Auto-BPO: Automated Gradient-Free Black-Box Prompt Optimization

Yan Liu
y.liu17@student.tue.nl
Eindhoven University of Technology
Eindhoven, North Brabant
Netherlands

Naga Dheeraj Mukkara
n.d.mukkara@student.tue.nl
Eindhoven University of Technology
Eindhoven, North Brabant
Netherlands

Tingrui Huang
t.huang2@student.tue.nl
Eindhoven University of Technology
Eindhoven, North Brabant
Netherlands

Jingyao Lu
j.lu3@student.tue.nl
Eindhoven University of Technology
Eindhoven, North Brabant
Netherlands

Anh Tram Thi Pham
a.pham.thi.tram.anh@student.tue.nl
Eindhoven University of Technology
Eindhoven, North Brabant
Netherlands

Abstract

Large language models (LLMs) have demonstrated remarkable capabilities in following instructions and understanding human preferences. However, effectively aligning LLMs with human intent remains challenging, primarily due to the gap between the intentions conveyed through prompts and the model's interpretation of them. Existing approaches typically guide LLMs toward human-aligned behavior using reinforcement learning and direct preference optimization methods. Yet, these approaches face critical challenges: they are computationally expensive and prone to training instability on large models or datasets, incompatible with closed-source, API-only models, and lack interpretability, obscuring why they align with human preferences.

To address these challenges, we propose a novel, efficient, and interpretable alignment paradigm, Automated Black-Box Prompt Optimization (Auto-BPO). Auto-BPO operates without modifying the model and employs a gradient-free prompt optimizer to automatically rewrite unclear or poorly structured human prompts into forms better understood by LLMs, leading to responses that better align with human preferences.

Experimental results demonstrate that Auto-BPO can enhance the model's understanding of human intent and the consistency of its responses without accessing internal parameters, providing a new pathway toward scalable alignment of closed-source large language models.

CCS Concepts

• **Information systems → Language models**; *Data mining*; **Information extraction**; **Language models**.

Keywords

Large Language Models (LLMs), Black-Box Prompt Optimization (BPO), Prompt Engineering, Gradient-Free Methods

ACM Reference Format:
Yan Liu, Naga Dheeraj Mukkara, Tingrui Huang, Jingyao Lu, and Anh Tram Thi Pham. 2026. Auto-BPO: Automated Gradient-Free Black-Box Prompt Optimization. In *Proceedings of The 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '26)*. ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Recent advances in LLMs have significantly transformed the information technology landscape [20]. One clear example of this change is the rise of online conversational agents, such as ChatGPT and Claude [15], which can generate coherent and meaningful responses to user queries across a wide range of topics. By leveraging these systems, users can greatly enhance their efficiency in tasks like acquiring knowledge, exploring creative ideas, brainstorming collaboratively, and organizing activities [21].

However, while LLMs have achieved impressive fluency and reasoning ability, effectively aligning their behavior with human intent remains a major challenge. The gap between how users express their intentions through prompts and how models interpret those prompts often leads to inconsistent or suboptimal responses. [18] Existing alignment approaches, such as Reinforcement Learning from Human Feedback (RLHF) [25], Reinforcement Learning from AI Feedback (RLAIF) [6], and Direct Preference Optimization (DPO) [11], have made progress in modeling human preferences [17]. Yet, these methods require gradient access, extensive human annotation, and significant computational cost, making them impractical for closed-source LLMs like GPT-4 or Claude [1]. As a result, current techniques cannot be directly applied to improve the alignment of black-box models that are only accessible through APIs.

Shin et al. [12] proposed AutoPrompt, a gradient-guided automated prompt generation method designed to explore the knowledge embedded in pretrained language models under unsupervised settings. This approach reformulates downstream tasks as fill-in-the-blank problems, eliminating the need for manually crafted prompts and thus significantly reducing human bias and labor.

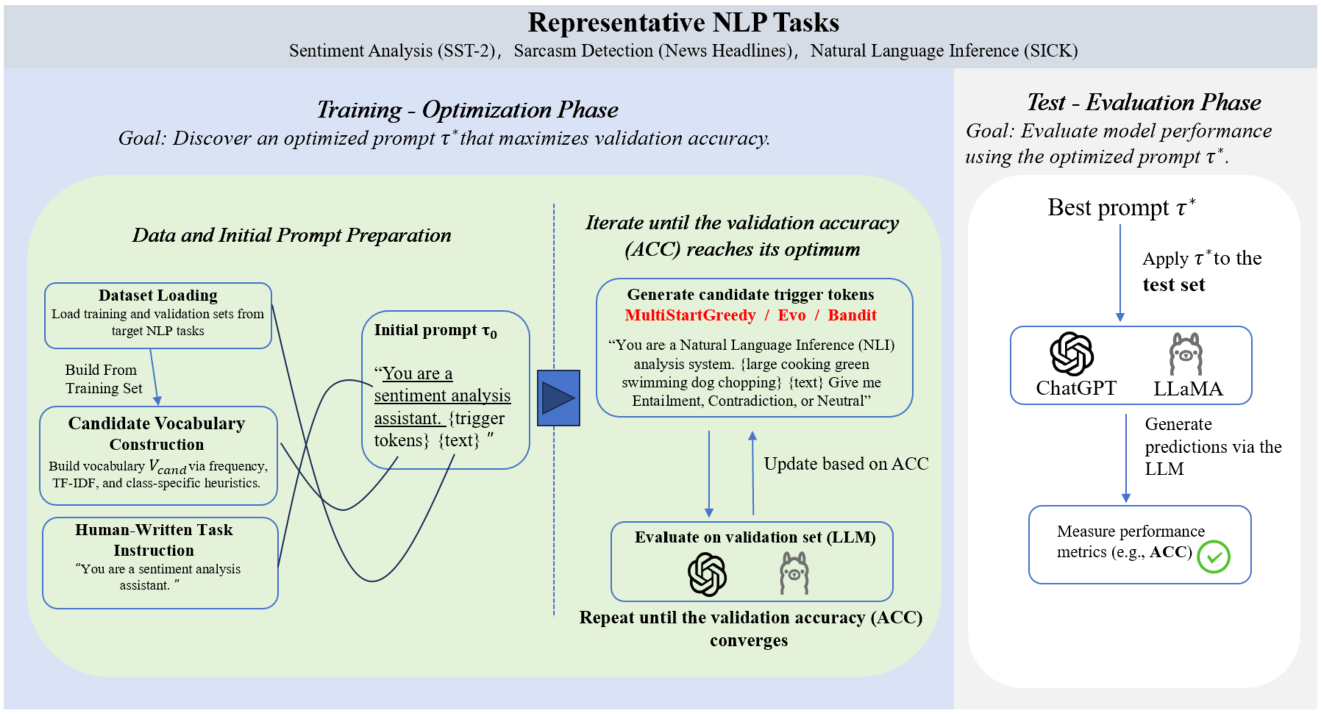
Representative NLP Tasks
Sentiment Analysis (SST-2), Sarcasm Detection (News Headlines), Natural Language Inference (SICK)
Training - Optimization Phase
Goal: Discover an optimized prompt τ* that maximizes validation accuracy.
Data and Initial Prompt Preparation
Dataset Loading
Load training and validation sets from target NLP tasks
Build From Training Set
Candidate Vocabulary Construction
Build vocabulary V_cand via frequency, TF-IDF, and class-specific heuristics.
Human-Written Task Instruction
"You are a sentiment analysis assistant."
Initial prompt τ0
"You are a sentiment analysis assistant. {trigger tokens} {text}"
Iterate until the validation accuracy (ACC) reaches its optimum
Generate candidate trigger tokens
MultiStartGreedy / Evo / Bandit
"You are a Natural Language Inference (NLI) analysis system. {large cooking green swimming dog chopping} {text} Give me Entailment, Contradiction, or Neutral"
Update based on ACC
Evaluate on validation set (LLM)
Repeat until the validation accuracy (ACC) converges
Test - Evaluation Phase
Goal: Evaluate model performance using the optimized prompt τ*.
Best prompt τ*
Apply τ* to the test set
ChatGPT
LLaMA
Generate predictions via the LLM
Measure performance metrics (e.g., ACC)

