# Automatic Prompt Optimization via Heuristic Search: A Survey

**Wendi Cui**[1*], **Jiaxin Zhang**[1,2], **Zhuohang Li**[3], **Hao Sun**[4], **Damien Lopez**[1], **Kamalika Das**[1,2], **Bradley Malin**[3,5], **Sricharan Kumar**[1,2]

[1]Intuit [2]Intuit AI Research [3]Vanderbilt University [4] University of Cambridge
[5]Vanderbilt University Medical Center

## Abstract

Recent advances in Large Language Models (LLMs) have led to remarkable achievements, making *prompt engineering* increasingly central to guiding model outputs. While manual methods (e.g., "chain-of-thought," "step-by-step" prompts) can be effective, they typically rely on intuition and do not *automatically* refine prompts. In contrast, **automatic prompt optimization** employing heuristic-based search algorithms can systematically explore and improve prompts with minimal human oversight. This survey proposes a **comprehensive taxonomy** of these methods, categorizing them by *where* optimization occurs, *what* is optimized, *what criteria* drive the optimization, *which* operators generate new prompts, and *which* iterative search algorithms are applied. We further highlight specialized **datasets and tools** that support and accelerate automated prompt refinement. We conclude by discussing key **open challenges** for future opportunities for more robust and versatile LLM applications.

## 1 Introduction

The rapid evolution of **Large Language Models** (LLMs) has catalyzed significant progress in diverse tasks (Bubeck et al., 2023; Yang et al., 2023b). As these models become more capable, the *design of the prompt* used to interface with them has emerged as a crucial factor in terms of both prompt content and format (Zhu et al., 2023). Manual approaches such as *"chain-of-thought"* (Wei et al., 2023) prompting or instructing the model to *"Let's think step by step"* (Kojima et al., 2023) can yield enhanced performance in certain scenarios, but these methods remain fundamentally reliant on human intuition and repeated trial-and-error.

In contrast, **automatic prompt optimization** aims to systematically discover and refine prompts, minimizing human efforts and potentially uncovering highly effective solutions that manual experimentation might overlook (Zhou et al., 2023). These techniques treat prompt design as a *search problem*, wherein a **heuristic-based search algorithmic** process iteratively evaluates candidate prompts and adapts them based on performance feedback or other criteria (Pryzant et al., 2023; Chen et al., 2024a; Yang et al., 2023a). Other lines of work applies reinforcement learning (Zhang et al., 2022; Deng et al., 2022; Sun et al., 2023a) or ensemble methods (Hou et al., 2023; Pitis et al., 2023) to optimize prompts dynamically or adaptively. This survey focuses on heuristic-based search methods due to their ability to unify a broad range of strategies under an interpretable, modular framework that is compatible with both discrete and soft prompt spaces. We further concentrate on **instruction-focused** approaches, emphasizing the clarity and structure of the instruction while *optionally* incorporating in-context examples. Instruction-based prompting continues to be the dominant paradigm, making it a practical and impactful focus. By narrowing our scope to this intersection, we aim to provide a coherent taxonomy and detailed analysis of methods that are both theoretically grounded and widely applicable.

We begin by examining the fundamental dimensions of *where* optimization happens (optimization space) and *what* is optimized (optimization target). We then review *what criteria* to optimize (optimization criteria), recognizing that many practitioners now consider objectives beyond task performance. Then, we dive into how optimization happens by categorizing *which operators* are used to create new prompt candidates and *which iterative algorithm* guides the refinement loop. With a growing body of literature addressing these topics, we also review *benchmarking datasets* covering a wide range of

*The corresponding GitHub repository for this paper will be updated at `https://github.com/jxzhangjhu/Awesome-LLM-Prompt-Optimization`. Correspondence to {wendi_cui, jiaxin_zhang}@intuit.com.



*Findings of the Association for Computational Linguistics: ACL 2025*, pages 22093–22111

domains. Moreover, we survey a range of *tools* that can automate or streamline prompt optimization workflows, enabling rapid iteration with less manual effort. We conclude by identifying open problems. Addressing these challenges will unlock reliable, adaptable, and ethical LLM applications.

# 2 Preliminary

## 2.1 Prompt Composition

Prompts generally contain two main components:

1. **Instruction:** The instruction is a human-readable statement describing the task or objective. An instruction establishes the *intent* and *context* for the model, guiding it toward the desired behavior for the particular task.

2. **Examples:** In-context examples show the model how to map specific inputs to outputs. These examples clarify the nature of the task and can improve model performance (Brown et al., 2020).

Prompts may include several examples or none. For this survey, we cover the **instruction-focused** automatic prompt optimization, excluding research on *example-focused* optimization.

## 2.2 Heuristic-based Search Algorithms

Heuristic-based search algorithms provide a practical framework for optimization problems such as automatic prompt optimization (Pryzant et al., 2023; Zhou et al., 2023; Fernando et al., 2024). Unlike brute-force methods that evaluate every possible variation, heuristic methods apply problem-specific knowledge or strategies to navigate the search space more efficiently (Blum and Roli, 2003; Talbi, 2009). Key steps in a heuristic-based approach typically involve:

1. **Initialization:** Generating one or more candidates (e.g., randomly, using partial domain knowledge, or by perturbing a known baseline).

2. **Evaluation:** Measuring the performance of each candidate with respect to a chosen metric (e.g., accuracy on a validation set, or a scoring function provided by a user).

3. **Selection and Update:** Applying operators or transitions to the current set of candidates to create new candidates with improved performance, diversity, or both.

4. **Termination:** Determining when to stop (e.g., after a fixed number of iterations or once performance converges).

# 3 Where does Optimization Happen?

Prompt optimization in LLMs can be broadly categorized into *soft prompt space optimization* and *discrete prompt space optimization*.

## 3.1 Soft Prompt Space Optimization

Soft prompt optimization operates in a continuous space, allowing for smooth adaptations with techniques such as *gradient-based optimization*. Different approaches vary in whether they utilize gradients and how they apply them.

**Gradient for Embeddings** Using gradients for optimizing prompt embeddings is a widely used approach (Li and Liang, 2021; Zhang et al., 2021; Sun et al., 2022b,a, 2024). Li et al. (2024a) uses gradient descent to optimize prompts leveraging a loss function tailored to improve prompts' generalization capability across diverse domains.

To improve efficiency, some research chooses to estimate gradients, rather than directly computing them. ZOPO (Hu et al., 2024) employs *Zeroth-Order Optimization* to refine prompts without explicit gradient computation. It enables gradient estimation by using the Neural Tangent Kernel (Jacot et al., 2018) in a derived Gaussian process, approximating optimization dynamics in neural networks.

**Gradient-Based Target Selection** Another line of research utilizes gradients to identify target tokens to replace within a prompt candidate (Zhou et al., 2024; Zou et al., 2023; Zhao et al., 2024). Greedy Coordinate Gradient (GCG) (Zou et al., 2023) leverages gradient information to detect tokens that can minimize objective loss. It computes the gradient of the loss function with respect to the vector of each token, selecting the top-k tokens with the highest gradient values for modification. Zhao et al. (2024) enhance GCG with *Probe Sampling*, which accelerates prompt optimization by using two models: a smaller, faster "draft model" that assesses potential token replacement candidates and filters out unpromising ones; and a larger, more powerful "target model" which takes the filtered candidates for a full evaluation. Probe Sampling dynamically adjusts how many candidates are filtered by measuring the agreement between the draft and target models' rankings of a small "probe set" of candidates. This adaptive filtering minimizes unnecessary computation for the large target model, leading to substantial speed gains.

