Review of Reinforcement Learning for Large Language Models: Formulations, Algorithms, and Opportunities

Anonymous authors
Paper under double-blind review

Abstract

Large Language Models (LLMs) represent significant milestones in artificial intelligence development. While pre-training on vast text corpora and subsequent supervised fine-tuning establish their core abilities, Reinforcement Learning (RL) has emerged as an indispensable paradigm for refining LLMs, particularly in aligning them with human values, and teaching them to reason and follow complex instructions. As this field evolves rapidly, this survey offers a systematic review of RL methods for LLMs, with a focus on fundamental concepts, formal problem settings, and the main algorithms adapted to this context. Our review critically examines the inherent computational and algorithmic challenges arising from the integration of RL with LLMs, such as scalability issues, effective gradient estimation, and training efficiency. Concurrently, we highlight exciting opportunities for advancing LLM capabilities through new RL strategies, including multi-modal integration and the development of agentic LLM systems.

1 Introduction

Large Language Models (LLMs) have experienced rapid development in recent years, transforming from academic research tools to powerful systems with widespread commercial applications. The evolution from early models like GPT-1 (Radford et al., 2018) to current state-of-the-art systems such as GPT-5 (OpenAI, 2025), Claude-4 (Anthropic, 2025), Gemini-2.5 (Comanici et al., 2025), Llama-4 (Meta, 2025), Qwen-3 (Yang et al., 2025a), and DeepSeek-R1 (Guo et al., 2025) not only reflects exponential growth in model size, capabilities, and adoption but also signals a new era for AI. This acceleration has been driven by several key factors: architectural innovations in transformer models (Vaswani et al., 2017; Li et al., 2025d), increasingly large and diverse training datasets (Brown et al., 2020), and significant advancements in computational resources dedicated to training (Kaplan et al., 2020; Hoffmann et al., 2022).

The development of modern LLMs typically involves several distinct phases: initial pre-training on vast text corpora (Radford et al., 2019; Brown et al., 2020), refinement through Supervised Fine-Tuning (SFT) on specialized datasets (Raffel et al., 2020), and crucially, alignment and reasoning enhancement through Reinforcement Learning (RL) techniques (Christiano et al., 2017; Ouyang et al., 2022; Guo et al., 2025). Each developmental stage addresses specific limitations and enhances particular model capabilities: pre-training establishes foundational knowledge and linguistic understanding; SFT teaches instruction following and appropriate response formatting; and RL is instrumental in boosting alignment, safety, and complex reasoning abilities. These post-training techniques are proving indispensable for strong downstream task performance.

The transformative power of post-training methodologies, particularly those involving RL, becomes evident through compelling empirical results. According to OpenAI's technical report (OpenAI, 2023), the GPT-4 base model only marginally outperformed GPT-3.5 before post-training optimization. However, after applying Reinforcement Learning from Human Feedback (RLHF), GPT-4 achieved a remarkable 30-point improvement on TruthfulQA (Lin et al., 2021)—a benchmark specifically designed to evaluate a model's ability to discern facts from adversarially-selected falsehoods. Similarly, on competition-level mathematics problems from AIME 2024, OpenAI's o1 model (OpenAI, 2024)—trained with specialized RL techniques to enhance reasoning capabilities—achieved an impressive 83.3% score. This marks an absolute improvement of approximately 70

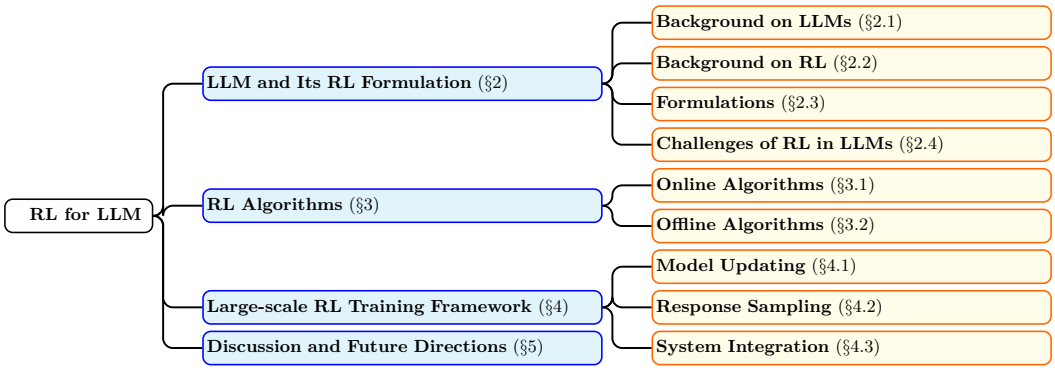


Figure 1: The organization of survey.

percentage points over their previous flagship, GPT-4o, which scored around 13.3% on the same problems. These dramatic performance improvements underscore the critical importance of post-training approaches, particularly those leveraging RL methodologies.

