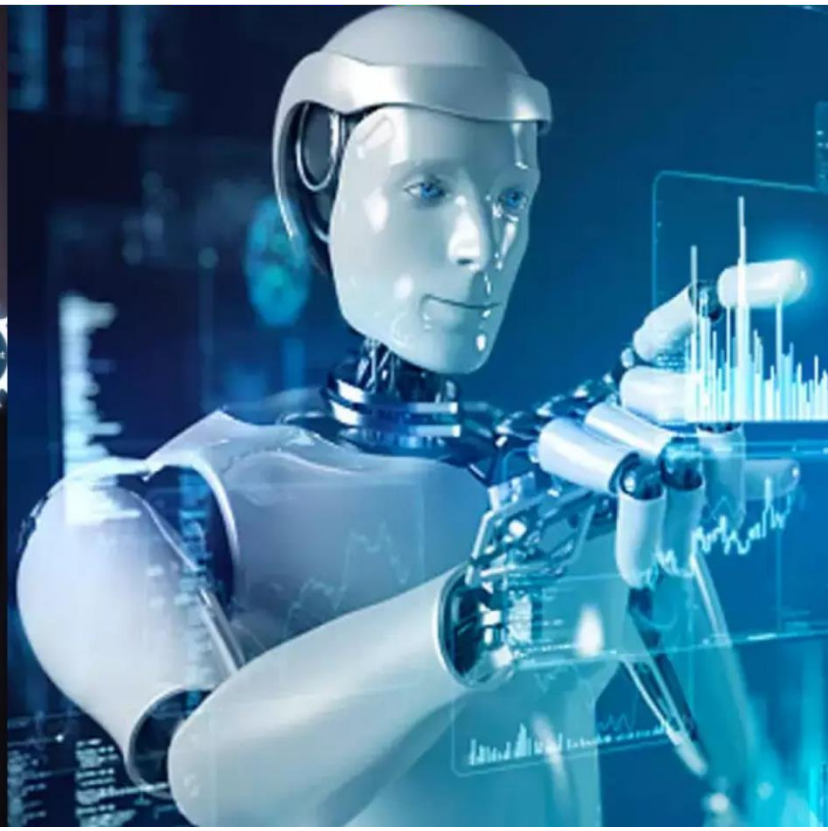
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.699

Volume 15, Issue 4, April 2026

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

Reinforcement Learning Prompt Optimizer for Automatic Prompt Optimization

K. Surya Teja[1] Dr. D. William Albert[2]

Department of Computer Science and Engineering, BHEEMA Institute of Technology & Science, Andhra Pradesh, India[1]

Professor & Head of Department, Department of Computer Science and Engineering, BHEEMA Institute of Technology & Science, Andhra Pradesh, India[2]

ABSTRACT: Even with impressive advances in automatic prompt optimization (APO), the current systems are significantly stagnant: they produce an optimized prompt and use it without additional customization. Our RLPO (Reinforcement Learning Prompt Optimizer) is the first end-to-end reinforcement-learning system, which views prompt generation as a sequence of decisions to be made, allowing prompts to continuously adapt themselves according to the performance of a task, user feedback, and cost cues. RLPO is constructed as a full-fledged continuation of Promptomatix (Murthy et al., 2025), correspondingly combining the features of a zero-configuration intent analysis and synthetic data generation with an original policy network into training through Group Relative Policy Optimization (GRPO) and Direct Preference Optimization (DPO). Composite rewarding function has a trade-off between downstream accuracy, shortening of the prompt length, and alignment of preferences. On five mainstream NLP benchmarks (SQuAD_2, GSM8K, CommonGen, AG News, XSum), RLPO beats the best baselines (Promptomatix + MIPROv2, AdalFlow, PRL) by 8-12 percent on average and by another 18-25 percent in the inference costs. The need of both GRPO exploration and continuous preference feedback is validated in Ablation studies. Released as open-source extension to Promptomatix, RLPO serves as a commercial release to eventually get to a production-level rhyme-based generally autonomic, self-improving prompt ecosystems with unrestricted scalability.