Figure 1: Auto-BPO enables automated, gradient-free prompt optimization for large language models treated as black boxes. Given a task instruction and labeled validation data, Auto-BPO constructs and evaluates candidate prompts by inserting token-level trigger sequences into a fixed template. Three independent optimizers—multi-start greedy search, evolutionary global exploration, and bandit-based adaptive optimization—can each be used independently, leveraging validation accuracy as the optimization signal to iteratively refine prompts. The prompt achieving the highest validation performance is then applied in the testing stage, enabling interpretable and gradient-free optimization without accessing model parameters or gradients.

However, this method can only be applied to open-source models with accessible parameters. Cheng et al. [1] proposed a Black-Box Prompt Optimization (BPO) framework to align LLMs without further training by optimizing user prompts based on human preferences, achieving up to 22% improvement in ChatGPT's win rate and surpassing traditional alignment methods such as PPO and DPO. Although BPO treats the target LLM as a black box without parameter updates, it still trains a lightweight seq2seq model as a prompt preference optimizer to map human-written prompts to optimized ones. Consequently, gradient-based optimization is still required during the training of this auxiliary model. To address this gap, our work explores a fully gradient-free alignment pathway that requires neither parameter access nor auxiliary model training. Unlike AutoPrompt, which depends on gradient signals from open-source models, or BPO, which still involves gradient-based optimization of a prompt preference model, our method treats the target LLM as a complete black box. We focus solely on optimizing the communication interface—the prompt itself— through discrete, search-based strategies guided purely by model feedback, without any gradient computation or parameter updates.

In this paper, we introduce Auto-BPO, an automated gradient-free method for prompt optimization for any task, illustrated in Figure 1. Unlike previous approaches that rely on backpropagation or access to internal model gradients, Auto-BPO treats the language model entirely as a black box and optimizes prompts through adaptive search over token-level trigger combinations. It investigates the interaction between human instructions and model behavior by automatically optimizing task-specific trigger tokens, leading to improved task accuracy and consistency.

Our contributions can be summarized as follows:

- We propose Auto-BPO, a novel automated and gradient-free framework for prompt optimization that treats large language models as black boxes. This framework enhances LLM alignment and performance by refining human-written prompts based solely on model outputs, making it fully compatible with API-only models and eliminating the need for internal access to parameters or gradients.
- Auto-BPO integrates three independent search strategies which are greedy, evolutionary, and bandit-based optimization. This provides an interpretable, low-cost, and friendly implementation solution adaptable to both open- and closed-source models such as GPT-3.5.
- Extensive experiments demonstrate that Auto-BPO consistently enhances accuracy and stability across multiple NLP tasks, including sentiment classification, sarcasm detection, and reasoning-oriented sentence understanding.

2 Related Work

2.1 From Manual Prompt Engineering to Automated Prompt Tuning

Prompt engineering has long been the primary approach for adapting LLMs to downstream tasks. In its early form, manual prompt engineering relied on human intuition, domain expertise, and repeated trial-and-error to craft instructions or examples that could elicit desired behaviors from models such as GPT-3.5. Techniques such as few-shot prompting [4], chain-of-thought prompting [19], and role-based prompting [8] allowed users to guide model reasoning without modifying parameters. While effective in specific contexts, this manual approach is inherently limited by high human effort, low reproducibility, and poor scalability. Even minor changes in syntax, punctuation, or wording can lead to significant performance fluctuations, revealing LLMs' extreme sensitivity to input formulation. Moreover, manually designed prompts are static and often fail to adapt to dynamic contexts or diverse downstream tasks, making the process costly, time-consuming, and difficult to generalize across models [7].

To overcome these challenges, Automated Prompt Optimization (APO) [26] has been proposed to replace manual, heuristic prompt design with algorithmic search and adaptation. APO formulates prompt generation as an optimization problem over discrete or continuous spaces, automatically exploring candidate prompts to maximize model performance. This process reduces human effort and improves efficiency, stability, and reproducibility across tasks and models.

Interestingly, optimized prompts are sometimes syntactically invalid or semantically meaningless yet yield superior results. This suggests that LLMs do not rely solely on human language rules but instead respond to internal activation patterns. Thus, the most effective prompts may be opaque to humans while precisely guiding model behavior, highlighting a tension between interpretability and performance.

