \mathcal{Y}$ by feeding queries x to the language model and getting answers

$$\hat{y} = \ell(x) \tag{1}$$

We assume that the quality of these answers is evaluated using a metric $r : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ such that the higher $r(y^*, \hat{y})$ is, the 'better' $\hat{y}$ matches the expected answer y^*. For example, in tasks where golden labels are available, this metric can be $r(y^*, \hat{y}) = \mathbb{1}\{\hat{y} = y^*(x)\}$.

Prompting It is well known that the performance of a language model in answering queries can be boosted through appropriate prompting—a natural language prefix or instruction that explains how to complete the task. We consider prompting strategies $\pi : \mathcal{X} \to \mathcal{X}$ that maps original queries x to modified prompts $x' = \pi(x)$. These prompts are then fed into the language model to get (ideally better) answers

$$\hat{y} = \ell(x' = \pi(x)) \tag{2}$$

Objective Given a dataset $\mathcal{D} = \{x^{(i)}, y^{*(i)}\}_{i=1}^N$ of queries and their expected answers, the objective of prompt optimization is to find prompting strategies that maximize the quality of answers:

$$\max_\pi r(y^*, \ell(\pi(x))) \tag{3}$$

It is worth noting that Equation (3) indicates the prompting strategy can be query-dependent, as opposed to, for instance, the query-agnostic zero-shot prompting strategies.

3 Prompting with Offline Inverse RL

Offline Demonstrations We highlight the existence and importance of prompt demonstrations generated in benchmarking existing prompting strategies — for instance, $\pi^{(1)}(x) = (x)$ is the trivial prompting, $\pi^{(2)} =$ zero-shot CoT prompting, $\pi^{(3)} =$ ToT prompting, and so on — the performance of those prompting strategies have been benchmarked from interactions the language model ℓ, and stored as logged datasets. Denoting with K the total number of prompting strategies considered, these demonstrations are constructed as

$$\mathcal{D}_{\text{dem}} = \{x^{(i)}, \pi^{(j)}, r^{(ij)} = r(y^{*(i)}, \ell(\pi^{(j)}(x^{(i)}))\}_{i\in\{1,\ldots,N\}, j\in\{1,\ldots,K\}} \tag{4}$$

Offline Inverse RL: Reward Modeling Following the key insight of Brown et al. (2019), we first learn a proxy reward model that can evaluate any given prompts, i.e., can evaluate any prompting strategy given any query. We denote the proxy rewards as $\hat{r}_\theta(x, \pi(x))$ with parameters θ.

In this work, we focus on the arithmetic reasoning tasks where the reward signal is binary. In this case, the reward model can be trained by performing a classification task, trying to determine whether the prompt can result in a correct answer when fed to the language model. Specifically, we consider the Cross-Entropy loss:

$$\mathcal{L}_{\text{CE}}(\theta; \mathcal{D}_{\text{dem}}) = -\mathbb{E}_{i\in\{1,\ldots,N\}, j\sim\{1,\ldots,K\}} \left[r^{(ij)} \log \sigma\left(\hat{r}_\theta\right) + (1 - r^{(ij)}) \log\left(1 - \sigma\left(\hat{r}_\theta\right)\right) \right] \tag{5}$$

where we use $\hat{r}_\theta$ to denote $\hat{r}_\theta(x^{(i)}, \pi^{(j)}(x^{(i)}))$ for conciseness, and σ is the sigmoid function. Different from the *online* reward signal provided by $r(y^*, \ell(x'))$ that requires expensive interactions with black-box LLMs, the proxy reward model $\hat{r}_\theta(x, \pi(x))$ can provide *offline* white-box feedback to prompts. Therefore, it can be more accessible in guiding the search for a better prompting strategy.

Connections to Reinforcement Learning from Human (AI) Feedback Before developing the prompt optimization step using the learned reward model, it would be necessary to first link and contrast Prompt-OIRL with the framework of reinforcement learning from human feedback (RLHF) or AI feedback (RLAIF). In general, the success of RLHF and RLAIF motivates our idea of integrating human expertise in prompting, and Prompt-OIRL can be interpreted as a special type of prompt policy learning from ***offline*** AI feedback, which differentiates from existing literature.

