LLM-Informed Discrete Prompt Optimization

Zeeshan Memon [*1] **Muhammad Arham** [*1] **Adnan Ul-Hasan** [2] **Faisal Shafait** [1 2]

Abstract

The advent of Large Language Models (LLMs) has significantly improved NLP tasks, but their performance depends on effective prompt engineering, where engineers iteratively craft prompts by observing the dynamics of LLMs. With the rising number of LLMs, each trained on different data sources and thus exhibiting different internal sensitivities, prompt engineering has become an increasingly cumbersome task. The solution to these challenges lies in an automated and reliable model capable of suggesting optimized prompts and adapting to various LLMs. Previous works have primarily focused on training learnable vectors or identifying discrete prompts, which were effective for earlier, smaller language models. However, contemporary LLMs require coherent text prompts tailored to their specific training instructions. In this paper, we address this gap by proposing a methodology for training a lightweight model that not only produces legible, optimized prompts but also adapts to different LLMs. The proposed methodology has demonstrated significant performance improvements with optimized prompts across different LLMs.

1. Introduction

Prompting has emerged as a pivotal methodology in leveraging the capabilities of LLMs to tackle a diverse array of Natural Language Processing (NLP) tasks. With the advent of powerful pre-trained language models such as GPTs (Radford et al., 2019), BERT and Roberta (Liu et al., 2019), prompting has gained significant attention as an alternative to traditional fine-tuning approaches for specific tasks. Unlike conventional fine-tuning methods that necessitate costly updates to the extensive parameters of the language model for each downstream task, prompting involves appending an additional piece of text to the inputs, guiding the LLM to generate desired outputs. The primary research interest surrounding prompting pertains to the identification of optimal prompts that enhance the performance of LLMs across a spectrum of tasks, often in scenarios with limited training examples. This paper studies optimizing such prompts, shedding light on their efficacy and impact on LLM performance.

Recent studies have tackled this challenge through various approaches, including the training of auxiliary models or differentiable representations of prompts (Qin & Eisner, 2021; Deng et al., 2022). However, these endeavors often presume access to internal state variables of the LLM (Shin et al., 2020; Lester et al., 2021), a condition not typically available to practitioners who interact with LLMs through APIs. Moreover, the soft prompts obtained through these methodologies are less interpretable in natural language (Shi et al., 2022), and natural language mapping of these soft prompts can be highly misleading as they lose the effectiveness when decoded to natural language(Bailey et al., 2023). Alternatively, some research employs discrete manipulations of prompts using Reinforcement Learning techniques or LLM-based feedback mechanisms (Zhang et al., 2022; Zhou et al., 2022). Nevertheless, these algorithms may necessitate low-level access to the LLM, generate outputs that are difficult to comprehend or rely on undirected Monte-Carlo search strategies across the semantic space of prompts.

Powerful models such as GPT-3.5 exhibit a reluctance to process nonsensical tokens and prioritize legible text, owing to their fine-tuning on instruction-based tasks (Cherepanova & Zou, 2024). This trait underscores the importance of crafting coherent and legible prompts for effectively prompting these models. Previous prompt optimization techniques mainly focussed on optimizing learnable embeddings and then finding the nearest word from vocabulary. These methods worked well for previous GPTs till GPT-2 but failed with contemporary instruction-tuned LLMs.

In this paper, we propose an approach aimed at fine-tuning

[*]Equal contribution [1]School of Electrical Engineering and Computer Science, National University of Sciences and Technology (NUST), Islamabad, Pakistan [2]Deep Learning Laboratory, National Center of Artificial Intelligence (NCAI), Islamabad, Pakistan. Correspondence to: Zeeshan Memon <zeeshan.bese20seecs@seecs.edu.pk>, Muhammad Arham <marham.bese20seecs@seecs.edu.pk>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

a lightweight language prompter model, using a dataset comprising prompts and optimized prompt data. Moreover, to ensure the adaptability of this prompter model across various LLMs, each of which is trained distinctively and exhibits sensitivity to different vocabulary tokens for the same task, we conduct fine-tuning of the language model head, accompanied by the addition of a linear layer tailored to accommodate the specifics of the target LLM. Subsequently, we evaluate the performance of the prompter model across multiple LLMs and observe its performance gain over user-generated prompts by a substantial margin across diverse domains.