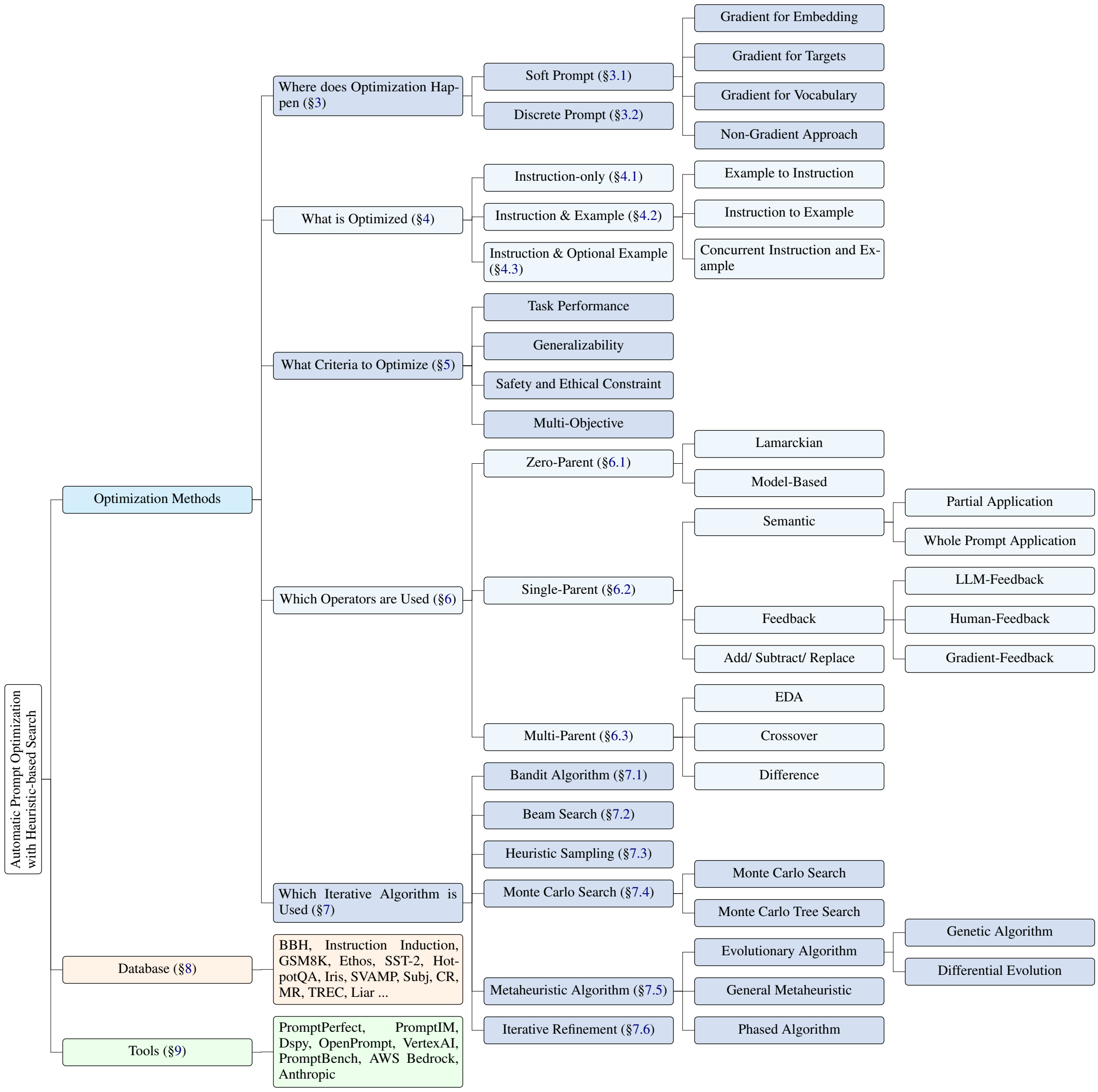


Figure 1: Taxonomy of Heuristic-based Search Algorithm in Automatic Prompt Optimization. Additional mapping of methods to taxonomy can be seen in Appendix Section A.

**Gradient for Vocabulary** In DPO (Wang et al., 2024b), gradients are estimated by a *Shortcut Text Gradient* approach that continuously relaxes the categorical word choices to a learnable smooth distribution over the vocabulary using Gumbel Softmax trick. This allows for the computation of gradients through the non-differentiable embedding lookup table. By ultimately learning a distribution over the vocabulary, DPO iteratively improves the quality of the generated prompts.

**Non-Gradient Approach** Other works optimize in soft prompt space but do not employ a gradient-based approach. InstructZero (Chen et al., 2024a) employs *Bayesian Optimization* to adjust prompt representations and propose new soft prompts without calculating gradients.

### 3.2 Discrete Prompt Space Optimization

Discrete prompt optimization treats prompts as fixed textual structures and refines them directly (Diao et al., 2022; Prasad et al., 2023). Unlike soft prompt methods, which adjust prompts in a continuous space, discrete methods optimize prompts in a non-differentiable space.

While soft prompt methods leverage *gradient*-based optimization, discrete approaches have developed *gradient-like* strategies suited for non-differentiable settings. ProTeGi (Pryzant et al., 2023) employs an LLM-based feedback system to generate pseudo-gradients and utilizes beam search to iteratively refine prompts, effectively mimicking the refinement process in gradient-based methods.

Beyond *gradient-like* approaches, other methods explore alternative optimization strategies. Evo-

Prompt (Guo et al., 2024) integrates evolutionary algorithms to iteratively refine prompts, employing semantic modification, crossover, and difference mechanisms for optimization. Cheng et al. (2024) trains a prompt-optimize model to rewrite prompts.

Overall, while soft prompt optimization excels in flexible and differentiable adjustments, discrete prompt optimization remains crucial for structured modifications where interpretability and explicit textual refinement are necessary.

# 4 What is Optimized?

## 4.1 Instruction-only Optimization

Early approaches primarily focused on refining the instruction itself through techniques such as rephrasing, adding constraints, or incorporating additional context (Yang et al., 2023a; Hsieh et al., 2024; Pan et al., 2024). *After* instruction optimization, some approaches introduce examples randomly to enhance task performance (Zhou et al., 2023; Pryzant et al., 2023). However, this choice often overlooks the interaction between added examples and instructions, resulting in suboptimal solutions (Wang et al., 2024a; Wan et al., 2024).

## 4.2 Instruction & Example Optimization

Recent research has increasingly focused on optimizing both instructions and examples. Existing approaches can be classified into three paradigms:

**Example to Instruction** This approach begins by selecting and preprocessing examples, which are then used to generate an appropriate instruction. MoP (Wang et al., 2024a) adopts this strategy by clustering examples into *Expert Subregions* and deriving specialized instructions for each cluster. This ensures that the instructions are well-aligned with the examples, improving task adaptation and overall effectiveness.

**Instruction to Example** In this approach, an initial instruction is used to generate examples that align with the intended task. MIPRO (Opsahl-Ong et al., 2024) follows this methodology by employing a default instruction to create successful input-output pairs as examples. This ensures that the examples complement the instruction and reinforce the model's understanding of the task.

**Concurrent Instruction and Example** This category focuses on optimizing both instructions and examples simultaneously. EASE (Wu et al., 2024) prioritizes selecting the best combination of instruction and examples from a pool of *pre-defined candidates*, using bandit algorithms to identify the most effective prompt structure. Whereas Adv-ICL (Long et al., 2024), dynamically generates *new instructions and examples*, refining both components iteratively with three models. By optimizing both elements in tandem, these methods ensure that instructions and examples are mutually reinforcing.

## 4.3 Instruction & Optional Example

Unlike prior approaches that strictly generate examples with instructions, PhaseEvo (Cui et al., 2024b) introduces a flexible framework capable of generating both few-shot and zero-shot prompts depending on what works best for the task. This adaptability allows the model to optimize performance dynamically, selecting whether examples are necessary based on empirical effectiveness.

# 5 What Criteria to Optimize

In heuristic-based automatic prompt optimization, the *objectives* or *criteria* for refinement vary widely depending on the application domain. While early research predominantly focused on *task performance*, growing interest in real-world deployments has led to broader optimization goals:

1. **Task Performance** Most approaches prioritize task-specific metrics and optimize prompts to enhance performance on the given task. Some approaches use a *validation set* to evaluate candidates (Guo et al., 2024), while others use a *surrogate model* of the objective function for candidate evaluation and selection (Opsahl-Ong et al., 2024; Chen et al., 2024b).

2. **Generalizability** Some methodologies extend beyond single-task performance, seeking prompts that generalize across multiple domains. Li et al. (2024a) introduce a *Concentrate-focused framework* to improve the domain generalizability of prompts by leveraging internal information from deep model layers. Their findings indicate that *prompts receiving higher attention from deep layers tend to generalize better* and that *prompts with stable attention distributions enhance generalization*. Their approach optimizes for generalizability in both soft and discrete spaces.

3. **Safety and Ethical Constraints** Ensuring safety is a critical aspect of prompt optimization for large language models. Studies such as *RPO*

(Zhou et al., 2024) emphasize the importance of designing prompt suffixes that resist adversarial manipulations and mitigate unintended behaviors. These safeguards are essential for defending against jailbreaking attempts.

4. **Multi-Objective Optimization** Multi-objective optimization plays a crucial role in balancing different priorities such as accuracy, efficiency, and safety. *SOS* (Sinha et al., 2024) employs an *interleaved* multi-objective evolutionary algorithms to optimize both task performance and safety where one objective is optimized first followed by another. In contrast, other approaches adopt a *parallel* optimization strategy of all objectives, followed by Pareto Optimization to derive the most effective prompts across multiple objectives (Menchaca Resendiz and Klinger, 2025; Yang and Li, 2023; Baumann and Kramer, 2024).