The application of RL in LLM development has expanded rapidly across diverse domains and objectives. Key areas include value alignment tasks that ensure models behave according to human preferences (Ouyang et al., 2022; Bai et al., 2022a;b), safety enhancement to reduce harmful outputs (Guan et al., 2024; Beutel et al., 2024), specialized domain optimization such as mathematical reasoning and code generation (Roziere et al., 2023; Shao et al., 2024; Hui et al., 2024), and emerging agentic capabilities that enable models to perform complex, multi-step tasks (Yao et al., 2023a; Yang et al., 2024b; Zheng et al., 2025c). Each application domain presents unique challenges in reward design, training stability, and evaluation metrics.

These expanding applications have driven significant innovations across multiple dimensions: new problem formulations that better capture the nuances of language-based tasks (Wei et al., 2022b; Rafailov et al., 2023), algorithmic advances that address the unique challenges of RL in high-dimensional spaces (Li et al., 2024c; Shao et al., 2024), and sophisticated training infrastructures capable of handling the computational demands of large-scale RL training (Yao et al., 2023b; Sheng et al., 2024). The rapid pace of development in this intersection has created a pressing need for systematic analysis and consolidation of emerging methodologies.

Driven by the rapid advancements in Reinforcement Learning (RL) and its pivotal role in enhancing modern Large Language Model (LLM) capabilities, this survey provides a systematic analysis of the intersection between RL and LLMs, with a particular focus on recent methodological advances. The organization of this survey is detailed in Figure 1:

- **Fundamental RL Concepts and Formulations:** We introduce core RL concepts (Sutton, 1988) and examine how they have been adapted for language model training. Special attention is given to the unique characteristics of RL in the LLM context, including reward modeling in text-based environments, challenges of estimating gradients, and performing gradient updates in the high-dimensional space.

- **Leading RL Training Algorithms**: We provide comprehensive analysis of prominent RL algorithms that have demonstrated success in LLM training, including PPO (Schulman et al., 2017), ReMax (Li et al., 2024c), and GRPO (Shao et al., 2024). Our examination highlights the connections and distinctions between these algorithms and conventional RL methods, adaptations required for language models' unique characteristics, and trade-offs between computational efficiency and learning effectiveness.

- **Distributed RL Training Frameworks:** We review existing infrastructures and methodologies for implementing RL at the scale required for modern LLMs, discussing memory-efficient training techniques, accelerated generation techniques (Kwon et al., 2023), strategies for distributed training and optimization (Sheng et al., 2024), and approaches to handle the computational challenges of large-scale RL deployment (Wu et al., 2025).

- **Future Directions and Open Challenges:** We outline promising research avenues at the RL-LLM intersection. This includes exploring necessary modifications to current methodologies, emerging techniques for more sample-efficient learning, strategies to address current limitations in scalability and stability, and advancing LLM capabilities through multi-modal integration and the development of agentic LLM systems.

To maintain focus, this survey deliberately excludes topics important to the broader field but not directly relevant to the methodological intersection of RL and LLMs. These include pre-training methodologies (for which readers are referred to (Albalak et al., 2024)), model explainability (Zhao et al., 2024), and concrete applications of LLMs in specific domains (Casper et al., 2023; Wang et al., 2025b). This survey aims to provide a comprehensive understanding of how RL techniques can enhance LLM capabilities, with a particular emphasis on problem formulation and algorithm design in terms of stability and efficiency.