Existing APO approaches include foundation-model-based self-optimization, evolutionary search, parameter-efficient fine-tuning (PEFT), reinforcement learning, and meta-learning. Each represents different trade-offs among efficiency, interpretability, and computational cost.

2.2 Preference Alignment via Reinforcement Learning and Direct Optimization

Prompt generation and selection play a critical role in instruction-following and human preference understanding [1, 2, 10, 23]. Deng et al. [2]propose a reinforcement learning approach that eliminates the dependence on gradient access or manual engineering by training a parameter-efficient policy network to generate discrete prompts via sequential token-wise selections. Prasad et al. [10] introduce a gradient-free edit-based method that improves instructional prompts through phrase-level operations such as delete, swap, paraphrase, and add, making it suitable for API-based language models. Zhang et al. [23] present a test-time prompt editing framework that uses reinforcement learning to adapt and refine prompts for individual queries, supporting interpretable, query-specific prompt optimization across diverse tasks. [1] Cheng et.al propose a model-agnostic Black-Box Prompt Optimization (BPO) method, which does not require training the large language model itself but only trains a small sequence-to-sequence model. By optimizing user prompts without updating the model parameters, it enables large language models to better align with human intent and has been shown to significantly improve performance in experiments.

While prior work has advanced prompt selection and optimization through rule-based policies, learning-based methods, and large language models, existing approaches suffer from several limitations. As models scale, the computational cost and complexity of training increase dramatically, particularly when relying on unstable reinforcement learning algorithms such as PPO and DPO. Moreover, most high-performing LLMs, such as GPT-4 and Claude 2, are closed-source and accessible only via APIs, making these training approaches infeasible for external users. Finally, these methods often lack interpretability, as the exact relationship between model updates and improvements in human preferences remains opaque.

2.3 Black-Box Prompt Optimization

With the increasing deployment of large-scale LLMs such as GPT-3.5 and GPT-4 as commercial APIs, users are often restricted to black-box access — obtaining model outputs without internal gradients or parameters. This constraint has motivated a growing body of research on Black-Box Prompt Optimization (BPO), which aims to adapt or align LLMs by optimizing prompts solely through model queries.

Early work by Sun et al. [14] introduced Black-Box Tuning, applying derivative-free optimization in low-dimensional subspaces and showing that such methods can rival gradient-based tuning. Diao et al. [3] extended this idea to discrete prompts using policy-gradient estimation, enabling efficient optimization through model queries alone.

Cheng et al. [1] further positioned BPO as an alignment framework, leveraging human preference feedback to align model behavior without parameter updates, outperforming PPO- and DPO-based alignment. To improve transferability, Zheng et al. [24] proposed Subspace Learning (BSL), which meta-learns task-shared low-dimensional spaces for efficient adaptation. Yu et al. [22] extended BPO to vision-language models using evolutionary search for joint multimodal prompt tuning.

Overall, BPO offers a scalable, model-agnostic solution for optimizing prompts in closed-source LLMs, achieving strong task performance and alignment purely through inference-level interaction.

3 Proposed Method: Automated Black-Box Prompt Optimization

To address the challenge of optimizing prompts for LLMs without accessing their gradients or parameters, we propose Auto-BPO, an automated gradient-free framework for black-box prompt optimization. As illustrated in Figure 1, Auto-BPO begins with a human-written task instruction and a labeled validation dataset. It then constructs candidate prompts by inserting token-level trigger sequences drawn from a predefined vocabulary. Each candidate

Table 3.1: Key Notations.

Symbol	Description
N	Number of validation samples
L	Length of the trigger sequence (number of tokens)
$\mathcal{V}$	Candidate vocabulary set
$\tau = (t_1, \ldots, t_L)$	Trigger sequence to be optimized
x_i, y_i	Input instance and ground-truth label for example i
$T(\tau, x)$	Prompt template inserting τ into input x
f_θ	Target LLM treated as a black box
$J(\tau)$	Validation accuracy for τ
R	Number of independent restarts in the multi-start greedy search
P, $G_{\max}$	Population size and the maximum number of generations in the evolutionary algorithm
ϵ	Exploration rate in the ϵ-greedy strategy used for bandit-based search

prompt is evaluated through model inference, and the resulting validation accuracy serves as the optimization feedback.

Auto-BPO aims to automatically discover trigger tokens that maximize model performance on the validation set. To provide flexibility and robust performance across diverse tasks, the framework employs three distinct gradient-free search strategies: (1) Multi-Start Greedy Search, (2) Evolutionary Search, and (3) Bandit-Based Prompt Search. All optimization strategies adhere to a unified problem formulation (defining the search space, objective function, and evaluation process), thereby providing a consistent and flexible toolkit for black-box prompt tuning.

Problem formulation. The objective of Auto-BPO is to learn an optimal sequence of discrete trigger tokens $\tau^* = (t_1, t_2, \ldots, t_L)$, which together constitute the *trigger prompt* inserted into a fixed task template. We define a fixed task template into which the learned trigger prompt τ is inserted. The template takes the following form:

```
You are a <task> assistant {trigger tokens}
{text}
```

Here:

- `<task>` specifies the target task, such as `sentiment analysis`, `sarcasm detection`, or `natural language inference`.
- `{trigger tokens}` corresponds to the learned sequence $\tau = (t_1, t_2, \ldots, t_L)$.
- `{text}` is the input instance, typically a sentence or short paragraph.

Each token t_i is selected from the candidate vocabulary $\mathcal{V}$, and the complete model input is formed by combining the task instruction, the trigger prompt τ, and the input example according to a predefined template. Formally, we denote:

$$\tau = (t_1, t_2, \ldots, t_L), \quad t_i \in \mathcal{V}. \tag{1}$$

where $\mathcal{V}$ is the candidate vocabulary and L is the predetermined, fixed trigger length. Given a validation dataset with N samples, $\{(x_i, y_i)\}_{i=1}^{N}$, where each x_i is a textual input (e.g.,"remarkable procession of sweeping pictures") and y_i is its corresponding label (e.g., 0 for negative sentiment). The performance of a trigger prompt configuration τ is evaluated using the empirical validation accuracy, defined as:

$$J(\tau) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[f_\theta\left(T\left(\tau, x_i\right)\right) = y_i\right]. \tag{2}$$

Here, f_θ is the fixed, black-box LLM, $T(\tau, x_i)$ is the full composed input formed by prepending the trigger tokens τ to the instance x_i, and $\mathbf{1}[\cdot]$ is the indicator function returning 1 if the model prediction matches the ground truth label y_i. The overall search objective is to solve the discrete search problem by identifying the optimal token sequence τ^* within the fixed search space $\mathcal{V}$ that maximizes the accuracy $J(\tau)$:

$$\tau^* = \arg\max_{\tau \in \mathcal{V}} J(\tau). \tag{3}$$

This optimization relies only on black-box feedback from the model's output predictions.