The fact that Prompt-OIRL first learns a reward model and then performs policy optimization using the reward model makes the method related to RLHF and RLAIF. From such a perspective, Prompt-OIRL treats prompt evaluation and optimization as a process of *aligning prompting strategies with the preferences of the LLMs*, contrasting to aligning LLM responses with human preferences. In analogy with the RLHF that improves the performance of LLMs with regard to human preferences, we hypothesize that such a perspective could lead to more effective prompts while avoiding the need for random exploration of the prompt space, which is computationally expensive.

That said, there are several key differences that make Prompt-OIRL distinguishable from the existing RLHF and RLAIF literature. RLHF (or RLAIF) often includes the following steps: 1. sampling from pre-trained language models to generate dialogue data; 2. human (or AI) raters are then asked to rank those generated data according to given criteria such as harmless and helpful; 3. a preference model is trained on the ranked dataset, as proxy to the human (or AI) raters; 4. the language model is fine-tuned with the learned preference model using reinforcement learning.

By contrast, we would like to highlight the differences in objective, non-interactive offline learning problem setting, and optimization procedure flexibility: First, Objective: The objective of Prompt-OIRL is to evaluate and optimize prompting strategies, rather than enhance LLMs' alignment to non-differentiable objectives. Importantly, the learned reward model in Prompt-OIRL can work in isolation as a prompt evaluator. Second, Non-Interactive Offline Setting: Prompt-OIRL works in a purely offline setting, rather than assuming access to human or AI raters to *actively* generate feedback. Third, the optimization procedure in Prompt-OIRL is highly flexible, which is not necessary to be a language model as in RLHF and RLAIF. For instance, the prompt optimization process can collaborate with a human prompting engineer, who proposes potential prompting strategies and selects the best according to the learned reward model.

Prompt Optimization with the Reward Model. Given the learned reward model is an effective proxy of the performance evaluator for query-prompting strategy pairs, we would then be able to solve the prompt optimization problem of Euqation (3) with an alternative ***offline*** objective that is feasible to execute in inference-time — without the requirement of a language model ℓ and the non-accessible inference-time golden label y^*:

$$\max_{\pi} \hat{r}(x, \pi(x)) \tag{6}$$

In general, any policy optimization technique has the potential to be applied in solving Equation (6). To name a few examples, such a technique can be any of the previous approaches used in ***online*** prompt optimization like RL (Deng et al., 2022), beam search (Pryzant et al., 2023), evolution strategies (Zhou et al., 2022b).

In this work, we would like to highlight the effectiveness of optimizing prompts with regard to such a learned reward model as an accessible offline proxy to the true reward. We choose to use a general-purpose language model to generate a batch of candidate prompts and select the best one according to the learned reward model as a minimal example to isolate the source of gains. We leave the investigation of other approaches, which could be more tailored and better-performing, yet computationally more expensive as a promising direction for future exploration.

Readers interested in extended discussions on Offline Inverse Reinforcement Learning, Prompt Optimization Alternatives, and Off-Policy Evaluation please refer to the appendix.

4 Related Work

Automatic Prompt Engineering The Automatic Prompt Optimization (APO) (Pryzant et al., 2023) leverages training data and initial prompts to perform a "gradient descent" process operated in the natural language space guided by large language models. A generate-and-select process is then applied to update the prompting mechanism. Automatic Prompt Engineer (APE) (Zhou et al., 2022b) asks LLMs to generate a set of instruction candidates based on demonstrations and then

Table 1: Prompt-OIRL differentiates from existing literature on Prompt Generation by (1) working in the offline setting **without access to the LLMs**; (2) being able to perform **offline prompt evaluation**; (3) generating **query-dependent prompts**; (4) **utilizing existing expert knowledge** to reduce the difficulty in RL; (5) generating **human-readable prompts**; (6) having the most general **natural language prompt space**; (7) **free from gradient** information of LLMs.