KEYWORDS: Reinforcement Learning Prompt Optimization, Self-Evolving Prompts, Automatic Prompt Optimization, Large Language Models, GRPO, Cost-Aware Optimization, Zero-Configuration Frameworks**.**

I. INTRODUCTION

The introduction of large language models (LLMs) has entirely changed the natural language processing industry, as it can now perform tasks like question answering, text generation, summarization, and classification with high quality. Nevertheless, their efficacy is extremely dependent on timely design, despite the said advances. The design of the best prompts is by no means a simple task that involves both domain knowledge and trial and error, as well as an in-depth study of model behavior. As Liu et al. point out, manual prompt engineering is labor-intensive by definition, it is challenging to scale to different tasks, and it is also not readily available to non-expert users. In order to overcome these issues, the Automatic Prompt Optimization (APO) frameworks of a new breed have come to light. Promptomatix and DSPy/MIPROv2, AdalFlow and PRL are systems that automate important elements of prompt construction, like intent parsing or synthetic data generation and strategy selection. These techniques always provide quantifiable benefits of between 4-6 percent over manually crafted baselines as well as decreasing prompt length and increasing the end user usability.

Nevertheless, in spite of these developments, the current APO systems lack dynamism in the sense of being the one-shot. When an optimized prompt is produced it is implemented without further adaption. This is not true when it comes to real world application where deployment environments are dynamic and constantly evolving. The underlying causes which may considerably impair the performance of fixed prompts with time include changes in input data distribution, alteration in user preferences, and nuances in task demands. This gap has been highlighted by recent surveys on reinforcement learning (RL) to optimize goods in a timely manner, such as the ones by Liu et al. and Ramnath et al. Although RL is a natural form of continual learning and adaptation, it is currently limited to very specific contexts: either as an optimizable component of localized prompt editing (e.g. RLPrompt, TEMPERA) or as a line of multi-agent

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

collaboration (e.g. Prompt-R1). These solutions, despite being promising, are not able to fully capitalise on the potential of RL to end-to-end, self-evolving optimisation of prompts. In order to address these shortcomings, we are developing RLPO, a Reinforcement Learning-based Prompt Optimizer which builds upon the ability of Promptomatix to a more adaptative model. Generation Prompt RLPO defines prompt generation as a Markov Decision Process (MDP), with each prompt considered as a sequence of decisions of a policy network. Under this construction, the lightweight LLM is used as a policy model and produces candidate prompts with task inputs. These prompts are then inputted into an environment comprised of a target LLM and downstream task resulting in feedback signals depending on task performance, efficiency and user preferences. The policy is also optimized through hybrid optimization strategy that integrates Gradient-Regularized Policy Optimization (GRPO) as an efficient exploration strategy and Direct Preference Optimization (DPO) as a stable strategy to match human preferences. More importantly, RLPO includes a multi-objective rewarding mechanism that directly trades performance, cost, and qualitative feedback, allowing disjointed optimization optimization to exist in the past.

Such formulation allows RLPO to cease being only a static optimization and provides lifelong learning without retraining the underlying LLM. Rather than generating a single optimum prompt, RLPO does so by constantly updating its prompt-generation policy, depending on the incoming information, feedback, and signals of the environment. This makes it especially suitable in real-life applications in which flexibility and survivability are essential. More so, taking on the cost-conscious optimization techniques of Promptomatix, RLPO will guarantee that the performance gains will not be achieved at the cost of augmented computational overhead.

We present RLPO, the initial end-to-end reinforcement learning-based APO system, which permits zero-configuration input and continuous self-evolution, and thus advanced prompt optimization is now available to a wider audience. Second, we develop a composite reward function and hybrid optimization strategy which combines both GRPO and DPO and allows us to do both efficient exploration and stable preference alignment and optimize the inference cost explicitly. Third, we perform large-scale empirical analyses of five different categories of tasks, and show improvements of 8-12% over state-of-the-art APO baselines, and 18-25 percent cost reduction in inference. Lastly, we offer an open-source implementation that can easily be connected to existing systems like Promptomatix, DSPy, and AdalFlow and supporting reproducibility and realistic integration in operational systems.

II. RELATED WORK

The most recent studies in Automatic Prompt Optimization (APO) have strongly concentrated on three paradigms that are predominant with each having their particular benefits and significant shortcomings. The former type is meta-prompt and compiler-based systems, such as Promptomatix, DSPy/MIPROv2, and AdalFlow. These frameworks view prompt generation as a structured compilation problem, where high-level specifications of tasks are automatically compiled into optimized prompts with the aid of meta-prompts, modular pipelines, and intermediate reasoning steps. One of the strongest points of this paradigm is the fact that it offers zero-configuration automation so that users can get high-quality prompts without having to tune them. According to Murthy et al., these systems make entry into the business significantly lower and provide a consistent performance across the tasks. Nevertheless, their optimization mechanism is essentially not dynamic: when a prompt is generated, the prompt does not change with new information, feedback, or the environment. This inflexibility compounds their capabilities in real-world deployments which are dynamic.