While all operate under the same gradient-free black-box setting, three search strategies (Greedy, Evolutionary, and Bandit) differ in how candidate prompts are generated, evaluated, and updated. These diverse optimization perspectives independent, efficient, and interpretable solutions for improving prompt quality. The key symbols and notations used throughout this section are summarized in Table 3.1. The following subsections detail each optimization strategy, describing its search formulation, update mechanism, and algorithmic design.

3.1 Multi-Start Greedy Search for Prompt Optimization

The Multi-Start Greedy Search (MSGS) is a local optimization method that aims to achieve stable and continuous improvement in model performance (e.g., $J(\tau)$), especially under a limited query budget. In black-box settings, calling the model many times can be costly, so a simple and efficient greedy search is preferred. In each step, MSGS makes small local changes to the current prompt sequence τ (such as replacing or swapping tokens) and only accepts those changes that improve accuracy. This process iteratively and reliably moves toward a better trigger prompt τ^*.

Algorithmic principle. The MSGS algorithm starts from a randomly initialized trigger sequence $\tau^{(0)}$, where each token in $\tau^{(0)}$ is independently sampled from the candidate vocabulary $\mathcal{V}$ to form a list of L trigger tokens sampled uniformly at random. This initialization provides diverse starting points for the search process and helps the algorithm escape local optima. At each iteration, MSGS explores a local neighborhood $\mathcal{N}(\tau^{(k)})$ around the current trigger prompt $\tau^{(k)}$, obtained by mutating one or more tokens:

$$\mathcal{N}(\tau^{(k)}) = \{\tau' \mid \tau' = \text{Mutate}(\tau^{(k)})\}. \tag{4}$$

Mutation operations include token-level *insert*, *delete*, *swap*, and *replace* operations. At each iteration, the best-performing neighbor is accepted only if it yields an improvement in accuracy (the objective function $J(\tau)$):

$$\tau^{(k+1)} = \arg\max_{\tau' \in \mathcal{N}(\tau^{(k)})} J(\tau'), \quad \text{if } J(\tau^{(k+1)}) > J(\tau^{(k)}). \tag{5}$$

The search halts after P_{stop} consecutive non-improving steps. To mitigate local optima, the process restarts R times using a new random initialization each time. The final optimized output is the best result across all R restarts:

$$\tau^* = \arg \max_{r \in \{1,\ldots,R\}} J(\tau_r^{(K_r)}), \tag{6}$$

where K_r denotes the number of iterations before convergence for restart r.

Convergence and complexity. Each greedy run is guaranteed to converge because $J(\tau)$ strictly increases upon acceptance and $\mathcal{V}$ is finite. The computational cost is $O(R \times L \times N_{\text{neigh}})$ dominated by forward evaluations of f_θ (no backpropagation required). MSGS thus achieves strong anytime performance and often finds near-optimal prompts with minimal computational overhead.

Advantages. MSGS is simple, interpretable, and budget-efficient. It mitigates local optima, produces human-readable prompt edits, and captures token-level dependencies via full-prompt neighborhood evaluation. Moreover, its gradient-free nature ensures compatibility with closed-source, API-only models.

3.2 Evolutionary Search for Prompt Optimization

Evolutionary Algorithms (EAs) emulate natural selection to search vast, discrete spaces effectively. They provide global exploration capabilities through crossover and mutation, avoiding premature convergence—a key limitation of greedy and bandit methods. In the Auto-BPO framework, EA serves as the global optimizer, generating a diverse pool of candidate prompts that balance exploration and exploitation.

Algorithmic principle. The algorithm initializes a population $\mathcal{P}^{(0)} = \{\tau_1, \ldots, \tau_P\}$ sampled from $\mathcal{V}$. Each τ_i is evaluated by its fitness $J(\tau_i)$. At each generation, four steps are performed:

(1) **Selection:** High-fitness individuals are selected with probability proportional to $J(\tau_i)$.
(2) **Crossover:** Pairs of prompts exchange token segments to generate offspring, encouraging diversity.
(3) **Mutation:** Each offspring undergoes random token-level modifications with probability p_{mut}.
(4) **Elitism:** The top-E individuals are preserved to ensure fitness does not decrease.

This evolutionary cycle repeats for $G_{\max}$ generations or until convergence. The best-performing τ in the final generation is returned as τ^*.

Complexity and convergence. Each generation evaluates P individuals, leading to a total cost of $O(P \times G_{\max})$ model queries. Elitism ensures monotonic improvement in best fitness. While global optimality is not guaranteed, empirical performance across tasks shows robust convergence to high-quality prompts.

Advantages. EA achieves effective global search through population diversity and stochastic recombination. It is gradient-free, compatible with API-based models (e.g., GPT-3.5), and offers interpretable prompt evolution trajectories over generations, illustrating how token-level improvements accumulate.

3.3 Bandit-Based Prompt Search

To optimize discrete tokens efficiently without gradient information, we adopt a multi-armed bandit formulation based on the ϵ-greedy algorithm [16]. Each trigger position in the prompt is treated as an independent bandit arm, where each possible token corresponds to an available action. This approach focuses on adaptive action selection which chooses the best token per position rather than modeling full state transitions as in reinforcement learning. It provides a lightweight and fully black-box optimization mechanism that is suitable for both open- and closed-source models.