## 6 Which Operators are Used

For iterative optimization, generating new candidate prompts is essential. These generation methods, referred to as *operators*, are categorized based on the number of *parent prompts* needed. Parent prompts are existing prompts used to derive new candidates.

### 6.1 Zero-Parent Operators

**Lamarckian** The Lamarckian operator is an LLM-based operator that emulates the Lamarckian adaptation process, which refers to the idea of feeding back learned improvements (i.e., successful outputs or reasoning traces—phenotypes) into the prompt itself (genotype) to inform future generations. This mimics the Lamarckian notion of inheriting acquired traits (Fernando et al., 2024). By analyzing concrete inputs that yield correct outputs, it attempts to reverse-engineer the instruction prompt. This operator is widely adopted in research, especially during initialization (Zhou et al., 2023; Hu et al., 2024; Wu et al., 2024; Fernando et al., 2024; Wang et al., 2024a). MIPRO (Opsahl-Ong et al., 2024) extends this concept further by incorporating additional contextual information, such as identifying patterns within raw datasets, which enables LLMs to better comprehend task intentions. Table 1 presents an example of a Lamarckian operator.

| |
|---|
| I gave a friend an instruction and some input.<br>The friend read the instruction and wrote an<br>output for every one of the inputs.<br>Here are the input-output pairs:<br><br>## Example ##<br>{*input output pairs*}<br><br>The instruction was: |

Table 1: Lamarckian Operator Prompt Example

**Model-Based** This approach uses models to generate the next candidate. Chen et al. (2024a) and Sabbatella et al. (2023) build probabilistic models of prompt performance using Bayesian Optimization. MIPRO (Opsahl-Ong et al., 2024) uses a Tree-structured Parzen estimator to build a surrogate model to select the instruction and example pair. INSTINCT (Lin et al., 2024b) uses a trained neural network for score prediction. Although such models do not take any parent prompts directly, they do tap into learning from previous prompts to enhance their performance.

### 6.2 Single-Parent Operators

**Semantic** These operators generate candidates that share semantic meaning with their parent, either through the use of LLMs or alternative methods. Semantic operators can be categorized into:

- **Partial Application:** Semantic operators selectively modify specific sections of a prompt while maintaining the overall structure. This targeted approach enables precise adjustments, allowing for fine-tuning of particular aspects without altering the entire prompt. In AELP (Hsieh et al., 2024) and SCULPT (Kumar et al., 2024), partial application of LLM-based semantic operators is utilized to systematically adjust key components of extensive prompts.
- **Whole Prompt Application:** Other semantic operators apply transformations to the entire prompt. PhaseEvo (Cui et al., 2024b) uses an LLM-based semantic operator to perform the *last-mile* optimization in a phased optimization process. FIPO (Lu et al., 2025) finetunes a model to perform such rewriting. Xu et al. (2022) translates a candidate to another language and back as a way to create new candidates. For soft prompts, Shen et al. (2023) uses a perturbation kernel to perturb the sampled candidate embeddings from the previous iteration to create new prompts.

**Feedback** These methods utilize feedback from various sources to optimize prompts. They usually involve two steps: feedback generation and feedback application. Such feedback in a soft prompt space can be considered as the gradient. Based on the feedback generation process, such operators can be categorized as:

- **LLM-Feedback:** These operators leverage LLMs to evaluate and refine prompts. Such an approach taps into LLMs' self-reflection (Shinn et al., 2023) ability to pinpoint deficiencies and suggest refinements, facilitating the automated creation of more robust and effective prompts through continuous self-improvement (Cheng et al., 2024; Dong et al., 2024; Ye et al., 2024).
- **Human-Feedback:** Human feedback plays a crucial role in prompt optimization, providing nuanced, context-aware evaluations that automated systems may miss. PROMPST (Chen et al., 2024b) integrates human-designed rules to offer corrective feedback when errors arise, ensuring more precise refinements. APOHF (Lin et al., 2024a) leverages human preferences as an indicator for selecting the most effective prompts.
- **Gradient-Feedback:** Gradient-feedback involves using optimization techniques that adjust prompts based on gradient-based signals derived from the model's performance metrics. This approach is particularly effective for soft-prompt optimization and allows for precise and efficient adjustments (Wang et al., 2024b; Sordoni et al., 2023; Zhou et al., 2024).

**Add/Subtract/Replace** These operators refine prompts by inserting, removing, or substituting elements (Prasad et al., 2023; Juneja et al., 2024; Zhang et al., 2024d). DPO (Wang et al., 2024b) models each word as a genotype and applies these operators within an evolutionary algorithm framework for end-to-end prompt optimization.

### 6.3 Multiple-Parent Operators

**EDA** These operators generate new candidates from multiple parent prompts. OPRO (Yang et al., 2023a) adds both parent prompts and their performance on the validation set as additional information. IPO (Du et al., 2024) applies a similar strategy but on multimodal tasks. LCP (Li et al., 2024b) introduces contrastive examples to boost performance. Table 2 is an example of an EDA operator.

**Crossover** This operator follows genetic algorithms and combines components from two-parent prompts to create new prompts (Baumann and Kramer, 2024; Yang and Li, 2023; Jin et al., 2024b; Guo et al., 2024). The idea is to mix traits of both parents to compose a diverse candidate.

| |
|---|
| You are a mutator. Given a series of prompts, your task is to generate another prompt with the same semantic meaning and intentions.<br><br>## Existing Prompts ## |
| {*existing prompt*} |
| The newly mutated prompt is: |

Table 2: EDA Prompt Example

**Difference** These LLM operators analyze differences between prompts to identify patterns for generating new candidates. EvoPrompt-DE (Guo et al., 2024) uses the Differential Evolution algorithm with such operators.

## 7 Which Iterative Algorithm is Used

Iterative algorithms are crucial in automatic prompt optimization. They guide the selection and application of operators to refine prompts effectively.

### 7.1 Bandit Algorithms

Bandit algorithms are a class of decision-making strategies designed to balance the exploration-exploitation trade-off. Wu et al. (2024) formulates automatic prompt selection as a bandit problem, employing a neural bandit algorithm to predict the effectiveness of different sets of exemplars based on their embeddings. Similarly, Shi et al. (2024) introduces the BAI-FB framework, which efficiently explores and identifies the optimal prompt from a candidate pool while operating under a constrained budget. These approaches demonstrate the effectiveness of bandit-based methods in refining prompt selection and improving overall model performance.

### 7.2 Beam Search

The Beam Search method systematically expands a set of promising prompt candidates and prunes less effective ones, allowing efficient exploration of large search spaces. ERM (Yan et al., 2024), ProTeGi (Pryzant et al., 2023) use beam search to iteratively select candidates for optimization.

### 7.3 Heuristic Sampling

Heuristic sampling is a method that utilizes rule-based strategies to efficiently select representatives from large sets of candidates minimizing computational resources while maintaining high accuracy.

Chen et al. (2024b) employs heuristic-based sampling to prioritize the most promising prompts from an extensive pool, guided by human feedback to ensure their relevance and effectiveness.

### 7.4 Monte Carlo Search

**Monte Carlo Search** Monte Carlo search is a probabilistic algorithm that uses random sampling to explore and evaluate possible actions or decisions within a given problem space, enabling the estimation of optimal strategies through repeated simulations. APE (Zhou et al., 2023) leverages the Monte Carlo search to enhance prompt engineering by systematically exploring a vast array of potential prompts and assessing their effectiveness.

**Monte Carlo Tree Seach** Monte Carlo Tree Search (MCTS) is a search algorithm that builds a search tree incrementally through a series of selection, expansion, simulation, and backpropagation steps. PromptAgent (Wang et al., 2024c) constructs a search tree where each node represents a potential prompt. By keeping a state-action value function that calculates the potential rewards from following the path, the system iteratively refines prompts to enhance performance.

### 7.5 Metaheuristic Algorithms

Metaheuristic algorithms are high-level, problem-independent optimization strategies designed to efficiently explore large and complex search spaces where exact methods are infeasible. Inspired by natural processes such as evolution, physical systems, or social behavior, these algorithms guide the search toward optimal or near-optimal solutions through iterative improvement, balancing exploration and exploitation (Talbi, 2009).