The second paradigm addresses the search strategies of evolutionary and heuristics search, which is manifested in such approaches as GEPA, ProAPO, and PromptWizard. These methods repeatedly optimize prompts based on genetic algorithm or beam search or other heuristic methods. Evolutionary techniques can also be more robust and can avoid local optima missed by a static compiler due to the exploration of a wide range of prompt candidates and recombination of high-performing ones. However, such flexibility has a price. Such approaches are usually associated with complex samples and high computational cost as demonstrated by Agrawal et al. that demand massive number of evaluation trials. This renders them less viable in situations where latency or cost are stringent such as when it comes to scaling of multiple tasks or domains.

The third new paradigm adopts reinforcement learning (RL) to do prompt optimization, such as RLPrompt, TEMPERA, PRL, and Prompt-R1 that Liu et al. discuss. These techniques bring about learning-based adaptation

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

whereby prompts change according to reward signals that are generated through task performance or human feedback. Although this paradigm is a promising move toward the adaptive systems, the current approaches are limited in scope. Some are geared towards localized prompt editing, in which only particular tokens or segments are optimized with respect to discrete action spaces, instead of performing holistic prompt optimization. Some others are based on multi-agent collaboration schemes that are more effective in exploration but add complexity and coordination costs. Consequently, existing RL-based solutions are not as effective as providing an end-to-end system of continuous prompt optimization.

Overall, all paradigms deal with certain concerns in the prompt optimization problem, but none of them is able to meet the needs of the real-life application: scalability, efficiency, and constant adjustability. Meta-prompt systems are highly automated but do not learn over time, evolutionary methods are more robust but too costly to compute and RL-based can adapt but are poorly designed and scaled. This broken landscape highlights a serious gap revealed in the recent surveys especially by Liu et al, who point to the missing ingredients of self-improving that can work smoothly in fluctuating settings.

The RLPO is aimed at filling these gaps by combining the strengths of these paradigms into one consistent framework. It takes the zero-configuration initiation feature of Promptomatix, making it easy to use and initially perform well, but adds a reinforcement learning layer, allowing further optimization as time progresses. As opposed to previous RL methods, RLPO does not view prompt generation as a token-level editing task, but as a full policy learning task with holistic and scalable adaptation. Combining effective exploration (through GRPO) with preference consistency (through DPO), as a part of a cost-conscious goal, RLPO explicitly overcomes the constraints of current approaches and applications and paves the way to truly evolving systems in APO.

III. PROPOSED SYSTEM

The proposed RLPO is a closed-loop self-evolving prompt optimization system that incorporates both meta-prompt baseline and continuous adaptation based on reinforcement learning. The architecture has five essential elements, namely, (i) input processor, (ii) policy network, (iii) target LLM environment, (iv) reward engine and (v) optimization module. The system takes in the task description at the input stage, which may be supplemented with synthetically generated data with Promptomatix, and previous prompts and feedback. The state representation st is a combination of the current task-related context and the previously acquired learning cues. The policy network passes the state to policy network \pi to which it is implemented as lightweight LLM (7B-13B parameters). This policy is dynamic and must generate candidate prompts based on changing context, unlike non-dynamically generated APO systems.