Algorithmic principle. The bandit-based search treats each token position $i \in \{1, \ldots, L\}$ as an independent multi-armed bandit, where each candidate token $t \in \mathcal{V}$ corresponds to an arm. At the beginning, all reward estimates are initialized to zero, and the initial trigger sequence $\tau^{(0)}$ is formed by sampling L tokens uniformly at random from the candidate vocabulary $\mathcal{V}$. At each iteration, a position is randomly selected and updated following an ϵ-greedy strategy: with probability ϵ, the algorithm explores by sampling a random token, and with probability $1 - \epsilon$, it exploits the token with the highest empirical reward estimate. The updated prompt is evaluated on the validation set to obtain a reward signal, which is then used to refine the corresponding local reward estimates. Throughout the optimization, the best-performing prompt is continuously tracked, and after T iterations, the final trigger sequence τ^* is evaluated on the held-out test set.

Complexity and convergence. Per iteration, token selection requires $O(|\mathcal{V}|)$ operations, while model evaluation dominates the cost (C_{eval}). Thus, the total time complexity is $O\big(T \cdot (|\mathcal{V}| + C_{\text{eval}})\big)$ and space complexity is $O(L|\mathcal{V}|)$ due to per-token statistics. Under stationary rewards, ϵ-greedy converges to a locally optimal configuration. Larger ϵ encourages exploration but slows convergence, while smaller ϵ accelerates exploitation but risks local optima.

Advantages and limitations. The bandit method is simple, lightweight, and entirely black-box because it requires only scalar feedback from model outputs and thus works with API-based LLMs. Its ϵ-greedy control balances exploration and exploitation effectively, enabling fast adaptation to promising token combinations. However, because it assumes position-wise independence, it may miss cross-token dependencies and is sensitive to ϵ tuning. Despite this, it remains an efficient and practical option for rapid prompt optimization.

3.4 Experimental Settings

To evaluate the effectiveness of Auto-BPO, we conducted extensive experiments to answer two research questions: RQ1: *How effectively does Auto-BPO improve task performance compared to the original, unoptimized prompt across various LLMs?* RQ2: *How sensitive is the performance of Auto-BPO to the length of the trigger tokens across different LLMs?* The detailed experimental settings are described below.

Datasets. We evaluate Auto-BPO on three representative text classification benchmarks that collectively span sentiment polarity, sarcasm detection, and semantic reasoning. A brief summary of the datasets is provided in Table 3.2.

SST-2 [13] (Stanford Sentiment Treebank) is a binary sentiment dataset of 67,349 movie-review sentences labeled as positive or negative. *News Headlines* [9] is a sarcasm detection corpus containing 28,000 news headlines collected from online media such as *The Onion* and *HuffPost*, each annotated as sarcastic or non-sarcastic.

SICK [5] (Sentences Involving Compositional Knowledge) includes 9,927 sentence pairs annotated as entailment, contradiction, or neutrality. We use it as a three-class classification task to assess Auto-BPO's robustness on reasoning-oriented data.

Table 3.2: Datasets used in our experiments with task descriptions and representative examples.

Dataset	Description	Example (Sentence → Label)
SST-2	Binary sentiment classification with positive/negative labels.	"The movie is charming and beautifully acted." → **Positive** "The plot is dull and predictable." → **Negative**.
News Headlines (Sarcasm)	Sarcasm detection on 28k news headlines from *The Onion* and *HuffPost*.	"World ends tomorrow, women most affected." → **Sarcastic** "Local man wins lottery for the third time." → **Non-sarcastic**.
SICK	Natural language inference labeled as *entailment*, *contradiction*, or *neutral*.	"A man is playing guitar." / "A man is making music." → **Entailment** "A woman is dancing." / "A man is sitting." → **Contradiction**.

Evaluation Metrics. All datasets are standardized for prompt-based evaluation with explicit ground-truth labels. For all datasets, the evaluation metric is classification accuracy (ACC), which is calculated as the proportion of correctly predicted samples over the total number of samples. This metric is used consistently across all model–algorithm combinations.

Candidate Vocabulary Preparation. To build a candidate vocabulary ($\mathcal{V}$), words are selected from the training data using three methods: (i) *frequency-based selection*, (ii) *TF-IDF importance analysis*, and (iii) *class-specific analysis*.

Frequency-based selection. This method selects the top K words ranked by frequency. For each word w, its frequency $f(w)$ is calculated as

$$f(w) = \sum_{i=1}^{N} \mathbf{1}_{\{w \in \mathcal{W}_i\}}, \tag{7}$$

where N is the total number of samples, $\mathcal{W}_i$ is the set of words in sample i, and $\mathbf{1}_{\{\cdot\}}$ equals 1 if the word is present and 0 otherwise. Words that appear less than a minimum frequency $f_{\min}$ or in more than a maximum fraction $r_{\max}$ of the samples are removed. In our experiments, $f_{\min}$ is set to 2 and $r_{\max}$ is set to 0.5.

TF-IDF importance analysis. This method selects words based on their Term Frequency-Inverse Document Frequency (TF-IDF) scores, which is a statistical measure that reflects how important a word is to a sample (document) in a collection (corpus) This method calculates the average TF-IDF score for each word across the entire validation dataset. Words with the highest average scores are considered the most representative and are selected to form the final candidate vocabulary.

Class-specific analysis. This method identifies words that are more prominent in certain classes. For binary classification tasks, let $f_{\text{pos}}(w)$ and $f_{\text{neg}}(w)$ denote the counts of w in the positive and negative classes, respectively. The class-specific score of a word is defined as

$$\text{diff\_score}(w) = \frac{|f_{\text{pos}}(w) - f_{\text{neg}}(w)|}{f_{\text{pos}}(w) + f_{\text{neg}}(w)}. \tag{8}$$

For three-class tasks, such as NEUTRAL, CONTRADICTION, and ENTAILMENT, let $f_c(w)$ be the count of w in class c, and $f_{\text{total}}(w) = \sum_{c=1}^{C} f_c(w)$. The class-specific score is computed by

$$\text{diff\_score}(w) = \frac{\text{std}(f_1(w), f_2(w), \ldots, f_C(w))}{f_{\text{total}}(w)}, \tag{9}$$

where std$(\cdot)$ denotes the standard deviation. Words with higher scores appear more in some classes than others, and only those with $f_{\text{total}}(w) \geq f_{\min}$ are retained.