Method	Offline Learning	Prompt Evaluation	Query Dependent Prompt	Expert Heuristics Inspired	Human Readable Prompt	Prompt Space*	LLM Gradient Free	Solver
Soft-Prompt	✗	✗	✓	✗	✗	Embeddings	✗	Gradient-Guided Search
APO	✗	✗	✗	✓	✓	$\mathcal{X} = \mathcal{V}^\infty$	✓	Beam Search
APE	✗	✗	✗	✗	✓	$\mathcal{X} = \mathcal{V}^\infty$	✗	Evolution Strategy
TEMPERA	✗	✗	✓	✓	✓	Edit	✓	RL
RLPrompt	✗	✗	✗	✗	✗	$\{\mathcal{V}^2, \mathcal{V}^5\}$	✓	RL
Prompt-OIRL (ours)	✓	✓	✓	✓	✓	$\mathcal{X} = \mathcal{V}^\infty$	✓	Offline Inverse-RL

* Embeddings: the language of LMs; Edit: including operations like swap, delete, etc.; $\mathcal{V}$ is the vocabulary, and the superscript over it denotes the length of prompts. $\mathcal{V}^\infty$ denotes the natural language space — the most general interpretable format.

evaluate the performance of those instructions by computing corresponding scores, finally, APE improves the best candidates by proposing semantically similar instruction variants to specific tasks. There are also works that explore the usage of auxiliary models or differentiable representations of the prompt (Shin et al., 2020; Li & Liang, 2021; Qin & Eisner, 2021; Vu et al., 2021; Gu et al., 2021). Yet those methods require access to the embeddings of the LMs for soft prompt optimization, hence falling short in reusability across LMs. In the work of Hao et al. (2022), reinforcement learning is used to optimize a promptist in text-to-image generation tasks in pursuance of more aesthetically pleasing images preserving the original user intentions. Zhang et al. (2022a) proposes TEMPERA to perform test time prompt editing using reinforcement learning. TEMPERA designs the action space to be editing operations, such as swapping, deleting, adding, or changing verbalizers. The most relevant work is RLPrompt Deng et al. (2022), which uses reinforcement learning to optimize LLMs to specialize as prompting agents. Yet their prompts generated by RLPrompt are task-agnostic, and limited to natural language word combinations that can not be easily transferred to insights for human prompt engineers.

The distinctions between this study and related works are summarized and highlighted in Table 1.

Learning with Human Expertise and Imitation Learning Human expertise has pushed forward the progress of machine learning in many domains, such as computer vision (Deng et al., 2009; Redmon et al., 2016), natural language processing (Rajpurkar et al., 2018; Ouyang et al., 2022), and robotics (Atkeson & Schaal, 1997; Zhang et al., 2018; Mandlekar et al., 2020). By injecting prior knowledge of the task from human experts, machine learning algorithms are able to achieve human-level performance or even outperform human (Silver et al., 2016; Vinyals et al., 2019).

Imitation learning and learning from demonstrations are widely studied in the field of reinforcement learning and robotic learning (Ng et al., 2000; Abbeel & Ng, 2004; Ho & Ermon, 2016; Garg et al., 2021), aiming at learning policies with a batch of expert trajectories. Working as the most straightforward solution, Behavior Cloning (BC) (Pomerleau, 1988; Florence et al., 2022) optimizes a policy through supervised learning, yet may suffer from compounding error or instability in learning from sub-optimal demonstrations (Ross et al., 2011; Wu et al., 2019). It is shown in Mandlekar et al. (2021) that data quality is of great importance in the success of BC. It is worth noting that, while imitation learning assumes the underlying reward mechanism is unknown, the learning from demonstration literature always uses the demonstrations as a warm-start for reward-sparse tasks like robotics manipulation (Schaal, 1996; Nair et al., 2018).

In prompt generation tasks, the demonstrations are always imperfect, T-REX (Brown et al., 2019) extrapolates beyond a learned reward function to achieve super-demonstration performance. We build our work based on T-REX to address the difficulty of learning from imperfect demonstration. We also benchmark the performance of BC, with a specific consideration on the quality of demonstrations (Wu et al., 2019; Mandlekar et al., 2021).

5 Experiment

We aim to provide empirical evidence for the following quantitative questions: ***1. How effective is Prompt-OIRL?*** We answer this question from two different perspectives: First, we benchmark the generalization ability of the learned reward model by showing their prediction performance on held-out query and held-out prompting strategies in Section 5.1; we then benchmark the performance of optimized prompting strategies on different arithmetic reasoning tasks in Section 5.2. ***2. How efficient is Prompt-OIRL?*** We answer this question in Section 5.3 by comparing alternative approaches requiring additional interactions with general-purpose LLMs that can be adopted and used as prompt evaluators.