**Evolutionary Algorithms** Evolutionary algorithms are widely utilized in prompt optimization, drawing inspiration from natural selection to iteratively refine prompts (Li and Wu, 2023; Jin et al., 2024b). Two algorithms in this category are *Genetic Algorithm (GA)* and *Differential Evolution(DE)*: GA applies evolutionary principles, including mutation, selection, and crossover, to iteratively enhance prompts, ensuring gradual improvement across successive generations. DE generates new prompt candidates by utilizing the differences between existing solutions, promoting diversity while converging toward optimal solutions. EvoPrompt (Guo et al., 2024) systematically compares both across identical tasks, demonstrating that each algorithm excels in different scenarios.

**General Metaheuristic** Additional metaheuristic algorithms such as Hill Climbing, Simulated Annealing, Tabu Search, Harmony Search, and others following metaheuristic principles are widely adopted for automatic prompt optimization as well (Zhang et al., 2024a; Sun et al., 2023b; Yang et al., 2024; Jin et al., 2024a; Lin et al., 2024b; Gao et al., 2025; Tang et al., 2025). Pan et al. (2024) specifically conducted a comparison among them.

**Phased Algorithms** Cui et al. (2024b) proposes a phased algorithm adopting a metaheuristic framework to increase the efficiency of the optimization process. By creating four phases balancing exploration and exploitation, they achieve several magnitudes of efficiency improvements compared to other algorithms in terms of LLM calls.

### 7.6 Iterative Refinement

Iterative Refinement refers to the other algorithms that repeatedly use different operators to refine prompts. Gradient descent is a widely-adopted example (Hu et al., 2024; Wang et al., 2024b; Zhou et al., 2024).

## 8 Common Datasets Used

Considering the broad applicability of prompt optimization, a variety of databases across different domains were used, as shown in Table 4 in the Appendix. The two most common ones are:

- **BBH (Big-Bench Hard)** (Aarohi and bench authors, 2023): A subset of the broader Big-Bench project, BBH is designed to probe the limits of language models with especially challenging or nuanced tasks.
- **Instruction Induction** (Honovich et al., 2022): This dataset explicitly focuses on inferring new task instructions from examples, making it particularly relevant for evaluating *instruction-based* prompt optimization approaches.

## 9 Common Tools

Automatic prompt optimization tools aim to accelerate and streamline the optimization process. Below, we provide an overview of notable tools and their key characteristics. Table 3 gives a high-level overview of these tools.

| Tool | Optimization Space | Key Features | Open Source |
|---|---|---|---|
| PromptPerfect | Discrete | Web-based, optimized for user queries | No |
| PromptIM | Discrete | Iterative refine with human in the loop | Yes |
| Dspy (Khattab et al., 2024) | Discrete | Task decomposition and example bootstrap | Yes |
| OpenPrompt (Ding et al., 2021) | Soft/Discrete | Predefined templates for prompt learning | Yes |
| Vertex AI (Wan et al., 2024) | Discrete | Google Cloud-based optimization | No |
| PromptBench (Zhu et al., 2023) | Discrete | Test prompt robustness | Yes |
| AWS Bedrock | Discrete | Playground with evaluation and A/B testing | No |
| Anthropic Claude | Discrete | Interactive editor with live feedback | No |

Table 3: Comparison of automatic prompt optimization tools.

**PromptPerfect** PromptPerfect is a commercial platform that offers automated prompt optimization services. It allows users to input a prompt and target a specific LLM. The platform then uses its internal algorithms to refine and improve the prompt. It provides a user-friendly interface and is designed to be accessible even to users without deep technical expertise.

**PromptIM** PromptIM is an experimental open-source library. Given an initial prompt, a dataset, and evaluators, PromptIM iteratively refines the prompt using a meta prompt to suggest improvements based on evaluation scores. It integrates with LangSmith for data and prompt management and supports optional human feedback. Contrary to other tools, PromptIM prioritizes keeping humans in the loop throughout the optimization process.

**DSPy** DSPy is a framework developed by Stanford researchers that takes a more declarative approach to building complex LLM applications. Instead of directly writing prompts, users define the desired "program" as a series of declarative steps. DSPy implements MIPRO (Opsahl-Ong et al., 2024) and uses LLMs to generate and optimize the underlying prompts to fulfill the program's goals. This approach allows for more structured and modular development of LLM applications and facilitates prompt optimization as part of the program execution. DSPy emphasizes the decomposition of complex tasks into simpler sub-tasks and is widely used in research (Soylu et al., 2024).

**OpenPrompt** OpenPrompt (Ding et al., 2021) is an open-source toolkit specifically designed for prompt-learning. It provides a standardized framework for implementing and experimenting with various templating, verbalizing, and optimization strategies. OpenPrompt offers pre-defined templates for different prompting methods, such as prefix-tuning and P-tuning. Its combinability across different Pretrained LMs, task formats, and optimization methods makes it a valuable tool.

**Vertex AI Prompt Optimizer** Google Cloud's Vertex AI platform offers a Prompt Optimizer as part of its suite of tools. This service allows users to experiment with and optimize prompts following learning from Wan et al. (2024). Integrated within the Vertex AI ecosystem, the Prompt Optimizer benefits from Google's cloud infrastructure and provides a scalable solution for prompt optimization tasks.

**PromptBench** PromptBench (Zhu et al., 2023) is an open-source benchmark designed to evaluate the robustness of prompts under adversarial perturbations. Rather than introducing new task datasets, PromptBench applies systematic transformations—such as instruction negation or paraphrasing—to existing prompts across a variety of standard NLP datasets. It helps researchers assess how well prompt optimization methods preserve model performance under distributional shift.

**AWS Bedrock** AWS Bedrock includes a Prompt Engineering Playground within its cloud platform, allowing users to prototype and evaluate prompts across multiple foundation models. The interface supports A/B testing, real-time inference, and evaluation metrics. While not open-source, it provides a practical environment for optimizing and comparing prompt variants in production-ready workflows.

**Anthropic Claude Prompt Tools** Anthropic's Claude platform offers interactive tools for prompt optimization directly through its web interface. These tools provide live feedback, suggest rewrites, and support prompt experimentation tailored specifically to Claude models. While proprietary, they are useful for developers seeking to iteratively refine instructions with guidance grounded in Claude's internal safety and helpfulness principles.

## 10 Open Challenges

**Soft to Discrete Projection** Soft prompt spaces enable continuous optimization but often lack interoperability provided by discrete prompts. To address this, some methods project soft embeddings back into discrete space. Hu et al. (2024), Wen et al. (2023) and Zhao et al. (2024) adopt a pre-generated finite set of unique candidates, where gradient-updated embeddings are mapped back to the closest entry in this set. However, this approach heavily depends on the quality of the pre-generated set. Another approach leverages an open-source LLM as a converter to translate soft prompts into discrete instructions (Chen et al., 2024a; Lin et al., 2024b), offering a more flexible and adaptive solution. Further research in this area could enhance both optimization effectiveness and interpretability.

**Dynamic N-shot Selection** While *Instruction & Example* paradigms have shown significant improvements by jointly optimizing examples and instructions (Wan et al., 2024; Menchaca Resendiz and Klinger, 2025), recent findings indicate that few-shot prompting does not always enhance performance and can *"consistently degrades its performance"* (DeepSeek-AI et al., 2025). This highlights the necessity of *Instruction & Optional Example* paradigm which dynamically selects between few-shot and zero-shot prompting based on empirical effectiveness rather than a fixed strategy. Initial steps in this direction have been explored by Cui et al. (2024b), and future optimization approaches should emphasize adaptability, tailoring prompt structures to specific tasks for optimal performance.

**Concurrent Optimization** For complex tasks using LLMs, multiple agents might be involved (Zhang et al., 2024c; Cui et al., 2024a; Zhang et al., 2024b). Traditionally, humans will define the scope of each agent and optimize them separately. Recent research has introduced automated concurrent optimization, which optimizes multiple prompts concurrently. DLN-2 (Sordoni et al., 2023) allows concurrent optimization of two prompts by considering both LLMs as probabilistic layers in a network. It treats the first prompt's output as a latent variable. Using variational inference to approximate the latent variable with a simpler distribution, DLN-2 optimizes the Evidence Lower Bound to refine both prompts. MIPRO (Opsahl-Ong et al., 2024) extends this concept to multi-stage optimization, treating each stage as a module and using Bayesian Search to identify the best prompt combinations. These methods represent a shift towards concurrent prompt optimization, reducing human effort while improving adaptability for complex task scenarios.