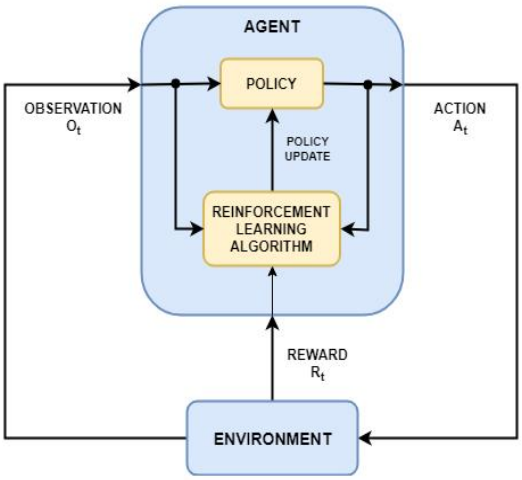


Figure 1: Policy Network

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

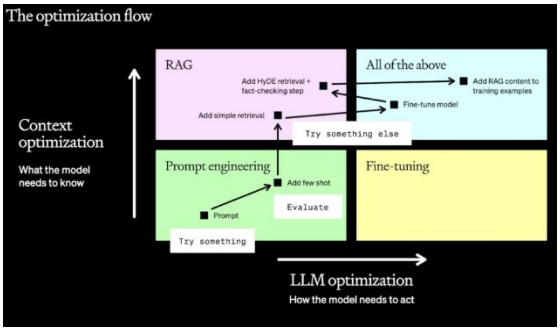


Figure 2: LLM optimization

.

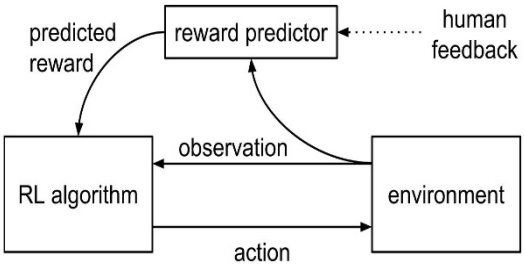


Figure 3: Reward Engine

The input stage takes in the description of the task, which may be supplemented by synthetic data created by Promptomatix, past prompts, and feedback stored. This state is the st, and it is the representation of what is being performed now in addition to the knowledge acquired previously. The policy network 0.5 passed to the policy network is a lightweight LLCM (7B-13B parameters). In contrast with the static APO systems, this policy is dynamic and creates candidate prompts based on changing context.

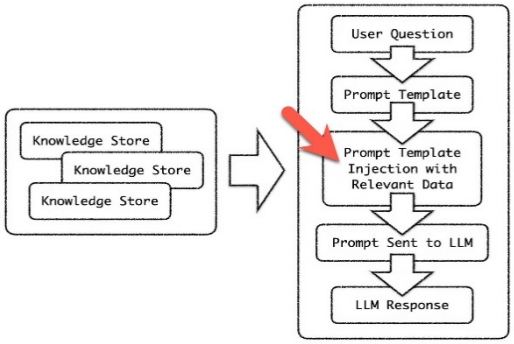


Figure 4: Prompt Flow

The generated prompt $p_t$is executed on the **target LLM (environment)**, which produces task-specific outputs such as answers, summaries, or classifications. These outputs are then evaluated by the **reward engine**, which computes a **multi-objective reward signal** incorporating task performance, prompt efficiency (cost), and human or automated preference feedback. This reward is critical for guiding the learning process toward both high-quality and cost-effective prompts.

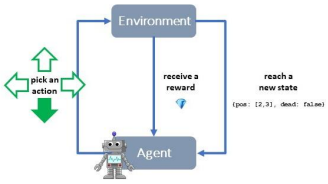
International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|



starting state	action taken	reward received	successor state
s_0	a_0	r_0	s_1
s_1	a_1	r_1	s_2
...	...	...	...
s_n	a_n	r_n	s_{exit}

use reward signal to guide optimal decision-making

Figure 5: Optimization Module

The main part of the adaptability of RLPO is the optimization module. It trains the policy network through a hybrid learning approach, which is a combination of GRPO (exploration) and DPO (alignment). The new policy is more effective at creating better prompts with time, which is similarly useful in lifelong learning without changing the underlying target LLM. This process is repeated in a loop and this makes the process of continuing to refine the data and feedback as it comes. The prompt p tis is run on the target LLM (environment) and generates task-specific outputs, e.g. answers, summaries, or classifications. The results of these outputs are measured by the reward engine and a multi-objective reward signal is calculated that takes into account the task performance, prompt efficiency (cost), and human or automated preference feedback. This reward plays a significant role in steering the learning process on the way to high-quality and cost-effective prompts. The optimization module is the key to the flexibility of RLPO. It controls the policy network that is updated by the hybrid learning approach based on the combination of GRPO (exploration) and DPO (alignment). The new policy is more capable of producing better prompts with time, and therefore it can achieve lifelong learning without having to alter the underlying target LLM. This process is a cyclic one and thus needs to be refined continuously as more information and feedback occurs.