Tasks We choose to use the arithmetic reasoning domain and the tasks of MultiArith (Roy & Roth, 2016), GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021) because they are widely studied in zero-shot prompting such that rich expert-crafted prompting knowledge is available, and many of those prompts have been benchmarked on those tasks, facilitating our offline data collection procedure.

Prompting Strategies We consider 6 existing zero-shot prompting strategies in constructing our offline demonstration dataset, as described in Table 2

Table 2: Prompting strategies used in offline training dataset collection.

No.	Effective Prompts Discovered by Experts and Algorithms	Explanation
1	"The answer is:"	direct prompting
2	"Let's think step by step:"	zero-shot CoT (Kojima et al., 2022)
3	"Let's work this out in a step by step way to be sure we have the right answer:"	APE discovered (Zhou et al., 2022b)
4	"First, decompose the question into several sub-questions that need to be solved, and then solve each question step by step:"	Least-to-most (Zhou et al., 2022a)
5	"Imagine three different experts are answering this question. All experts will write down 1 step of their thinking, and then share it with the group. Then all experts will go on to the next step, etc. If any expert realizes they're wrong at any point then they leave."	Tree-of-thought (Hulbert, 2023)
6	"3 experts are discussing the question, trying to solve it step by step, and make sure the result is correct:"	multi-agent debate (Liang et al., 2023)

LLMs To demonstrate the general applicability of Prompt-OIRL, we experiment with datasets generated with LLMs at different scales, including the GPT 3.5 model (Ouyang et al., 2022), TigerBot-13B-chat (TigerResearch, 2023), and LLaMA-2-7B-chat (Touvron et al., 2023)

Offline Data Processing and Embeddings For LLM ℓ and task k, we re-organize the offline demonstration dataset using the embedding function $\mathcal{E} : \mathcal{V}^\infty \mapsto \mathbb{R}^{1536}$, which maps a sequence of natural language context to a fix-length vector of size 1536 (Greene et al., 2022). Therefore, using $e_x^{(i)} = \mathcal{E}(x^{(i)}), e_\pi^{(j)} = \mathcal{E}(\pi^{(j)}), r^{(ij)} = \mathbb{1}\{y^* = \ell(\pi^{(j)}(x^{(i)}))\}$ to denote the embeddings and reward instantiation, our demonstration dataset Equation (4) in implementation can be expressed as follows:

1. **Training Data:** $\mathcal{D}_{\ell,k}^{(\text{train})} = \{e_x^{(i)}, e_\pi^{(j)}, r^{(ij)}\}_{i\in\{1,...,N\},j\in\{1,...,K\}}$
2. **Test Data on Query:** $\mathcal{D}_{\ell,k}^{(\text{test q})} = \{e_x^{(i)}, e_\pi^{(j)}, r^{(ij)}\}_{i\in\{N+1,...,N+M\},j\in\{1,...,K\}}$
3. **Test Data on Prompts:** $\mathcal{D}_{\ell,k}^{(\text{test p})} = \{e_x^{(i)}, e_\pi^{(j)}, r^{(ij)}\}_{i\in\{N+1,...,N+M\},j\in\{K+1,...,K+P\}}$

The GSM8K task contains 7473 queries with golden answers for training and 1319 held-out queries with golden answers for testing; the SVAMP task contains 19690 examples, which are split into a training query-answer set of size 15000 and a testing query-answer set of size 4690; the MAWPS task contains 7685 examples, which are split into a training query-answer set of size 6000 and a testing query-answer set of size 1685. Therefore, for each prompting strategy, there are 7473, 15000, and 6000 demonstrative examples that can be collected through the interactive logs with a certain language model for the three tasks that can be used for training, respectively. Our processed offline datasets and code for processing will be provided as open-source assets to facilitate future studies.

Reward Modeling: Implementation Matters In our experiments, we find the gradient boosting methods (Chen et al., 2015; Ke et al., 2017) are significantly better than using neural networks in reward modeling — in terms of both computational efficiency and performance. We demonstrate the effectiveness using a minimalist approach by directly applying the XGBoost models and leaving further investigation on model selection to future work. That said, to exclude some potential alternatives — we have explored using fully connected neural networks as the reward model or using a ranked dataset mimicking the conventional RLHF setting, yet both of those choices perform worse than our presented implementation.