**Additional Challenges** Beyond the challenges discussed earlier, several open issues remain critical. *Multi-objective optimization* continues to be a complex area, requiring methods that can balance performance, safety, generalizability, and efficiency simultaneously (Sinha et al., 2024; Menchaca Resendiz and Klinger, 2025; Yang and Li, 2023). Recent work explores Pareto-front approximations, but reliable aggregation of heterogeneous metrics remains unsolved. In addition, *Scalability across Domains and Tasks* is limited by overfitting to specialized datasets. General-purpose optimizers must learn transferable representations or search strategies applicable in diverse settings. Another area is *On-line fashion optimization*. Existing methods take thousands of API calls (Cui et al., 2024b) or require specific training (Hu et al., 2024), making it impractical for online optimization. Incremental update rules and memory-efficient surrogate models could empower near real-time regimes.

## 11 Conclusion

This survey has explored heuristic-based search algorithms for *Automatic Instruction-Focused Prompt Optimization*, organizing methods along five key dimensions: the optimization space, the optimization target, the optimization criteria, the operators, and the iterative algorithms. The goal was to allow **mixing and matching** components like a toolkit, enabling the design of new prompt optimization pipelines by combining different operators with various search or learning algorithms. To make this more concrete, one might think of our framework like a recipe book: operators are the ingredients (e.g., "add", "replace", "rephrase"), and the iterative algorithms are the cooking methods (e.g., "bake" with genetic algorithms, "slow simmer" with beam search). Different combinations yield different flavors—and innovations. We hope this survey could jump-start researchers in understanding the existing landscape and inspire new research on the practical application of LLMs.

## 12 Limitations

The work does not cover In Context Learning optimization or methods using reinforcement learning. We also focus on works after 2023 so previous work is not fully covered. Such space can be expanded for future works.

## Acknowledgments

This work includes contributions from Vanderbilt University researchers, supported by funding from Intuit.

## References

Srivastava Aarohi and BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Jill Baumann and Oliver Kramer. 2024. Evolutionary multi-objective optimization of large language model prompts for balancing sentiments. *arXiv preprint arXiv:2401.09862*.

Christian Blum and Andrea Roli. 2003. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR)*, 35(3):268–308.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. “language models are few-shot learners". In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Lichang Chen, Jiuhai Chen, Tom Goldstein, Heng Huang, and Tianyi Zhou. 2024a. InstructZero: Efficient instruction optimization for black-box large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 6503–6518. PMLR.

Yongchao Chen, Jacob Arkin, Yilun Hao, Yang Zhang, Nicholas Roy, and Chuchu Fan. 2024b. PRompt optimization in multi-step tasks (PROMST): Integrating human feedback and heuristic-based sampling. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3859–3920, Miami, Florida, USA. Association for Computational Linguistics.

Jiale Cheng, Xiao Liu, Kehan Zheng, Pei Ke, Hongning Wang, Yuxiao Dong, Jie Tang, and Minlie Huang. 2024. Black-box prompt optimization: Aligning large language models without model training. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3201–3219, Bangkok, Thailand. Association for Computational Linguistics.

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

Wendi Cui, Zhuohang Li, Damien Lopez, Kamalika Das, Bradley A. Malin, Sricharan Kumar, and Jiaxin Zhang. 2024a. Divide-conquer-reasoning for consistency evaluation and automatic improvement of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 334–361, Miami, Florida, US. Association for Computational Linguistics.

Wendi Cui, Jiaxin Zhang, Zhuohang Li, Hao Sun, Damien Lopez, Kamalika Das, Bradley Malin, and Sricharan Kumar. 2024b. Phaseevo: Towards unified in-context prompt optimization for large language models. *arXiv preprint arXiv:2402.11347*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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*.

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*.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Hai-Tao Zheng, and Maosong Sun. 2021. Openprompt: An open-source framework for prompt-learning. *arXiv preprint arXiv:2111.01998*.

Yihong Dong, Kangcheng Luo, Xue Jiang, Zhi Jin, and Ge Li. 2024. PACE: Improving prompt with actor-critic editing for large language model. In *Findings of the Association for Computational Linguistics: ACL*

*2024*, pages 7304–7323, Bangkok, Thailand. Association for Computational Linguistics.

Yingjun Du, Wenfang Sun, and Cees Snoek. 2024. Ipo: Interpretable prompt optimization for vision-language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 126725–126766. Curran Associates, Inc.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. 2024. Promptbreeder: self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Shuzheng Gao, Chaozheng Wang, Cuiyun Gao, Xiaoqian Jiao, Chun Yong Chong, Shan Gao, and Michael Lyu. 2025. The prompt alchemist: Automated llm-tailored prompt optimization for test case generation. *arXiv preprint arXiv:2501.01329*.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2024. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In *International Conference on Learning Representations (ICLR)*.

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

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

Cho-Jui Hsieh, Si Si, Felix Yu, and Inderjit Dhillon. 2024. Automatic engineering of long prompts. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10672–10685, Bangkok, Thailand. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Wenyang Hu, Yao Shu, Zongmin Yu, Zhaoxuan Wu, Xiaoqiang Lin, Zhongxiang Dai, See-Kiong Ng, and Bryan Kian Hsiang Low. 2024. Localized zeroth-order prompt optimization. In *Advances in Neural Information Processing Systems*, volume 37, pages 86309–86345. Curran Associates, Inc.

Arthur Jacot, Franck Gabriel, and Clément Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, pages 8571–8580.

Can Jin, Hongwu Peng, Shiyu Zhao, Zhenting Wang, Wujiang Xu, Ligong Han, Jiahui Zhao, Kai Zhong, Sanguthevar Rajasekaran, and Dimitris N. Metaxas. 2024a. Apeer: Automatic prompt engineering enhances large language model reranking. *arXiv preprint arXiv:2406.14449*.

Feihu Jin, Yifan Liu, and Ying Tan. 2024b. Zero-shot chain-of-thought reasoning guided by evolutionary algorithms in large language models. *arXiv preprint arXiv:2402.05376*.

Gurusha Juneja, Nagarajan Natarajan, Hua Li, Jian Jiao, and Amit Sharma. 2024. Task facet learning: A structured approach to prompt optimization. *arXiv preprint arXiv:2406.10504*.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. DSPy: Compiling declarative language model calls into self-improving pipelines. In *Proceedings of the Twelfth International Conference on Learning Representations*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Shanu Kumar, Akhila Yesantarao Venkata, Shubhanshu Khandelwal, Bishal Santra, Parag Agrawal, and Manish Gupta. 2024. Sculpt: Systematic tuning of long prompts. *arXiv preprint arXiv:2410.20788*.

Chengzhengxu Li, Xiaoming Liu, Zhaohan Zhang, Yichen Wang, Chen Liu, Yu Lan, and Chao Shen. 2024a. Concentrate attention: Towards domain-generalizable prompt optimization for language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 3391–3420. Curran Associates, Inc.

Mingqi Li, Karan Aggarwal, Yong Xie, Aitzaz Ahmad, and Stephen Lau. 2024b. Learning from contrastive prompts: Automated optimization and adaptation. *arXiv preprint arXiv:2409.15199*.

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

Xiaoyu Li, Wei Zhang, Jiahui Chen, and Hongwei Liu. 2024c. Iris: Benchmarking large language models for information retrieval tasks. *arXiv preprint arXiv:2401.12345*.

Yujian Betterest Li and Kai Wu. 2023. Spell: Semantic prompt evolution based on a llm. *arXiv preprint arXiv:2310.01260*.

Xiaoqiang Lin, Zhongxiang Dai, Arun Verma, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024a. Prompt optimization with human feedback. *arXiv preprint arXiv:2405.17346*.

Xiaoqiang Lin, Zhaoxuan Wu, Zhongxiang Dai, Wenyang Hu, Yao Shu, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024b. Use your instinct: Instruction optimization for llms using neural bandits coupled with transformers. *arXiv preprint arXiv:2310.02905*.

Xuan Do Long, Yiran Zhao, Hannah Brown, Yuxi Xie, James Xu Zhao, Nancy F. Chen, Kenji Kawaguchi, Michael Shieh, and Junxian He. 2024. Prompt optimization via adversarial in-context learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7308–7327, Bangkok, Thailand. Association for Computational Linguistics.

Junru Lu, Siyu An, Min Zhang, Yulan He, Di Yin, and Xing Sun. 2025. FIPO: Free-form instruction-oriented prompt optimization with preference dataset and modular fine-tuning schema. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 11029–11047, Abu Dhabi, UAE. Association for Computational Linguistics.

Yarik Menchaca Resendiz and Roman Klinger. 2025. MOPO: Multi-objective prompt optimization for affective text generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5588–5606, Abu Dhabi, UAE. Association for Computational Linguistics.