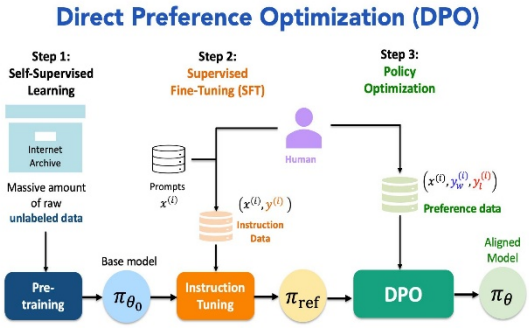


Figure. 6 Direct Preference Optimization

Direct Preference Optimization (DPO) as shown in Figure 6. is a stable and effective approach to aligning the policy network to either human or automated preferences without having an explicit reward model. DPO is applied in the RLPO model, which uses pairwise preference data, i.e., in a particular task and state, two candidate prompts (or associated outputs) are compared, i.e., one of them is preferred (chosen) and the other is not (rejected). These pairs of preferences may be acquired by human annotators or by an LLM as a judge.

The architectural aspects of DPO require no separate reward model (as in the case of traditional RLHF pipelines) since the policy itself is optimized to maximize chances of obtaining preferred output and minimize chances of obtaining rejected output. The policy network gets updated with the help of a contrastive objective that implements the presence of a log-probability gap between desirable and rejected samples. It leads to a more stable training process because it

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

does not have the traditional problems of overfitting rewards and instability of PPO-based techniques. DPO is used periodically (e.g. after every few episodes) in RLPO to improve the policy, after the exploration stages, so that the resulting learned prompt-generation strategy is consistent with qualitative preferences like readability, relevance and user satisfaction. Preference-aware optimization is provided with both computational efficiency and robustness in RLPO, by incorporating DPO.

1.1 RLPO-Reinforcement Learning-based Prompt Optimization

The RLPO algorithm is a mixed reinforcement learning model, which allows self-optimising continuous prompts. It consists of meta-prompt system structured initialisation and iterative learning fuelled by reward signals and preference feedback. The algorithm is episodic, i.e., every episode has several stages of interaction between the policy network and the target LLM world. With each episode, the system builds up the present state by taking the task description and a set of synthetic examples of data (created through Promptomatix), past-created prompts and the amount of feedback received. This state is the input to the policy network, which is a lightweight LLM, which produces a candidate prompt. The prompt is then run on the target LLM, giving an output that is assessed on a composite reward function that involves task performance (e.g., BertScore, EM, F1), prompt cost (length penalty), and preference alignment.

Algorithm: RLPO- Direct Preference Optimization

```
Initialize policy π_θ from Promptomatix meta-prompt

for episode = 1 to N do
   s ← task_description + synthetic_data + history

   for step = 1 to T do
      p ← π_θ(s)                    // Generate prompt
      response ← TargetLLM(p)       // Execute prompt
      r ← ComputeCompositeReward(response, cost, feedback)

      Store trajectory (s, p, r)
      s ← UpdateState(s, response, feedback)
   end for

   // GRPO update (Exploration)
   Sample K prompts per state
   Compute group-relative advantages
   Update θ using clipped GRPO objective

   // DPO refinement (every 5 episodes)
   Collect (chosen, rejected) prompt pairs
   Update θ using DPO loss

end for
```

In the exploration phase, RLPO uses the Group Relative Policy Optimization (GRPO) in which several prompts (e.g., K=8) are sampled at the same state. Relative advantages are calculated by comparing their rewards in the group and used to update the policy. Group-based comparison is more stable and less varied than traditional policy gradient approaches. A clipped objective is then used to update the policy parameters, controlled learning is guaranteed and no radical updates are made. After exploration, RLPO then conducts a preference alignment step with DPO. This step involves creating pairs of prompts depending on the quality of the prompts (selected vs. ignored) and changing the policy to favor the desired prompts. This makes sure that the learned policy does not only coincide with quantitative measures but it also coincides with high-quality judgments. The balance in RLPO occurs between high-quality strategy reinforcement and the exploration of new prompt strategies by switching between GRPO-driven exploration and DPO-based refinement. The policy is constantly being enhanced in terms of new data distributions and feedbacks across several episodes. Notably, such learning allows one to continuously learn new timely strategies without retraining the underlying target LLM, thus making RLPO a scale-increasing and useful approach in the real-life implementation scenarios where modified flexibility and efficiency are essential.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