5.1 Offline Prompt Evaluation: How good is the reward model?

Experiment Setup In order to benchmark the effectiveness of the reward model, we verify its generalization ability by (1) evaluating training-time prompting strategies for held-out queries and (2) evaluating held-out prompting strategies for held-out queries. To enable the latter verification, we additionally collect interactive logs on each language model for 10 held-out prompting strategies for evaluation. We compare against using the language models for self-criticism as baselines (Wang et al., 2023), where LLMs are used to check whether an answer is correct given both the query and the answer as input. In all experiments, we evaluate the performance with **held-out test queries**. In the following results, we use **Train Prompts** to denote the evaluation results using training time prompts, and use **Test Prompts** to denote the evaluation on held-out prompts.

Results We present the accuracy of different methods on different models and datasets in Figure 1 - Figure 3, and precision in Figure 4 - Figure 6.

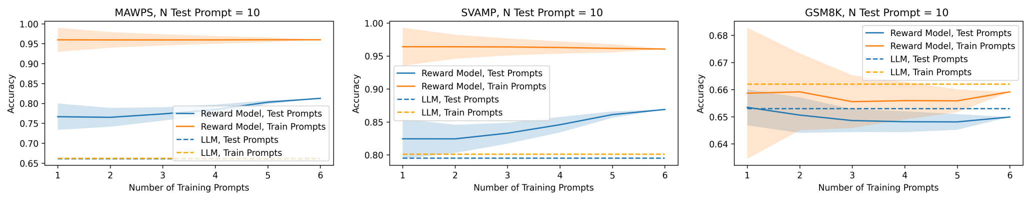

Figure 1: Accuracy on GPT 3.5 Turbo

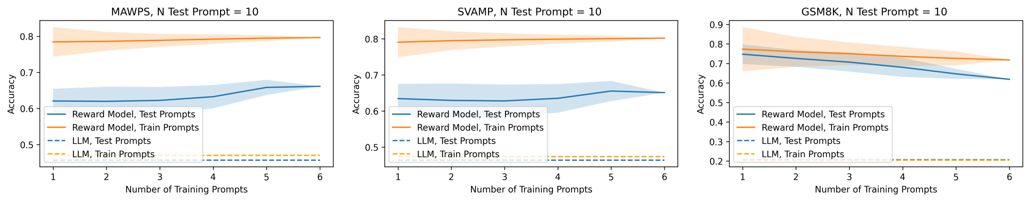

Figure 2: Accuracy on LLaMA-2-7B-Chat

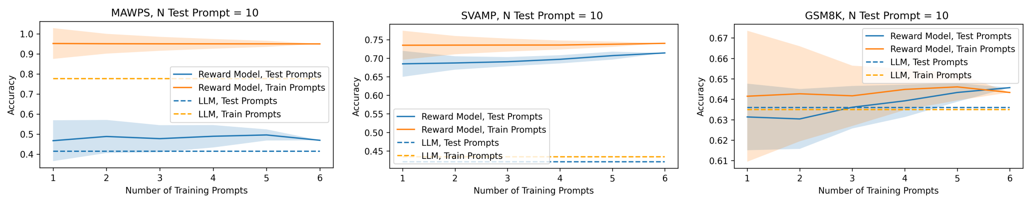

Figure 3: Accuracy on TigerBot-13B-Chat

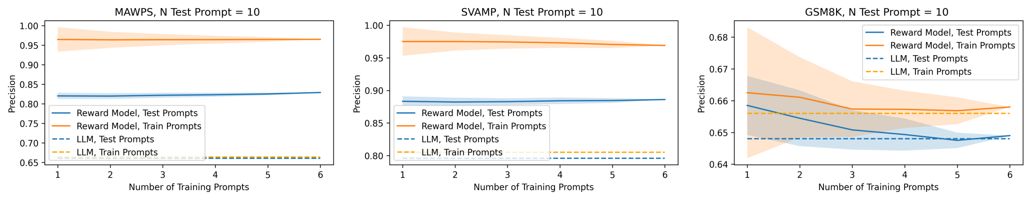

Figure 4: Precision on GPT 3.5 Turbo

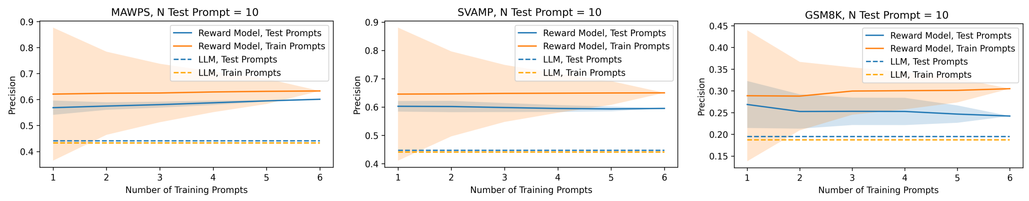

Figure 5: Precision on LLaMA-2-7B-Chat

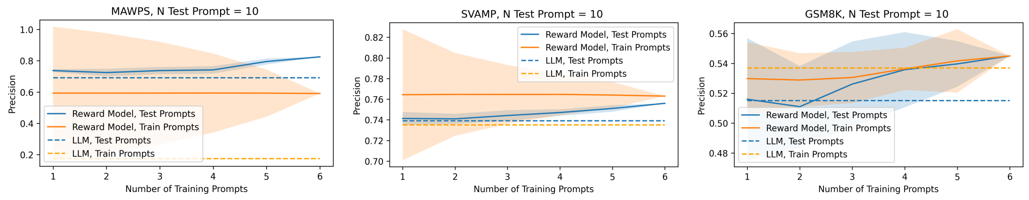

Figure 6: Precision on TigerBot-13B-Chat

Take-Aways *Prompt-OIRL learns a reward model that is able to accurately evaluate prompting strategies in an offline manner.* The learned reward model is effective in predicting whether a given prompt can solve a given query without access to the language model. The prediction accuracy and precision are in general much better than the self-critic baseline, where online interactions with LLMs are needed.

5.2 Improving Arithmetic Reasoning: How good are the optimized prompts?