Ioannis Mollas, Zafeiria Chrysopoulou, Spiros Karlos, and Grigorios Tsoumakas. 2022. Ethos: an online hate speech detection dataset. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2022)*.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. 2024. Optimizing instructions and demonstrations for multi-stage language model programs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9340–9366, Miami, Florida, USA. Association for Computational Linguistics.

Rui Pan, Shuo Xing, Shizhe Diao, Wenhe Sun, Xiang Liu, KaShun Shum, Jipeng Zhang, Renjie Pi, and Tong Zhang. 2024. Plum: Prompt learning using metaheuristics. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2177–2197, Bangkok, Thailand. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*.

Rakesh Patel, Shyam Upadhyay, Debanjan Mahata, Yaman Kumar, and Rakesh Gosangi. 2021. Svamp: Evaluating mathematical reasoning of transformers on realistic problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

Silviu Pitis, Michael R Zhang, Andrew Wang, and Jimmy Ba. 2023. Boosted prompt ensembles for large language models. *arXiv preprint arXiv:2304.05970*.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3845–3864, Dubrovnik, Croatia. Association for Computational Linguistics.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968, Singapore. Association for Computational Linguistics.

Antonio Sabbatella, Andrea Ponti, Antonio Candelieri, Ilaria Giordani, and Francesco Archetti. 2023. A bayesian approach for prompt optimization in pre-trained language models. *arXiv preprint arXiv:2312.00471*.

Maohao Shen, Soumya Ghosh, Prasanna Sattigeri, Subhro Das, Yuheng Bu, and Gregory Wornell. 2023. Reliable gradient-free and likelihood-free prompt tuning. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 2416–2429, Dubrovnik, Croatia. Association for Computational Linguistics.

Chengshuai Shi, Kun Yang, Zihan Chen, Jundong Li, Jing Yang, and Cong Shen. 2024. Efficient prompt optimization through the lens of best arm identification. In *Advances in Neural Information Processing Systems*, volume 37, pages 99646–99685. Curran Associates, Inc.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36.

Ankita Sinha, Wendi Cui, Kamalika Das, and Jiaxin Zhang. 2024. Survival of the safest: Towards secure prompt optimization through interleaved multi-objective evolution. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1016–1027, Miami, Florida, US. Association for Computational Linguistics.

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*.

Alessandro Sordoni, Eric Yuan, Marc-Alexandre Côté, Matheus Pereira, Adam Trischler, Ziang Xiao, Arian Hosseini, Friederike Niedtner, and Nicolas Le Roux. 2023. Joint prompt optimization of stacked llms using variational inference. In *Advances in Neural Information Processing Systems*, volume 36, pages 58128–58151. Curran Associates, Inc.

Dilara Soylu, Christopher Potts, and Omar Khattab. 2024. Fine-tuning and prompt optimization: Two great steps that work better together. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10696–10710, Miami, Florida, USA. Association for Computational Linguistics.

Hao Sun, Alihan Hüyük, and Mihaela van der Schaar. 2023a. Query-dependent prompt evaluation and optimization with offline inverse rl. *arXiv e-prints*, pages arXiv–2309.

Hong Sun, Xue Li, Yinchuan Xu, Youkow Homma, Qi Cao, Min Wu, Jian Jiao, and Denis Charles. 2023b. Autohint: Automatic prompt optimization with hint generation. *arXiv preprint arXiv:2307.07415*.

Tianxiang Sun, Zhengfu He, Hong Qian, Yunhua Zhou, Xuan-Jing Huang, and Xipeng Qiu. 2022a. Bbtv2: towards a gradient-free future with large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3916–3930.

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

Yifan Sun, Jean-Baptiste Tien, and Karthik Lakshmanan. 2024. Retrieval augmented prompt optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

El-Ghazali Talbi. 2009. *Metaheuristics: From Design to Implementation*. John Wiley & Sons, Hoboken, NJ.

Xinyu Tang, Xiaolei Wang, Wayne Xin Zhao, Siyuan Lu, Yaliang Li, and Ji-Rong Wen. 2025. Unleashing the potential of large language models as prompt optimizers: Analogical analysis with gradient-based model optimizers. *arXiv preprint arXiv:2402.17564*.

Ellen M Voorhees and Dawn M Tice. 2000. The trec-8 question answering track report. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*.

Xingchen Wan, Ruoxi Sun, Hootan Nakhost, and Sercan O. Arik. 2024. Teach better or show smarter? on instructions and exemplars in automatic prompt optimization. *arXiv preprint arXiv:2406.15708*.

Ruochen Wang, Sohyun An, Minhao Cheng, Tianyi Zhou, Sung Ju Hwang, and Cho-Jui Hsieh. 2024a. One prompt is not enough: automated construction of a mixture-of-expert prompts. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Ruochen Wang, Ting Liu, Cho-Jui Hsieh, and Boqing Gong. 2024b. On discrete prompt optimization for diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 50992–51011. PMLR.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*.

Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P. Xing, and Zhiting Hu. 2024c. Promptagent: Strategic planning with language models enables expert-level prompt optimization. In *International Conference on Learning Representations (ICLR)*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. In *Advances in Neural Information Processing Systems*, volume 36, pages 51008–51025. Curran Associates, Inc.

Zhaoxuan Wu, Xiaoqiang Lin, Zhongxiang Dai, Wenyang Hu, Yao Shu, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024. Prompt optimization with ease? efficient ordering-aware automated selection of exemplars. In *Advances in Neural Information Processing Systems*, volume 37, pages 122706–122740. Curran Associates, Inc.

Hanwei Xu, Yujun Chen, Yulun Du, Nan Shao, Wang Yanggang, Haiyu Li, and Zhilin Yang. 2022. GPS: Genetic prompt search for efficient few-shot learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8162–8171, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Cilin Yan, Jingyun Wang, Lin Zhang, Ruihui Zhao, Xiaopu Wu, Kai Xiong, Qingsong Liu, Guoliang Kang, and Yangyang Kang. 2024. Efficient and accurate prompt optimization: the benefit of memory in exemplar-guided reflection. *arXiv preprint arXiv:2411.07446*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2023a. "challenging big-bench tasks and whether chain-of-thought can solve them". *arXiv preprint arXiv:2309.03409*.

Heng Yang and Ke Li. 2023. InstOptima: Evolutionary multi-objective instruction optimization via large language model-based instruction operators. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13593–13602, Singapore. Association for Computational Linguistics.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. 2023b. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712*.

Sheng Yang, Yurong Wu, Yan Gao, Zineng Zhou, Bin Benjamin Zhu, Xiaodi Sun, Jian-Guang Lou, Zhiming Ding, Anbang Hu, Yuan Fang, Yunsong Li, Junyan Chen, and Linjun Yang. 2024. AMPO: Automatic multi-branched prompt optimization. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20267–20279, Miami, Florida, USA. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. 2024. Prompt engineering a prompt engineer. *arXiv preprint arXiv:2311.05661*.

Chenrui Zhang, Lin Liu, Chuyuan Wang, Xiao Sun, Hongyu Wang, Jinpeng Wang, and Mingchen Cai. 2024a. Prefer: prompt ensemble learning via feedback-reflect-refine. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

Jiaxin Zhang, Wendi Cui, Yiran Huang, Kamalika Das, and Sricharan Kumar. 2024b. Synthetic knowledge ingestion: Towards knowledge refinement and injection for enhancing large language models. *arXiv preprint arXiv:2410.09629*.

Jiaxin Zhang, Zhuohang Li, Kamalika Das, Bradley A. Malin, and Sricharan Kumar. 2024c. Sac3: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency. *arXiv preprint arXiv:2311.01740*.

Lechen Zhang, Tolga Ergen, Lajanugen Logeswaran, Moontae Lee, and David Jurgens. 2024d. Sprig: Improving large language model performance by system prompt optimization. *arXiv preprint arXiv:2410.14826*.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable prompt makes pre-trained language models better few-shot learners. *arXiv preprint arXiv:2108.13161*.

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*.

Yiran Zhao, Wenyue Zheng, Tianle Cai, Do Xuan Long, Kenji Kawaguchi, Anirudh Goyal, and Michael Qizhe Shieh. 2024. Accelerating greedy coordinate gradient and general prompt optimization via probe sampling. In *Advances in Neural Information Processing Systems*, volume 37, pages 53710–53731. Curran Associates, Inc.