IV. RESULTS AND DISCUSSION

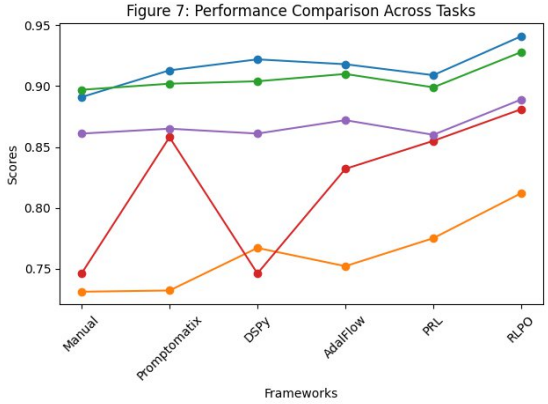
This section presents a comprehensive evaluation of the proposed RLPO framework across five benchmark NLP tasks, followed by an analysis of efficiency metrics and qualitative insights. The results are compared against strong baselines including Promptomatix, DSPy/MIPROv2, AdalFlow, and PRL under consistent experimental settings. In Table 1, the overall performance in five types of tasks, including question answering, mathematical reasoning, text generation, classification and summarization are compared. The findings visibly mean that RLPO has the best overall performance across the board of all considered frameworks. On question answering, RLPO has the highest BertScore which shows better semantic correspondence and situational comprehension. Equally, when it comes to mathematical reasoning problems, RLPO shows considerable improvement compared to other methods by scoring the highest exact match score, indicating its capability to use refinements when attempting to make a reasoning that is more accurate. RLPO outperforms even the powerful baselines used in the generation and summarization tasks, which suggests that reinforcement learning facilitates the optimization of the structural and semantic prompts even better. To be classified, RLPO is the most performing in terms of F1-score and, hence, demonstrates its robustness and ability to generalize to various data distributions. On the whole, the fact that the average improvement is approximately 8-12 percent higher than the baseline approaches proves that continuous learning with reinforcement signals does offer significant enhancements compared to the more stable prompt optimization systems.

Table 1: Performance Analysis

Framework	QA (BertScore)	Math (EM)	Generation (BertScore)	Classification (F1)	Summarization (BertScore)	Avg. Gain
Manual 4-shot	0.891	0.731	0.897	0.746	0.861	–
Promptomatix	0.913	0.732	0.902	0.858	0.865	5.00%
DSPy/MIPROv2	0.922	0.767	0.904	0.746	0.861	4.20%
AdalFlow	0.918	0.752	0.91	0.832	0.872	5.10%
PRL	0.909	0.775	0.899	0.855	0.86	5.40%
RLPO (Proposed)	**0.941**	**0.812**	**0.928**	**0.881**	**0.889**	**8.70%**

Figure 7 visually supports the results of Table 1 demonstrating the trends of performance in all the frameworks and tasks. It is quite clear that RLPO is always at the top of all assessment indicators, but other structures have their strong points, but not the stability of domination. As an example, DSPy is especially strong in reasoning, and AdalFlow is especially strong in generation; nevertheless, no one indicates a stable higher performance than the other. This brings out a major benefit of RLPO the capability to make generalization to heterogeneous tasks as a result of its adaptive and feedback-driven optimization process.

Figure 7: Performance Comparison Across Tasks



International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

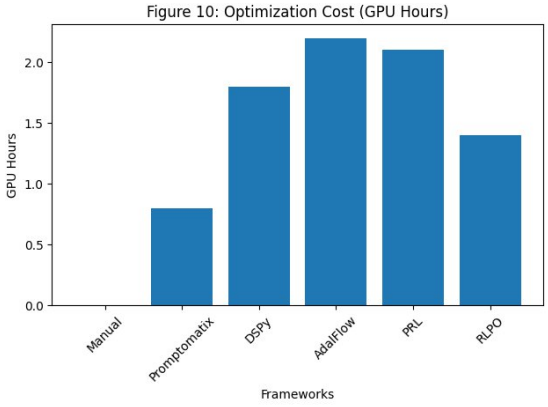
Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