Experiment Setup In this section, we use the reward model to perform query-dependent prompting optimization. We compare the following experiment setups: the **Training Best** is a baseline that selects the best prompting strategy out of the training prompts according to their training set performance, hence it is a query-independent baseline; the **Naive Mixture** considers all prompting strategies — including both training prompts and held-out testing prompts — equally, and select prompting strategies with the highest predicted outcome for every query; We consider another two LLM-confidence-based baselines, which use LLMs to reflect how confident they are for different answers generated by different prompting strategies. To be specific, the **LLM Conf. Train** selects the most confident answers that correspond to training prompts according to confidence scores provided by LLMs, and the **LLM Conf. Test** selects the most confident answers across all answers generated by both training and test prompts.

We use **IRL Test 0** to denote the results of not including test prompts in optimization. Such a setting manifests how much can be gained by using a query-dependent prompting strategy in test time when comparing it with the Training Best baseline. In **IRL Test 10** and **IRL Test 100**, different numbers of testing prompts are used for optimization. The way those approaches different from the Naive Mixture is that they prioritize the training prompts, and only select the test prompts when all training prompts are predicted to lead to wrong answers.

Results We present the success rate of getting correct answers on the test queries of different methods on different models and datasets in Figure 7 - Figure 9. In all experiments, we find our query-dependent prompting algorithms outperform the baselines. That said, it can also be observed that for the more challenging tasks like GSM8K, the ability of the LLMs matters: LLaMA-2-7B and TigerBot-13B-Chat performs much worse than GPT-3.5-turbo on such a task, this further leads to an imbalance in the training dataset: we can also observe the precisions on those settings have much larger space for future improvement.