2. Previous Work

The exploration of prompt tuning to condition LLMs for specific tasks can be divided into two principal streams of research: soft-prompt tuning and discrete (hard) prompts.

In the domain of soft-prompt tuning, as described in studies (Qin & Eisner, 2021; Lester et al., 2021), trainable vectors, also termed virtual tokens, are appended to the input layer to enable efficient task-specific tuning of LLMs. These virtual tokens are adapted to individual tasks and integrated with the standard user prompts during the inference stage. Although such methods have been effective in enhancing the performance of LLMs, their applicability is restricted by a lack of generalizability across various language models due to differences in their latent representations. Moreover, the opacity of these virtual tokens has catalyzed further research into discrete prompt optimization, which seeks to provide greater interpretability for model-tuning processes.

On the other hand, hard-prompt methods automate the process of manual prompt engineering by incorporating intelligible word tokens into the original prompt structure. For instance, AutoPrompt leverages a template-based approach where a specific number of trigger tokens are optimized via gradient descent for tasks such as sentiment classification (Shin et al., 2020). This strategy highlights the potential of prompt tuning as a viable alternative to extensive model finetuning, particularly with masked language models. Recent studies include methodologies like RLPrompt, which utilizes a smaller language model with a trainable head to refine user prompts into a specific set of discrete tokens (Deng et al., 2022). This strategy has shown promise in tasks such as few-shot classification and style-transfer tasks for models like DistilGPT and GPT-2. Nevertheless, it has been observed that the discrete tokens generated can often be nonsensical and difficult to interpret when applied to larger language models, diminishing their efficacy in newer instruction-tuned and dialogue-based LLMs.

Our methodology is influenced by the approach used in Promptist, which optimizes a small language model to generate discrete prompts specifically tailored for a Stable Diffusion model, leading to better aesthetic scores while maintaining relevance to the prompt (Hao et al., 2024). Extending this approach to generative language models, we apply parameter-efficient fine-tuning to a smaller language model, enhancing evaluation scores consistently across a varied set of language models. This strategy not only addresses the shortcomings of generalizability seen in soft-prompt methods but also enhances the transparency and efficacy of the prompts, thereby broadening the practical deployment of LLMs in specialized tasks.

3. Proposed Methodology

In this study, we propose a two-step fine-tuning process, as depicted in Figure 1. Each of these stages is discussed in the following subsections.

3.1. Stage 1 - Supervised Finetuning of Lightweight Language Model

The main goal of supervised fine-tuning for the lightweight model is to create clear and understandable text that includes important tokens for the specific LLM, while still keeping the text easy to read and in context. Unlike dealing with random or nonsensical text, modern LLMs trained on instructional data need input that makes sense. To do this, a lightweight language model that specializes in suggesting optimized prompts is improved through supervised fine-tuning. This process involves using a dataset containing prompts and their optimized versions, with the details of how this dataset is created discussed in Section 4.

3.2. Stage 2 - Language Head Fine Tuning

Stage 1 ensures the training of the language model for a specific task, i.e., to produce optimized prompts given the dataset pattern. However, it does not ensure whether the optimized prompt contains LLM-specific sensitive tokens. It makes the prompt more definite and detailed, but the performance across LLMs varies significantly as it is only trained on a specific dataset. During language modeling, the LM head has the same input dimensions, but the output dimensions are the same size as the vocabulary, providing a probability for each token's fit in a given position. Thus, the language head is responsible for modeling probabilities over the vocabulary.