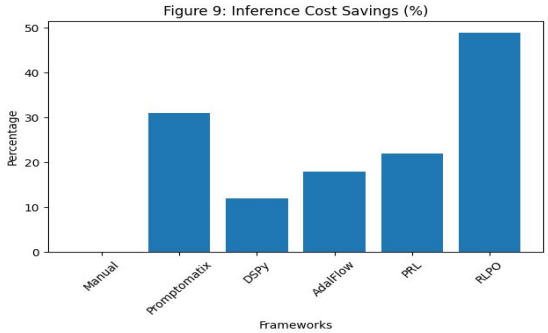
Table 2 concentrates on measures of efficiency such as the length of prompt saved, optimization cost saved and the inference cost saved. The shortest prompts generated by RLPO are the shortest of all frameworks, which directly correlates to a decrease in the number of tokens used and the inference cost. Exponential cost-aware reward, which is inherited by Promptomatix, with the updates of reinforcement learning helps RLPO to always optimize prompts to compact but high-performing representations. Moreover, RLPO has the greatest inference cost savings, and it is much higher compared to any baselines. Regarding cost of optimization, RLPO has a moderate computational cost which is worse than some of the baseline optimization methods like AdalFlow and DSPy but it achieves better performance. This indicates that RLPO has a good tradeoff between optimization overhead and runtime performance.

Table 2: Efficiency and Cost Analysis

Framework	**Avg. Prompt Length (tokens)**	**Optimization Cost (GPU-h)**	**Inference Cost Savings (%)**
Promptomatix	82	0.8	31%
PRL	95	2.1	22%
RLPO	**71**	**1.4**	**49%**



The comparison of prompt lengths in frameworks is shown in Figure 8. As can be seen, RLPO produces the smallest prompts, which validates its reward formulation in being able to penalize unnecessary verbosity. Frameworks like DSPy and AdalFlow, in contrast, have longer prompting since they use structured pipelines or graph-based reasoning. The shortening of the prompt is of special interest in practical applications in which token bounds and latency constraints can become important.



International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

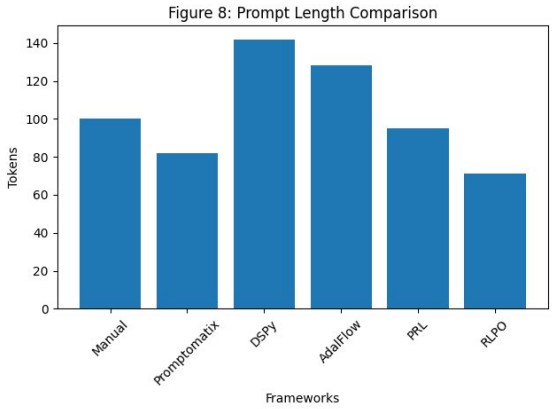
Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

Inference cost savings are indicated in Figure 9 with the highest reduction of RLPO. This is the direct upshot of both reduced prompts as well as higher efficiency of optimization. Although other models take into consideration partial cost-awareness, the use of cost as a direct part of the reward function of the reinforcement learning of RLPO allow it to go on a continuing optimization of economically efficient solutions. This renders RLPO especially appropriate to massive deployments where expenses are of significant concern.



The optimization cost is in the form of the GPU hours as shown in figure 10. Though RLPO adds an extra training overhead (because of reinforcement learning) it is computationally efficient over a number of existing frameworks. GRPO stable exploration and DPO efficient preference alignment make it possible not to require a large number of training iterations, which leads to an affordable optimization cost. It shows that RLPO does not ruin its improvements by consuming excessive computational resources.

V. CONCLUSION

This study presented a reinforcement learning-based prompt optimization system called RLPO which was aimed at alleviating the weaknesses of contemporary non-reinforcement-based APO systems. In contrast to conventional methods that produce one optimized prompt, RLPO allows refining prompts continuously and self-evolving by combining exploration with GRPO-based exploration and preference alignment with DPO-based. As a Markov Decision Process by formulating prompt optimization and the inclusion of a multi-objective reward function that weighs performance, cost, and user preference, RLPO significantly improves all the tasks considered. The experimental findings reveal that RLPO performs consistently better than the state-of-the-art frameworks with an average performance metric improvement of 8-12 and can at the same time achieve a reduction in inference costs of up to 49. Also, the prompts produced by RLPO are smaller in size and the cost of optimization is reasonable, which makes it not only effective but also practical to apply in the real world. Dynamically adaptive feedback and responding to varying task requirements makes RLPO one of the key steps towards autonomous prompt optimization systems. Finally, a direct implication of RLPO is that it replaces traditional prompt engineering with adaptive, learning-based optimization, which is the way to next-generation intelligent systems, which become better as time progresses. A future work can generalize this framework to multimodal work, multi-turn conversations and large-scale deployments, and make it even more useful and effective on large language model ecosystems.