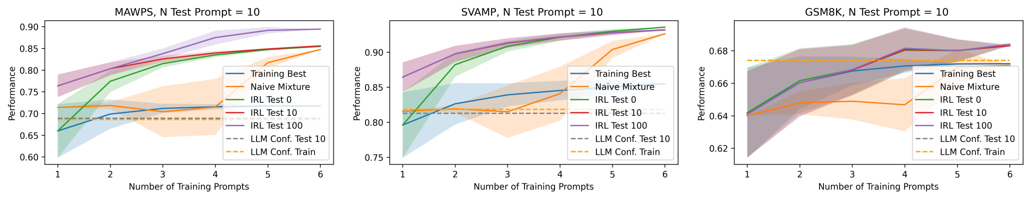

Figure 7: Prompting Success Rate on GPT 3.5 Turbo

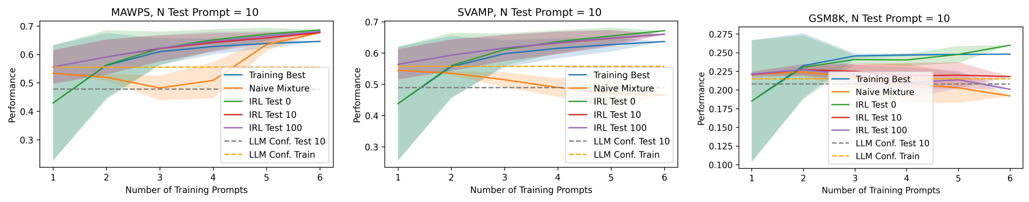

Figure 8: Prompting Success Rate on LLaMA-2-7B-Chat

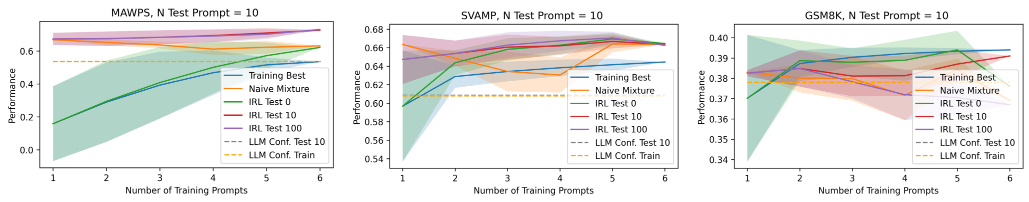

Figure 9: Prompting Success Rate on TigerBot-13B-Chat

Take-Aways Prompt-OIRL is able to improve the arithmetic reasoning ability of various LLMs. Its performance is relevant to the quality of the learned reward function. The performance of Prompt-OIRL increases when the scale of offline data increases.

5.3 How efficient is Prompt-OIRL?

As Prompt-OIRL conducts prompt evaluation in a purely offline approach, there is no additional cost in additional interactions with LLMs during inference time. Consider we have N potential prompts to evaluate and select from in the inference time. For one inference-time query, Prompt-OIRL is able to select the best prompt, hence only the selected prompt will be used in interacting with the LLMs to get the answer. Therefore, the number of LLM calling per inference query is 1, and the number of tokens used usually depends on the type of prompting strategies, varying from a few to hundreds, with an average of 200. In this case, the inference time cost is not increased, and the rate of successfully getting the correct answer increases.

In comparison, evaluating prompts using LLMs by querying their confidence or critical thoughts of different answers requires much more interaction. For every inference-time query, all prompting strategies should be applied first to get different answers, such a step takes N interaction and will use on average $200 \times N$ tokens. With those answers, the LLMs are then asked to check whether those answers are correct and provide confidence scores, which leads to another N interaction and additional usage of $200 \times N$ tokens. In this case, the inference time cost is increased up to $2N$ times, yet the improvement of success rate is limited.

References

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1, 2004.

Christopher G Atkeson and Stefan Schaal. Robot learning from demonstration. In *ICML*, volume 97, pp. 12–20. Citeseer, 1997.

Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pp. 783–792. PMLR, 2019.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, et al. Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4):1–4, 2015.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf`.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

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

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pp. 158–168. PMLR, 2022.

Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. Iq-learn: Inverse soft-q learning for imitation. *Advances in Neural Information Processing Systems*, 34: 4028–4039, 2021.

Ryan Greene, Ted Sanders, Lilian Weng, and Arvind Neelakantan. New and improved embedding model, 2022.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*, 2021.

Yaru Hao, Zewen Chi, Li Dong, and Furu Wei. Optimizing prompts for text-to-image generation. *arXiv preprint arXiv:2212.09611*, 2022.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Dave Hulbert. Tree of knowledge: Tok aka tree of knowledge dataset for large language models llm. `https://github.com/dave1010/tree-of-thought-prompting`, 2023.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474, 2020.

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

Xiaonan Li and Xipeng Qiu. Mot: Pre-thinking and recalling enable chatgpt to self-improve with memory-of-thoughts. *arXiv preprint arXiv:2305.05181*, 2023.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2023.

Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Peter West, Ronan Le Bras, Yejin Choi, and Hannaneh Hajishirzi. Generated knowledge prompting for commonsense reasoning. *arXiv preprint arXiv:2110.08387*, 2021.