Taking this as fundamental for prompter model adaptation across LLMs, we add a single MLP layer and make this layer and the language head trainable in stage 2. MLP layer acts as a parameter efficient method to compute the adapted embeddings, inspired by the work mentioned in [4].

Let $\mathbf{E} \in \mathbb{R}^{d \times n}$ represent the encoding matrix produced by the language model, where d is the dimension of the

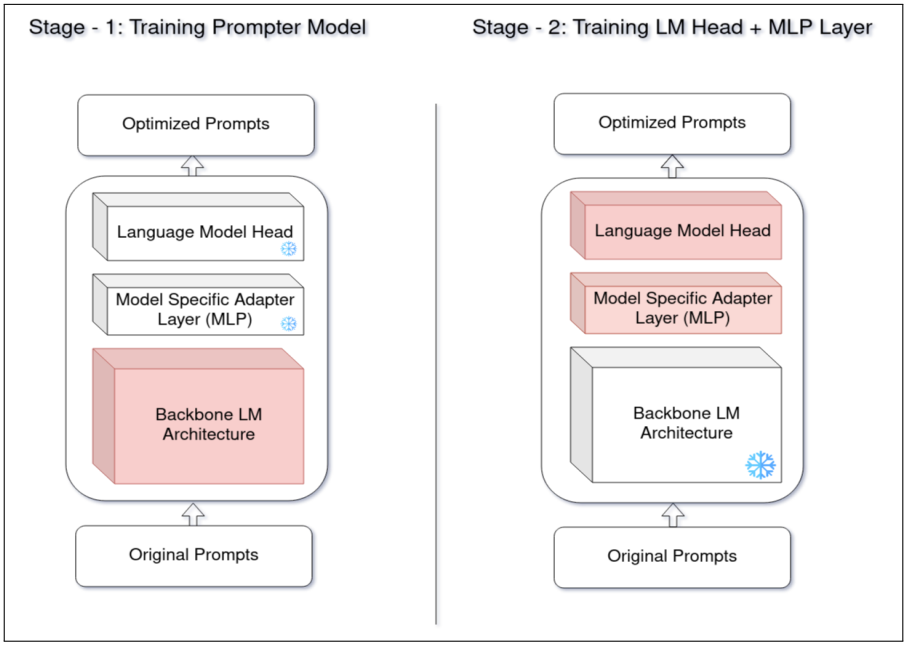

Figure 1. Proposed approach for training prompter model. In the first stage, a lightweight language model undergoes supervised fine-tuning to generate clear and contextually relevant prompts. In the second stage, only the language head and a single Multi-Layer Perceptron (MLP) layer are trained, allowing adaptability across different language models. This stage ensures that the optimized prompts contain model-specific tokens

encoded representation and n is the number of tokens.

$$\mathbf{E} = \text{Encoder}(\mathbf{X})$$

Here, $\mathbf{X}$ denotes the input tokens and Encoder$(\cdot)$ represents the fixed trained language model up to the encoding layer in Stage 1.

The MLP layer, which transforms the encoded representations, is defined as:

$$\mathbf{Z} = \text{MLP}(\mathbf{E})$$

where $\mathbf{Z} \in \mathbb{R}^{d \times n}$ is the transformed representation.

The language head, which maps the transformed representations to the vocabulary space, is defined as:

$$\mathbf{Y} = \mathbf{LM}_{\text{head}}(\mathbf{Z})$$

where $\mathbf{Y} \in \mathbb{R}^{v \times n}$ is the output logits, v is the size of the vocabulary, $\mathbf{LM}_{\text{head}} \in \mathbb{R}^{d \times v}$ is language head.

For each LLM, the MLP and Language Head combination is different and dynamically loaded during inference conditioned on the LLM for which the prompter module is used. By training the MLP + Language Head separately for each LLM, our language head functions in an adaptable fashion, as it is responsible for assigning probabilities across LLMs. By training this LM Head, it alternates between high probability tokens across LLMs, indicating that different LLMs are sensitive to different tokens. In this stage, a dataset of prompts and optimized prompts for adaptation, as discussed in Section 4, is used to train only the MLP and language head.