Finally, the words selected by the three methods are combined to form the final candidate vocabulary. If the total number of words is smaller than the desired size K, words with high class-specific scores are prioritized, followed by additional words from the TF-IDF and frequency lists until the vocabulary reaches K words. This vocabulary can then be used for downstream tasks, such as prompt optimization or feature extraction.

Implementation Details. Experiments were executed on NVIDIA A100-SXM4-40GB GPUs with Python 3.11. We test three large language models (LLMs) with different architectures and access types. GPT-2 is an open-source, small-scale model; GPT-3.5 is a proprietary black-box API; and LLaMA-3 is an open-source, instruction-tuned model designed for reasoning tasks. All models were executed with their default meta-parameter configurations. This selection covers both open- and closed-source models, allowing us to evaluate Auto-BPO's performance across different model scales and access types. Prompt optimization is carried out using three gradient-free algorithms. Greedy search iteratively replaces trigger tokens with locally optimal candidates. Evolutionary search maintains a population of prompts and applies mutation and crossover to explore the search space. Bandit search treats token selection as a multi-armed bandit problem, balancing exploration and exploitation. All three algorithms follow the same pipeline: candidate prompts are generated, evaluated through model queries, and updated to maximize validation accuracy.

4 Results

4.1 Evaluation and Ablation Study of Prompt Optimization Strategies

To evaluate the contribution of each optimization component in AutoBPO, we conducted a study comparing its optimized variants—Greedy, Evolutionary, and Bandit—against the baseline setting without optimization. As shown in Table 4.3, the accuracy of three large language models (GPT-2, GPT-3.5, and LLaMA-3) is compared between the AutoBPO variants with optimal trigger lengths and the baseline. It can be observed that using AutoBPO yields a significant enhancement in accuracy over the baseline for every model under evaluation.

On average, AutoBPO achieves a **21%** relative improvement in mean accuracy across the three models, with LLaMA-3 showing the largest gain (+37%) and GPT-3.5 the smallest (+1.5%), reflecting its stronger baseline performance.

Among the variants, the Greedy algorithm consistently achieves the highest accuracy, demonstrating its efficiency in identifying

Table 4.3: ACC results on three datasets using three models with optimal trigger lengths.

Model	Method	Datasets			Mean
		SST-2	News Headlines	SICK	
GPT-2	Original Prompt	0.48	0.53	0.46	0.49
	Auto-BPO (Greedy)	**0.70**	0.55	**0.58**	0.61
	Auto-BPO (Evolutionary)	0.51	**0.56**	0.50	0.52
	Auto-BPO (Bandit)	0.48	0.55	0.49	0.51
GPT-3.5	Original Prompt	0.90	0.80	0.33	0.68
	Auto-BPO (Greedy)	**0.97**	0.52	**0.58**	0.69
	Auto-BPO (Evolutionary)	0.70	0.56	0.49	0.58
	Auto-BPO (Bandit)	0.90	0.56	0.42	0.63
LLaMA-3	Original Prompt	0.87	0.60	0.15	0.54
	Auto-BPO (Greedy)	**0.97**	**0.67**	**0.58**	0.74
	Auto-BPO (Evolutionary)	0.96	0.58	0.51	0.68
	Auto-BPO (Bandit)	0.95	0.57	0.50	0.67

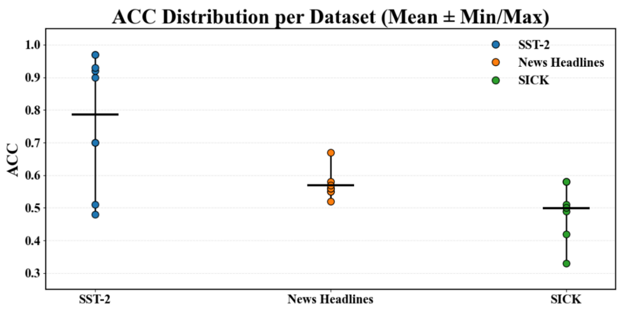

Figure 4.2.1: Distribution of ACC score across datasets and optimization methods at optimal trigger lengths.

locally optimal trigger tokens. On GPT-3.5, Greedy nearly reaches the performance ceiling on SST-2 (~0.97 ACC), showing that even lightweight optimization yields substantial gains. The Evolutionary algorithm provides steady improvements through broader search exploration, which is particularly beneficial for semantically complex tasks such as SICK. The Bandit method attains competitive performance with lower computational cost, offering a favorable balance between accuracy and efficiency, especially in resource-constrained scenarios.

Figure 4.2.1 further presents the accuracy distributions of all model–algorithm combinations across the three datasets (SST-2, News Headlines, and SICK), each evaluated at its optimal trigger length. Among them, SST-2 achieves the highest mean accuracy and shows the largest improvement range, suggesting that it is highly sensitive to optimization. News Headlines demonstrates moderate yet consistent gains, reflecting its reliance on surface-level lexical patterns. In contrast, SICK remains the most challenging dataset, where improvements are relatively limited due to its reasoning-intensive nature.

4.2 Parameter Sensitivity Analysis with Respect to Trigger Length

We evaluate the robustness of Auto-BPO under varying trigger lengths by inserting $L \in \{3, 5, 7, 9\}$ tokens into prompts, and conduct experiments on GPT-2, GPT-3.5, and LLaMA-3 across three datasets: SST-2, News Headlines, and SICK. The corresponding results are shown in Figure 4.3.1 and detailed in Appendix A.1.

Results indicate that trigger length has minimal impact on model performance, confirming the robustness of the method. Among optimization strategies, the Greedy algorithm performs most stably; the Evolutionary algorithm shows slight gains with longer triggers; and the Bandit approach remains reliable despite some variance. Task type proves more influential than sequence length: reasoning tasks benefit slightly from longer triggers, while sentiment classification does not. Overall, Auto-BPO's optimization strategies demonstrate robustness to changes in trigger length.

4.3 Efficiency and Runtime–Accuracy Trade-off Analysis

To evaluate the computational efficiency of Auto-BPO, we compare the runtime and accuracy of three optimization algorithms on the SST-2 task using LLaMA-3. As shown in Figure 4.4.1, runtime increases with trigger length L for all methods, reflecting the growth in search space complexity. Among them, Greedy search achieves the highest accuracy but incurs the greatest computational cost, as it evaluates the full prompt in each iteration. In contrast, the Bandit-based method is the fastest, updating only one token per step, yet yields the lowest accuracy. The Evolutionary algorithm offers a balanced trade-off between efficiency and performance.