Andy Zhou, Bo Li, and Haohan Wang. 2024. Robust prompt optimization for defending language models against jailbreaking attacks. In *Advances in Neural Information Processing Systems*, volume 37, pages 40184–40211. Curran Associates, Inc.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, et al. 2023. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv preprint arXiv:2306.04528*.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

## A Methods Categorization based on Taxonomy

Below are the categorizations for methods surveyed in this paper based on the taxonomy.

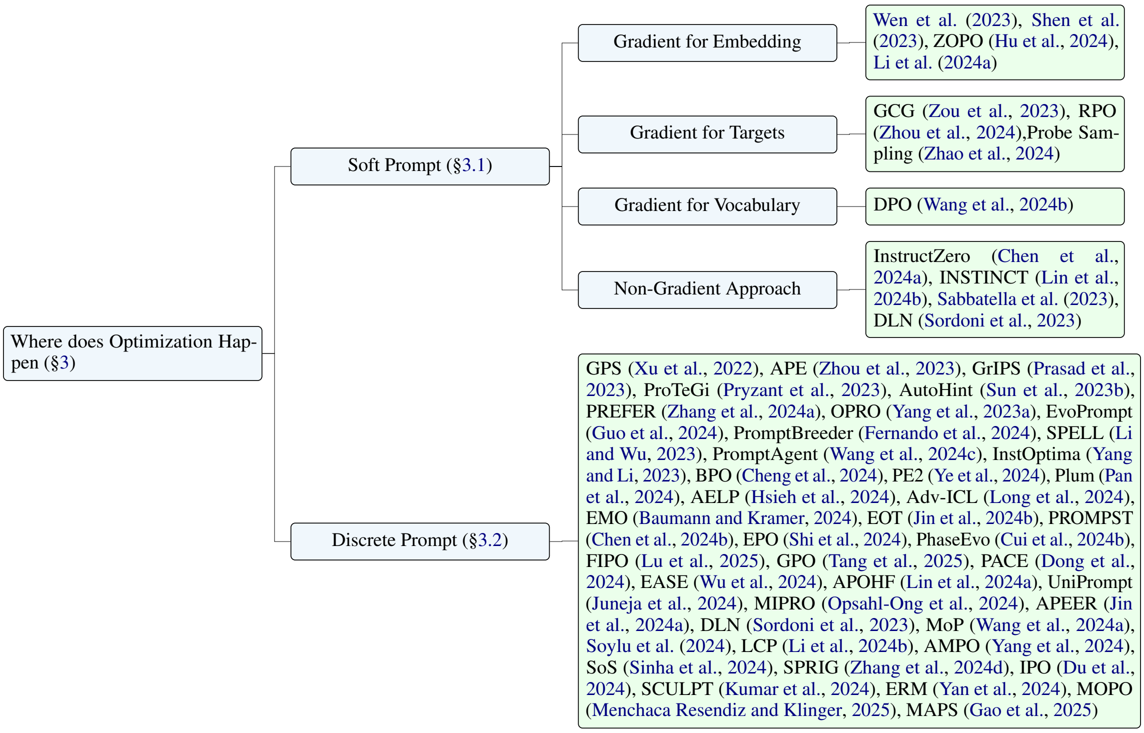


Figure 2: *"Where does optimization happen"* Categorization

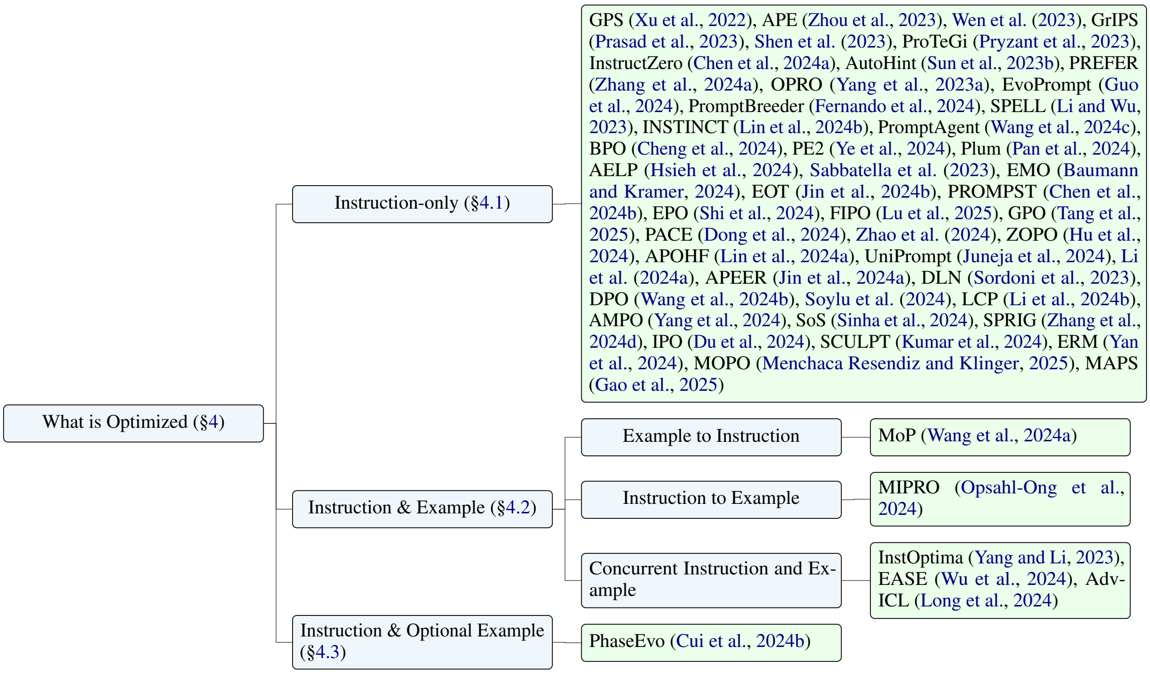


Figure 3: *"What is optimized"* Categorization

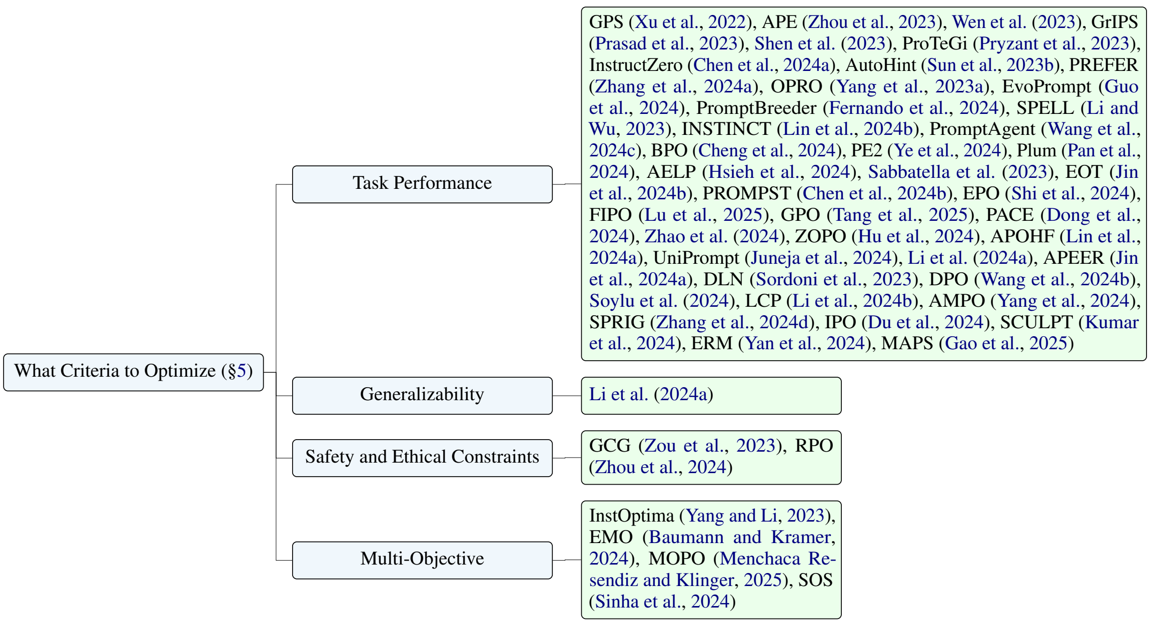


Figure 4: *"What criteria to optimize"* Categorization

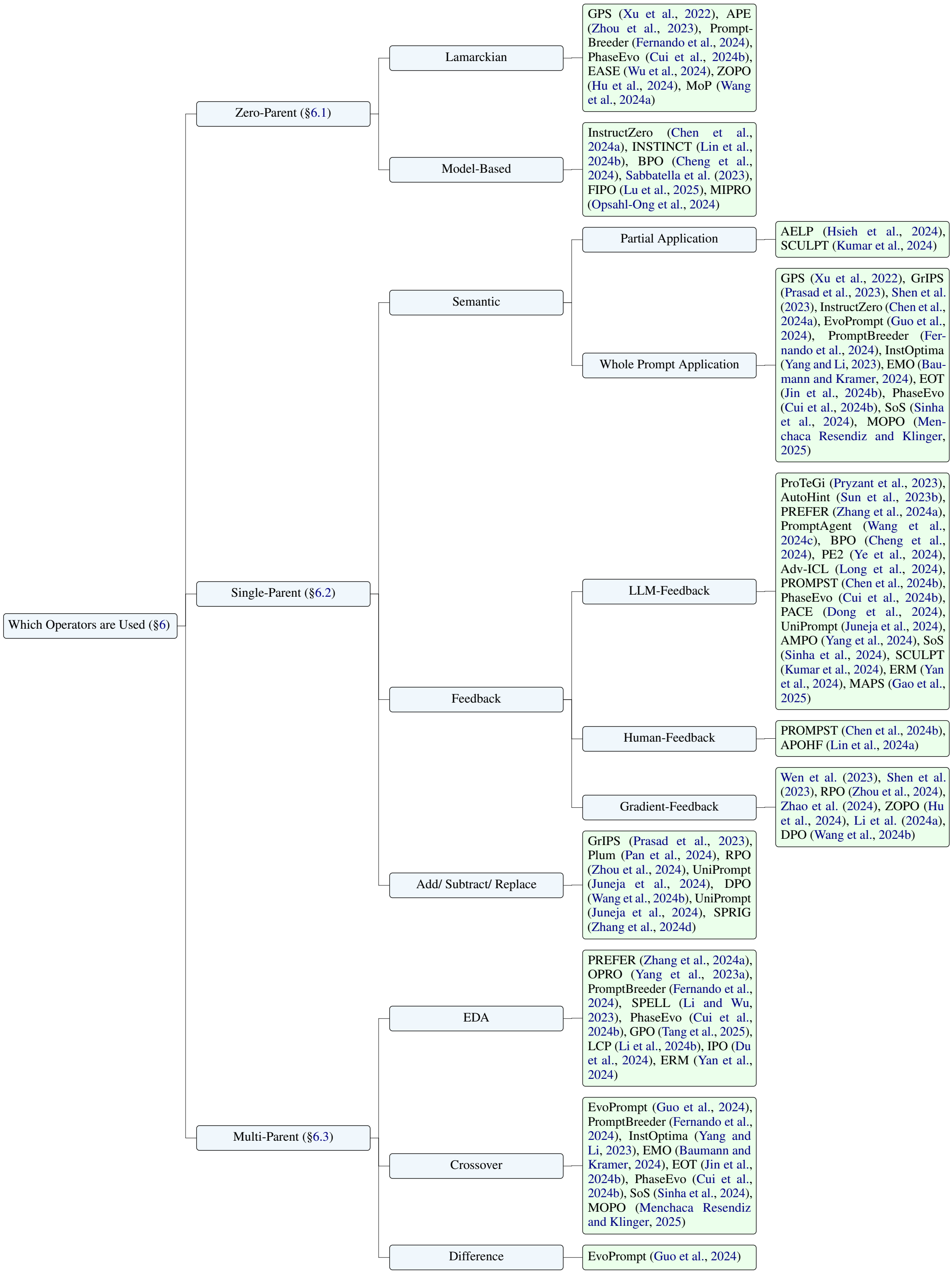


Figure 5: *"Which operators are used"* Categorization