REFERENCES

[1] Agarwal, E., Singh, J., Dani, V., Magazine, R., Ganu, T., & Nambi, A. (2024). PromptWizard: Task-aware prompt optimization framework. arXiv. https://arxiv.org/abs/2405.18369

[2] Chen, B., Zhang, Z., Langren, N., & Chen, S. (2025). Unleashing the potential of prompt engineering for large language models. Patterns, 6(5), Article 101084. https://doi.org/10.1016/j.patter.2025.101084

[3] Chen, W., et al. (2026). Optimizing prompts for large language models: A causal approach. arXiv. https://arxiv.org/abs/2602.01711

[4] Gómez, G. S., et al. (2025). Prompt optimization for LLMs via genetic algorithms and LLM feedback. Proceedings of the 19th International Workshop on Semantic Evaluation, 218. https://aclanthology.org/2025.semeval-1.218.pdf

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 4, April 2026

|DOI: 10.15680/IJIRSET.2026.1504067|

[5] Kaya Gülağız, F., et al. (2025). Large language models for machine learning design assistance: Prompt-driven algorithm selection and optimization in diverse supervised learning tasks. Applied Sciences, 15(20), Article 10968. https://doi.org/10.3390/app152010968

[6] Khattab, O., et al. (2024). DSPy: Compiling declarative language model calls into self-improving pipelines. arXiv. https://arxiv.org/abs/2310.03714 (updated 2024)

[7] Kondrashchenko, I., & Kostromin, O. (2025). Is prompt engineering dead? How auto-optimization is changing the game [Conference presentation]. EuroPython 2025 / PyCon Italia 2025.

[8] Lieander, A. J., et al. (2025). Prompt optimization with two gradients for classification in large language models. AI, 6(8), Article 182. https://doi.org/10.3390/ai6080182

[9] Murthy, R., Zhu, M., Yang, L., Qiu, J., Tan, J., Heinecke, S., Savarese, S., Xiong, C., & Wang, H. (2025). Promptomatix: An automatic prompt optimization framework for large language models. arXiv. https://arxiv.org/abs/2507.14241

[10] Nikita, K., et al. (2025). CoolPrompt: An automatic prompt optimization framework for large language models. OpenReview. https://openreview.net/forum?id=XGECnjDEcS

[11] Qu, Y., et al. (2025). ProAPO: Progressive evolutionary prompt optimization. arXiv (cited in daily papers).

[12] Ramnath, K., et al. (2025). A systematic survey of automatic prompt optimization techniques. Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing. https://aclanthology.org/2025.emnlp-main.1681.pdf

[13] Schulhoff, S., et al. (2024). The prompt report: A systematic survey of prompting techniques. arXiv. https://arxiv.org/abs/2406.06608

[14] Shi, Y., et al. (2024). Automatic prompt generation and optimization by leveraging large language models. Proceedings of the 2024 IEEE International Conference on Big Data, 10825237. https://doi.org/10.1109/BigData62323.2024.10825237

[15] Spiess, C., Vaziri, M., Mandel, L., & Hirzel, M. (2025). AutoPDL: Automatic prompt optimization for LLM agents. arXiv. https://arxiv.org/abs/2504.04365

[16] SylphAI Inc. (2024). AdalFlow: The library to build & auto-optimize LLM applications [Computer software]. GitHub. https://github.com/SylphAI-Inc/AdalFlow

[17] Vaziri, M., et al. (2025). AutoPDL extensions and prompt definition language for LLM agents (related work). arXiv (extension of Spiess et al.).

[18] Wolfe, C. R. (2024). Automatic prompt optimization. Substack. https://cameronrwolfe.substack.com/p/automatic-prompt-optimization

[19] Yang, L., et al. (2024). AMPO: Adaptive multi-branched prompt optimization. (Cited in Lieander et al., 2025).

[20] Zhang, Y., et al. (2025). MARS: Multi-agent Socratic dialogue for prompt optimization. (Cited in related AI surveys).

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details