These results highlight a clear runtime–accuracy trade-off: Greedy search is suitable for accuracy-critical scenarios, Bandit is optimal for resource-limited or real-time settings, and Evolutionary provides a middle ground. All algorithms scale stably with trigger length, demonstrating that Auto-BPO can flexibly adapt to different computational budgets and application requirements.

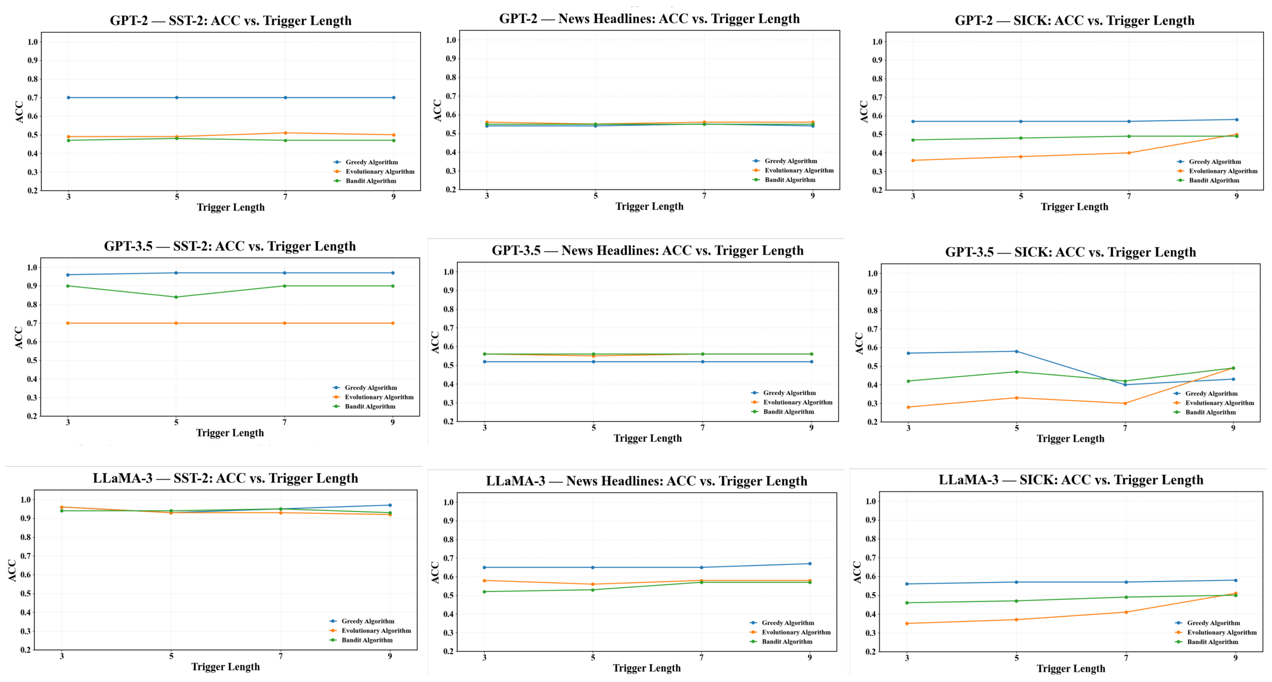

Figure 4.3.1: ACC performance of different optimization algorithms across models and datasets with varying trigger lengths.

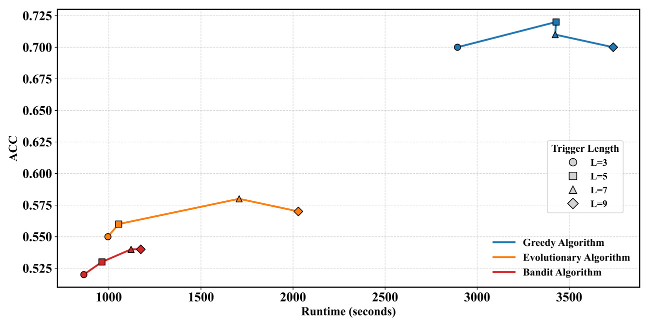

Figure 4.4.1: Comparison of runtime and ACC for Greedy, Evolutionary, and Bandit algorithms on the SST-2 task (LLaMA-3). Different shapes indicate trigger lengths (L = 3, 5, 7, 9).

5 Conclusions

In this paper, we proposed Auto-BPO, an automated and gradient-free framework for prompt optimization that treats large language models (LLMs) as complete black boxes. Unlike prior methods that rely on model gradients or internal access, Auto-BPO focuses on optimizing the prompt itself based solely on model outputs as feedback signals. The framework unifies three independent search strategies—greedy, evolutionary, and bandit-based optimization—offering a flexible and interpretable solution that adapts to both open- and closed-source LLMs. Extensive experiments across multiple NLP tasks, including sentiment analysis, sarcasm detection, and reasoning-based inference, demonstrate that Auto-BPO consistently improves model accuracy and stability. These results highlight the effectiveness of black-box prompt optimization as a practical direction for enhancing LLM performance without requiring parameter-level access.

Limitations and Future work

While Auto-BPO demonstrates strong performance in aligning black-box large language models through gradient-free prompt optimization, several limitations remain.

Optimization Efficiency. The optimization process becomes less efficient as the trigger length increases, since the discrete token space grows exponentially with L. Longer triggers do not always yield higher accuracy, suggesting that Auto-BPO's search may struggle in high-dimensional spaces under limited model query budgets. Moreover, each of its three adopted strategies—greedy, evolutionary, and bandit-based search—entails trade-offs between local refinement, global exploration, and computational cost, which may restrict scalability for longer or more complex prompts.

Experimental and Evaluation Scope. Due to time and resource constraints, experiments were limited in scale and repetition, particularly for API-based models such as GPT-3.5 that incur high query costs. Evaluations mainly focused on task accuracy, without assessing robustness, interpretability, or transferability. Future work should extend experiments to larger datasets and broader metrics to better assess Auto-BPO's generalizability and reliability.