2 LLM and Its RL Formulation

2.1 Background on LLMs

A Large Language Model (LLM) is a generative model that operates by predicting the next token in a sequence through probabilistic modeling. At its core, an LLM employs a transformer architecture (Vaswani et al., 2017) to capture complex patterns and dependencies in textual data. Formally, an LLM is denoted as π_θ, where $\theta \in \mathbb{R}^d$ represents the model's trainable parameters. The model selects tokens from a finite vocabulary $\mathcal{V} = \{1, \ldots, |\mathcal{V}|\}$, and given a context sequence $(a_1, \ldots, a_t)$, it models the conditional distribution of the next token as $a_{t+1} \sim \pi_\theta(\cdot|a_1, \ldots, a_t)$. This autoregressive generation continues until either a designated End-Of-Sentence (EOS) token appears or a predetermined maximum length T is reached.

Modern LLMs have undergone substantial scaling, with parameter counts ranging from several billion (e.g., Llama-3-8B (Grattafiori et al., 2024) and Qwen-2.5-7B (Yang et al., 2024a)) to hundreds of billions (e.g., GPT-4 (OpenAI, 2023) and PaLM (Chowdhery et al., 2023)). This dramatic increase in model size has been empirically shown to correlate with emergent capabilities—advanced functionalities that are not observed in smaller-scale models (Wei et al., 2022a; Srivastava et al., 2022). While scaling has yielded impressive advancements in model performance, it simultaneously introduces significant challenges in computational efficiency, optimization strategies, and resource requirements, which we will examine in greater detail in subsequent sections. Before delving into the core of this survey, we provide a concise overview of the LLM training pipeline, as illustrated in Figure 2.

Pre-training. Unlike traditional deep learning models (e.g., AlexNet (Krizhevsky et al., 2012) and ResNet (He et al., 2016)) that are typically trained from scratch for specific tasks, LLMs are *pre-trained* on extensive corpora, optimizing for next-token prediction (Radford et al., 2019; Brown et al., 2020; Raffel et al., 2020). This process exposes models to diverse internet-scale datasets comprising books, articles, code repositories, and web pages—often totaling trillions of tokens (Gao et al., 2020; Penedo et al., 2023). The self-supervised nature of next-token prediction enables models to capture complex patterns, linguistic structures, and factual knowledge without requiring explicit labels, making this approach both scalable and effective for learning rich representations from raw text.

From another view, pre-training inherently implements a form of multi-task learning (Caruana, 1997; Radford et al., 2019), whereby a single model learns to perform multiple related tasks simultaneously. In the context of LLMs, these "tasks" correspond to different domains of knowledge, reasoning patterns, and linguistic capabilities (Wei et al., 2021; Sanh et al., 2021). To minimze the prediction loss, the model must learn to predict tokens in contexts as varied as mathematical proofs, fictional narratives, technical documentation, and conversational dialogues. This multi-domain training enables the development of generalized capabilities that transfer across tasks (Bommasani et al., 2021; Rae et al., 2021).

Post-training. Despite impressive capabilities in language understanding, pre-trained LLMs often struggle to produce contextually appropriate responses when applied directly to downstream applications, frequently generating repetitive or irrelevant content. This stems from a fundamental domain shift: while pre-training