Jieyi Long. Large language model guided tree-of-thought. *arXiv preprint arXiv:2305.08291*, 2023.

Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Silvio Savarese, and Li Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *arXiv preprint arXiv:2003.06085*, 2020.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*, 2022.

Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 6292–6299. IEEE, 2018.

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 2, 2000.

R OpenAI. Gpt-4 technical report. *arXiv*, pp. 2303–08774, 2023.

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

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*, 2021.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

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

Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Subhro Roy and Dan Roth. Solving general arithmetic word problems. *arXiv preprint arXiv:1608.01413*, 2016.

Stefan Schaal. Learning from demonstration. *Advances in neural information processing systems*, 9, 1996.

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

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

TigerResearch. Tigerbot: A cutting-edge foundation for your very own llm. `https://github.com/TigerResearch/TigerBot`, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. Spot: Better frozen model adaptation through soft prompt transfer. *arXiv preprint arXiv:2110.07904*, 2021.

Rui Wang, Hongru Wang, Fei Mi, Yi Chen, Ruifeng Xu, and Kam-Fai Wong. Self-critique prompting with large language models for inductive instructions. *arXiv preprint arXiv:2305.13733*, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

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

Yueh-Hua Wu, Nontawat Charoenphakdee, Han Bao, Voot Tangkaratt, and Masashi Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pp. 6818–6827. PMLR, 2019.

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

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5628–5635. IEEE, 2018.

Tianjun Zhang, Xuezhi Wang, Denny Zhou, Dale Schuurmans, and Joseph E Gonzalez. Tempera: Test-time prompt editing via reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2022a.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*, 2022b.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022a.

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

Appendix

A Additional Related Work

Zero-Shot and Few-Shot Prompting The ability of Zero-shot prompting emerges in the language models trained on large amounts of data like GPT-3 and GPT-4 (Ouyang et al., 2022; OpenAI, 2023). And it was shown in Wei et al. (2021) that instruction-fine-tuning improves the zero-shot learning ability of language models.

Notwithstanding the impressive zero-shot performance exhibited by large language models, these models often exhibit suboptimal performance in executing more complex tasks under a zero-shot setting. Leveraging few-shot prompting presents a viable approach for facilitating in-context learning (Brown et al., 2020; Min et al., 2022). This technique necessitates the inclusion of demonstrations within the prompt, effectively guiding the model toward enhanced performance. These demonstrations act as conditioning mechanisms for succeeding examples, leading the model to generate better responses.

Chain-of-Thought Prompting In some more challenging tasks like complex arithmetic, commonsense, and symbolic reasoning tasks, the chain-of-thought (CoT) prompting is shown to be more effective in helping the language models to get correct answers (Wei et al., 2022). CoT includes additional reasoning steps in the few-shot prompting examples. Kojima et al. (2022) further introduces zero-shot CoT, showing that adding task-agnostic instruction can improve the model performance in specific tasks. In Zhang et al. (2022b), Auto-CoT combines the universality of zero-shot CoT and the capability of original CoT driven by demonstrations and proposes to automatically construct demonstrations based on clustering and diversity-based sampling that are beneficial for CoT reasoning.

Other Prompting Strategies Wang et al. (2022) further improves the few-shot CoT method by sampling multiple diverse reasoning paths and marginalizing those paths, choosing the most consistent answers among all sampled reasoning paths.

The Generated Knowledge Prompting Liu et al. (2021) improves commonsense reasoning by incorporating knowledge or information related to the questions to make more accurate predictions. Tree-of-thoughts (ToT) methods (Long, 2023; Yao et al., 2023) combines tree-based planning methods with reasoning skills of language models, and solves hard reasoning problems step by step via multiple round conversations. Hulbert (2023) also put forward a related idea that leverages multiple thoughts of a language model in a single prompt. Memory and Retrieval Augmented Generation (RAG) (Lewis et al., 2020), that is able to combine parametric memory and non-parametric memory like Wikipedia in completing knowledge-intensive tasks. MoT (Li & Qiu, 2023): Pre-thinking based on the external unlabeled dataset and then recalling the related knowledge during inference. In the concurrent work of Yang et al. (2023), LLMs are used as optimizers to solve a variety of optimization problems, including prompt optimization. It is worth noting the most important two differences are in the offline nature and the query-dependant prompting strategy of our method.