We leave these directions for future exploration to further enhance the scalability, efficiency, and applicability of automated black-box prompt optimization.

References

[1] Jiale Cheng, Xiao Liu, Kehan Zheng, Pei Ke, Hongning Wang, Yuxiao Dong, Jie Tang, and Minlie Huang. 2023. Black-box prompt optimization: Aligning large language models without model training. *arXiv preprint arXiv:2311.04155* (2023).

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

[3] Shizhe Diao, Zhichao Huang, Ruijia Xu, Xuechun Li, Yong Lin, Xiao Zhou, and Tong Zhang. 2022. Black-box prompt learning for pre-trained language models. *arXiv preprint arXiv:2201.08531* (2022).

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

[5] Aikaterini-Lida Kalouli, Hai Hu, Alexander F Webb, Lawrence S Moss, and Valeria De Paiva. 2023. Curing the SICK and other NLI maladies. *Computational Linguistics* 49, 1 (2023), 199–243.

[6] Mengdi Li, Jiaye Lin, Xufeng Zhao, Wenhao Lu, Peilin Zhao, Stefan Wermter, and Di Wang. 2025. Curriculum-RLAIF: Curriculum Alignment with Reinforcement Learning from AI Feedback. *arXiv preprint arXiv:2505.20075* (2025). doi:10.48550/arXiv.2505.20075

[7] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys* 55, 9 (2023), 1–35.

[8] Houda Louatouate and Mohammed Zeriouh. 2025. Role-based Prompting Technique in Generative AI-Assisted Learning: A Student-Centered Quasi-Experimental Study. *Journal of Computer Science and Technology Studies* 7, 2 (2025), 130–145.

[9] Rishabh Misra. 2018. News Headlines Dataset for Sarcasm Detection. https://github.com/rmisra/news-headlines-dataset-for-sarcasm-detection. Dataset.

[10] Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2022. Grips: Gradient-free, edit-based instruction search for prompting large language models. *arXiv* arXiv:2203.07281 (2022). https://arxiv.org/abs/2203.07281 Preprint.

[11] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. *arXiv preprint arXiv:2305.18290* (2023). doi:10.48550/arXiv.2305.18290 Version 3, July 2024.

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

[13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. 1631–1642.

[14] Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning*. PMLR, 20841–20855.

[15] Laura Villa, David Carneros-Prado, Adrián Sánchez-Miguel, Cosmin C Dobrescu, and Ramón Hervás. 2023. Conversational agent development through large language models: Approach with gpt. In *International Conference on Ubiquitous Computing and Ambient Intelligence*. Springer, 286–297.

[16] Doina Precup Volodymyr Kuleshov. 2000. Algorithms for the multi-armed bandit problem. *Journal of Machine Learning Research* (2000).

[17] Shuhe Wang, Shengyu Zhang, Jie Zhang, Runyi Hu, Xiaoya Li, Tianwei Zhang, Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy. 2024. Reinforcement learning enhanced llms: A survey. *arXiv preprint arXiv:2412.10400* (2024).

[18] Albert Webson and Ellie Pavlick. 2022. Do prompt-based models really understand the meaning of their prompts?. In *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies*. 2300–2344.

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

[20] Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. 2024. Search-in-the-chain: Interactively enhancing large language models with search for knowledge-intensive tasks. In *Proceedings of the ACM Web Conference 2024*. 1362–1373.

[21] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. 2024. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data* 18, 6 (2024), 1–32.

[22] Lang Yu, Qin Chen, Jiaju Lin, and Liang He. 2023. Black-box Prompt Tuning for Vision-Language Model as a Service.. In *IJCAI*. 1686–1694.

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

[24] Yuanhang Zheng, Zhixing Tan, Peng Li, and Yang Liu. 2024. Black-box prompt tuning with subspace learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 32 (2024), 3002–3013.

[25] Wenxuan Zhou, Ravi Agrawal, Shujian Zhang, Sathish Reddy Indurthi, Sanqiang Zhao, Kaiqiang Song, Silei Xu, and Chenguang Zhu. 2024. WPO: Enhancing RLHF with Weighted Preference Optimization. *arXiv preprint arXiv:2406.11827* (2024). doi:10.48550/arXiv.2406.11827 EMNLP 2024.

[26] Qipeng Zhu, Yanzhe Chen, Huasong Zhong, Yan Li, Jie Chen, Zhixin Zhang, Junping Zhang, and Zhenheng Yang. 2025. Uniapo: Unified multimodal automated prompt optimization. *arXiv preprint arXiv:2508.17890* (2025).

A

A.1 ACC results on three datasets using different LLMs

Table A.1.1. ACC results on three datasets using the GPT-2 model

SST-2

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.70	0.49	0.47
5	0.70	0.49	0.48
7	0.70	0.51	0.47
9	0.70	0.50	0.47

News Headlines

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.54	0.56	0.55
5	0.54	0.54	0.55
7	0.55	0.56	0.55
9	0.54	0.56	0.55

SICK

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.57	0.36	0.47
5	0.57	0.38	0.48
7	0.57	0.40	0.49
9	0.58	0.50	0.49

Table A.1.2. ACC results on three datasets using the GPT-3.5 model

SST-2

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.96	0.70	0.90
5	0.97	0.70	0.84
7	0.97	0.70	0.90
9	0.97	0.70	0.90

News Headlines

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.52	0.56	0.56
5	0.52	0.56	0.56
7	0.52	0.56	0.56
9	0.52	0.56	0.56

SICK

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.57	0.28	0.42
5	0.58	0.33	0.47
7	0.40	0.30	0.42
9	0.43	0.49	0.49

Table A.1.3. ACC results on three datasets using the LLAMA-3 model

SST-2

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.96	0.96	0.94
5	0.93	0.93	0.94
7	0.95	0.93	0.95
9	0.97	0.92	0.93

News Headlines

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.65	0.58	0.52
5	0.65	0.56	0.53
7	0.65	0.58	0.57
9	0.67	0.58	0.57

SICK

Trigger Length	Greedy Search	Evolutionary Search	Bandit Search
3	0.56	0.35	0.46
5	0.57	0.37	0.47
7	0.57	0.41	0.49
9	0.58	0.51	0.50