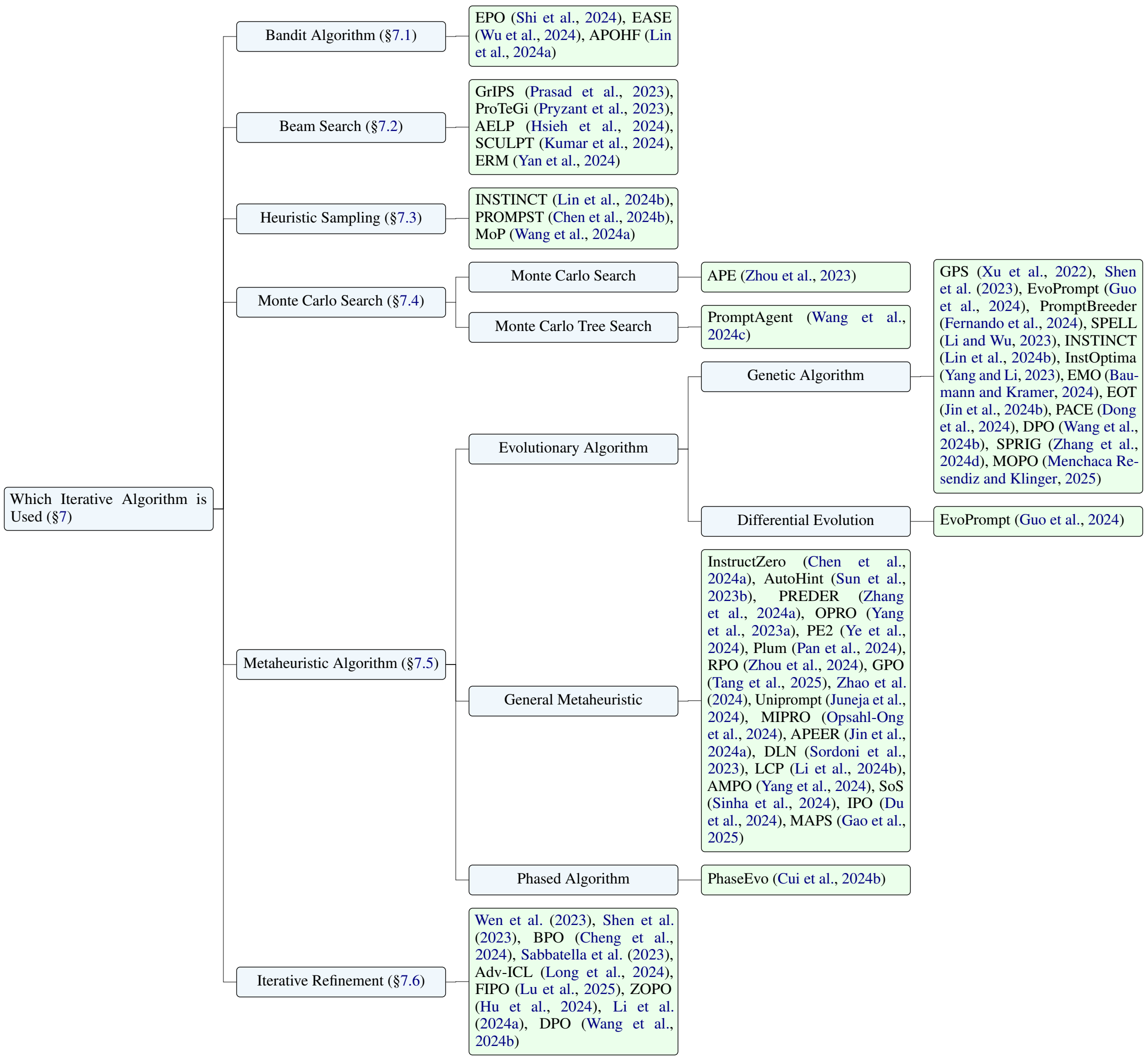


Figure 6: *"Which iterative algorithm is used"* Categorization

## B Datasets and Tools

| Dataset Name | Dataset Category |
|---|---|
| BBH (Aarohi and bench authors, 2023) | NLP Benchmark |
| Instruction Induction (Honovich et al., 2022) | NLP Benchmark |
| GSM8K (Cobbe et al., 2021) | Mathematical Reasoning |
| Ethos (Mollas et al., 2022) | Bias and Ethics Evaluation |
| SST-2 (Socher et al., 2013) | Sentiment Analysis |
| HotpotQA (Yang et al., 2018) | Question Answering |
| Iris (Li et al., 2024c) | Scientific Classification |
| SVAMP (Patel et al., 2021) | Mathematical Reasoning |
| Subj (Pang and Lee, 2004) | Subjectivity Detection |
| CR (Hu and Liu, 2004) | Sentiment Analysis |
| MR (Pang and Lee, 2005) | Sentiment Analysis |
| TREC (Voorhees and Tice, 2000) | Question Answering |
| Liar (Wang, 2017) | Misinformation Detection |

Table 4: Commonly used datasets