4. EXPERIMENT CONFIGURATION

4.1. Datasets Preparation

To fine-tune the language model, we used the ChatGPT-3.5-Turbo API to create a synthetic dataset of 1,000 pairs of general and optimized prompts, covering topics like academic writing, text-style transformation, and code generation.In the next stage, we generated data using the target language model for which the adapter is to be trained to ensure the synthetic data matched its distribution, including relevant tokens and keywords.

4.2. Configurations for LLM Fine-tuning

Our prompter model is based on the Microsoft Phi-2 language model. In the initial stage, we use parameter-efficient fine-tuning with 32 rank LoRA layers across key transformer components. The model is trained for ten epochs on a 24GB Nvidia RTX4090 GPU. In the second stage, we add a linear layer before the language model head, initialized as an identity matrix with zero bias. This configuration maintains the original data distribution while adding trainable parameters for the adapter head. The new linear layer is

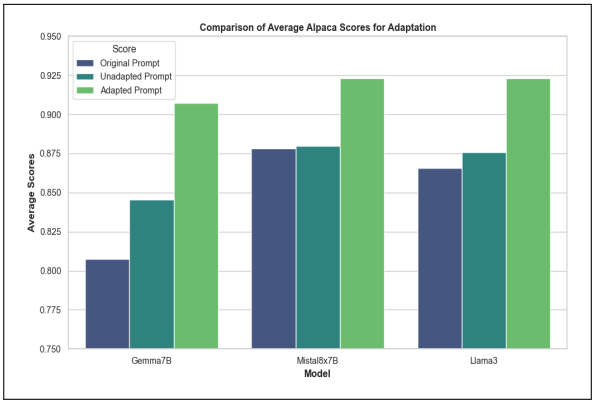

Figure 2. AlpacaEval Scores - Adapted Optimized Prompts vs. Non-Adapted Optimized Prompts on GPTeacher Dataset (Teknium, 2024)

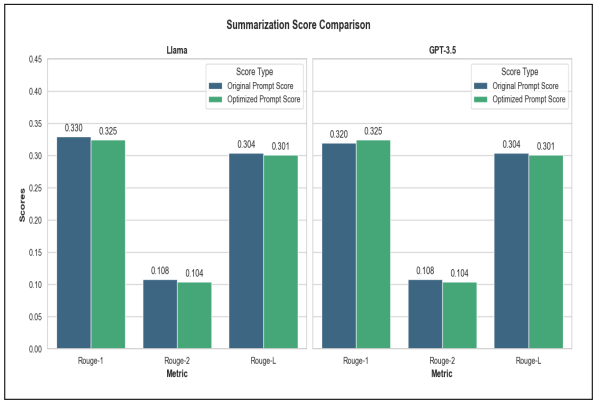

Figure 3. Rouge Score Comparison for Summarization Task for user and optimized prompts on CNN and Daily Mail datasets.

trained concurrently with the language head, optimizing latent representation mapping to the vocabulary.

5. Results

Two primary metrics are used to evaluate LLMs: the Massive Multi-task Language Understanding (MMLU) score and the AlpacaEval score [12]. The MMLU includes around 16,000 multiple-choice questions across 57 academic disciplines, serving as a benchmark for assessing knowledge retention and contextual accuracy in LLMs. The AlpacaEval score is an LLM-based automatic evaluation, validated against 20K human annotations and demonstrating a 0.98 correlation with human judgment (Li et al., 2023). It leverages strong LLMs like GPT-4 for automatic evaluation of responses. In this study, only the AlpacaEval score is utilized for evaluation, as prompting cannot enhance the inherent knowledge of LLMs. Our two-step methodology was evaluated to assess the impact of training stages and adaptation on language model performance. AlpacaEval scores were computed for optimized prompts from the baseline prompter model and the adapted model post-Stage 2 training. Results (Figure 2) demonstrate that adapted prompts significantly outperformed baseline prompts, emphasizing the effectiveness of training the language head and MLP for adaptation.

Table 1. AlpacaEval Scores for User and Optimized prompts across various model sizes

MODEL	USER PROMPT	OPTIMIZED PROMPT
GEMMA7B	0.807	0.914
VICUNA7B	0.843	0.885
LLAMA3-8B	0.865	0.921
QWEN1.5-14B	0.879	0.918
MISTRAL8X7B	0.878	0.923
CHATGPT3.5	0.865	0.900

Expanding on our findings, we further assessed our method's performance using the AlpacaEval score. Across six LLMs, significant improvements were observed for optimized prompts compared to user-generated ones, with LLM sizes ranging from 7 billion to 175 billion parameters. Particularly, smaller models like Gemma7B and Llama3-8B exhibited more noticeable improvements, revealing greater sensitivity and variation in these models. Notably, optimized prompts in smaller models achieved performance comparable to larger ones like Mistral8x7B and GPT-3.5, highlighting the impact of prompt optimization in enabling lightweight models to rival larger counterparts.

Similarly, the methodology's performance over closed-ended tasks, such as summarization, has also been examined. As illustrated in figure 3, both prompts and optimized prompts yield similar results for closed-ended tasks like summarization highlighted by Rouge-1, Rouge-2 and Rouge-L scores. This suggests a limitation of prompt optimization: it works best for tasks that require generative creative capabilities or have an open-ended nature. Additionally, the effectiveness of prompt optimization may vary slightly across Language Model Models (LLMs), as indicated by the slight improvement in the summarization task performance with optimized prompts for Llama 3 compared to GPT-3.

6. Limitations

Despite the promising results of our research, there are two main limitations to our proposed methodology.

6.1. Limited Augmentation of Inherent Knowledge

While our method of prompt optimization enhances the generative capabilities of Large Language Models (LLMs), it does not augment the inherent knowledge contained within

these models. Although the methodology excels in open-ended tasks, its effectiveness is not prominent in closed-ended tasks such as summarization. For the same reason, only the AlpacaEval Score is used to evaluate instead of MMLU, which assesses the inherent knowledge of LLMs.

6.2. LLM-Specific Training Requirements

For each new LLM, it is necessary to train a different language model head as guided in our training protocol. This makes the optimized prompt specific to that particular LLM. Our supervised learning approach is dependent on the training of this language model head, resulting in a lack of automatic adaptation. This limitation could be mitigated by incorporating reinforcement learning into the methodology to enable more dynamic and adaptive prompt optimization.

7. Conclusion

This study underscores the significance of prompt tuning in enhancing the performance of LLMs without necessitating explicit fine-tuning of their parameters. Considering constraints in accessing model details and computational resources, our research addresses the need for robust prompt tuning methods applicable to both closed-source and open-source LLMs. The suggested training methodology of the lightweight model demonstrates the efficacy of the model in crafting coherent and comprehensible text prompts, compatible with contemporary language models. Evaluation conducted through AlpacaEval, utilizing a potent language model such as GPT-4 as a benchmark, validates our approach's capability. Our findings consistently reveal enhanced performance across LLMs of varying sizes, ranging from models with 7 billion to over 100 billion parameters. In future research, we suggest implementing an agentic framework with Reinforcement Learning to enhance prompt optimization and adaptability for domain and task-specific prompting. This approach opens avenues for deeper exploration and advancement within the field.

References

Bailey, L., Ahdritz, G., Kleiman, A., Swaroop, S., Doshi-Velez, F., and Pan, W. Soft prompting might be a bug, not a feature. 2023.

Cherepanova, V. and Zou, J. Talking nonsense: Probing large language models' understanding of adversarial gibberish inputs. *arXiv preprint arXiv:2404.17120*, 2024.

Deng, M., Wang, J., Hsieh, C.-P., Wang, Y., Guo, H., Shu, T., Song, M., Xing, E. P., and Hu, Z. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint arXiv:2205.12548*, 2022.

Hao, Y., Chi, Z., Dong, L., and Wei, F. Optimizing prompts for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2024.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Li, X., Zhang, T., Dubois, Y., Taori, R., Gulrajani, I., Guestrin, C., Liang, P., and Hashimoto, T. B. Alpacaeval: An automatic evaluator of instruction-following models, 2023.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

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

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Shi, W., Han, X., Gonen, H., Holtzman, A., Tsvetkov, Y., and Zettlemoyer, L. Toward human readable prompt tuning: Kubrick's the shining is a good movie, and a good prompt too? *arXiv preprint arXiv:2212.10539*, 2022.

Shin, T., Razeghi, Y., Logan IV, R. L., Wallace, E., and Singh, S. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

Teknium. Gpteacher-general-instruct. https://huggingface.co/datasets/teknium/GPTeacher-General-Instruct, 2024. Accessed: 2024-05-30.

Zhang, T., Wang, X., Zhou, D., Schuurmans, D., and Gonzalez, J. E. Tempera: Test-time prompting via reinforcement learning. *arXiv preprint arXiv:2211.11890*, 2022.

Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., and Ba, J. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022.

Appendix

User and Optimized Prompts with AlpacaEval Scores

In this section, we compare the user-provided prompts and the optimized prompts generated using our proposed method. The quality of each prompt is evaluated using the AlpacaEval scoring metric to highlight the efficacy of our optimization approach. As illustrated in the table, the optimized prompts generally outperform the user-provided prompts in terms of

Table 2. User vs Optimized Prompts - Italicized AlpacaEval scores denote comparative performance, with green highlighting better AlpacaEval Score and red indicating lesser performance score.

User Prompt	Optimized Prompt
What are some highly-rated movies on Netflix right now? *(0.8)*	Locate some highly-rated movies playing on Netflix today, analyzing reviews and audience feedback. *(0.8)*
Can you explain Einstein's theory of relativity in simple terms? *(0.9)*	Simplify Einstein's theory of relativity for easy comprehension. *(0.97)*
What are some unique themes for a photography project? *(0.77)*	Explore unique themes for a photography project to showcase creativity and innovation. *(0.83)*
Write a short story about a detective solving a mystery in a small town. *(0.71)*	Craft an engaging short story centered around a detective investigating a perplexing mystery in a tight-knit small town community. *(0.83)*
Write a Python script to scrape data from a website. *(0.88)*	Write a Python script to scrape data from a website, extracting relevant information and organizing it in a structured format. *(0.91)*
How does the human digestive system work? *(0.91)*	Investigate the processes involved in the digestion of food by the human body. *(0.93)*
Create a simple to-do list application using React.js. *(0.87)*	Develop a user-friendly to-do list application using React.js, allowing users to add, update, and delete tasks. *(0.86)*
How do I write an abstract for a research paper on climate change? *(0.85)*	Discover tips for crafting an effective abstract for a research paper on climate change, including key sections and formatting guidelines. *(0.92)*
Find me a reliable source on the effects of diet on mental health. *0.7*	Explore reliable sources investigating the relationship between diet and mental health, including dietary patterns, nutritional deficiencies, and mental health disorders. *0.82*

AlpacaEval scores, demonstrating the effectiveness of our optimization method. However, it is important to note that in some cases, the user prompts achieve better scores than the optimized prompts. These instances typically involve well-defined and closed-ended tasks, where the initial prompts are already highly effective.

Conversely, in open-ended tasks, our optimized prompts show significant improvements over the user prompts. This indicates that our method is particularly beneficial for tasks that require creative and comprehensive responses, thereby enhancing the overall performance and versatility of Large Language Models in diverse applications.