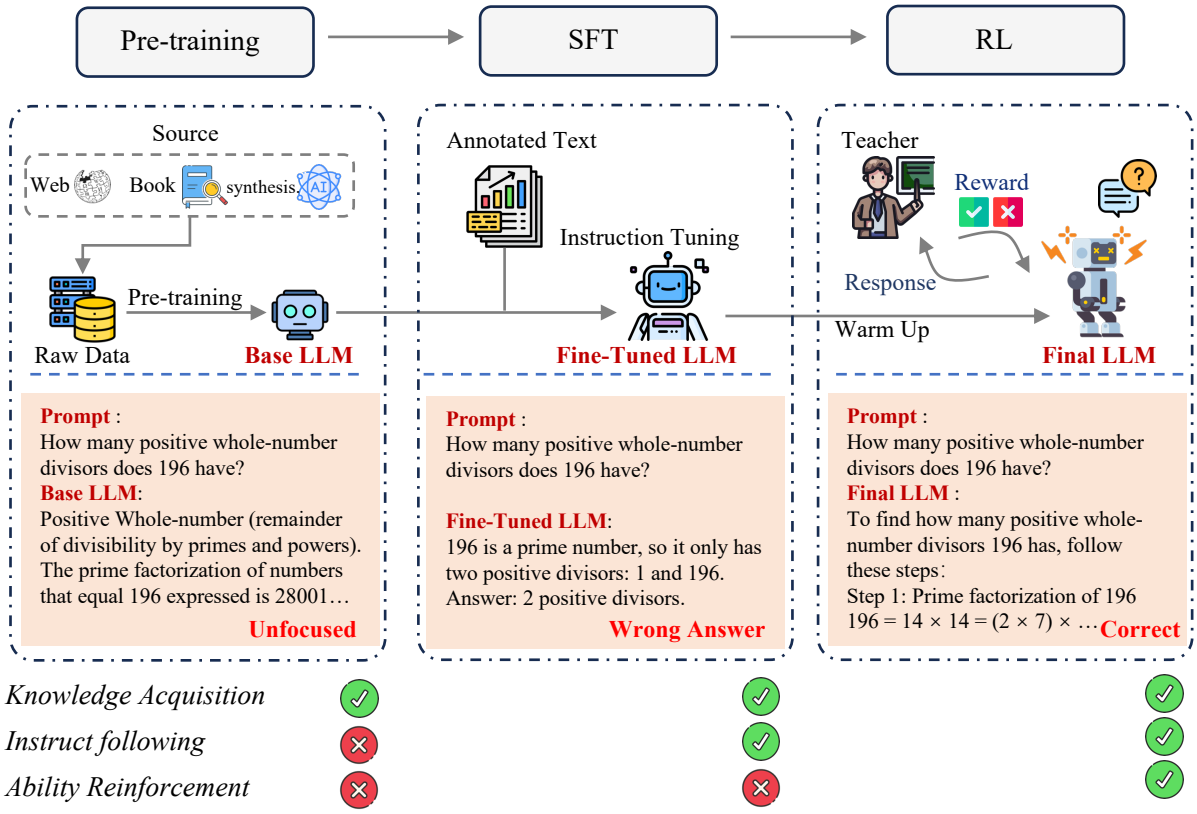


Figure 2: The three training stages of Large Language Models (LLMs): 1) Pre-training: A Base LLM is trained on vast raw data for next-token prediction, acquiring Knowledge Acquisition. Responses are typically unfocused and lack Instruction Following. 2) Supervised Fine-Tuning (SFT): The Base LLM is fine-tuned with instruction-response pairs, improving Instruction Following and structured output, though still prone to wrong answers in complex tasks. 3) Reinforcement Learning (RL): Optimized with human or AI feedback (reward signals), this stage dramatically enhances Ability Reinforcement, leading to a Final LLM that provides correct, well-reasoned, and aligned responses.

focuses on broad language modeling and understanding, downstream tasks require models to organize their knowledge and linguistic skills to generate well-structured, goal-oriented responses. To address these limitations, post-training methodologies have emerged as a crucial final development stage. As articulated by Zoph & Schulman (2025), post-training aims to “make the model behave like an assistant and follow the right format".

Post-training encompasses two primary directions of advancement. The first focuses on alignment with human values and preferences through Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Ziegler et al., 2019). This approach enables models to learn directly from human feedback, creating an iterative refinement process that better aligns responses with human expectations (Stiennon et al., 2020; Ouyang et al., 2022). The community now widely recognizes RLHF as a critical bridge between foundational pre-training capabilities and the practical requirements of reliable, helpful AI assistants.

The second direction enhances reasoning capabilities through techniques such as Chain-of-Thought reasoning (Wei et al., 2022b) and innovations in test-time scaling (Brown et al., 2024; Snell et al., 2024; Muennighoff et al., 2025), which leverage additional computational resources during inference to improve performance without further training. These advances have yielded remarkable results, exemplified by OpenAI’s o-series model (OpenAI, 2024), which achieves 71.7% accuracy on the software engineering benchmark SWE-bench (Jimenez et al., 2023), 87.7% on graduate-level GPQA tasks (Rein et al., 2023), and 96.7% on competition-level mathematics reasoning (Hendrycks et al., 2021)—often surpassing human expert performance.

Crucially, RL techniques serve as the unifying mechanism across both directions, providing principled methods for optimizing model behavior and enhancing capabilities. The following section provides essential background on these RL principles and their applications to language model development.

2.2 Background on RL

In this section, we present the mathematical foundations of RL (Sutton & Barto, 2018) and subsequently demonstrate how this framework elegantly maps to the LLM optimization problem.

Markov Decision Process. RL is fundamentally grounded in the Markov Decision Process (MDP) formalism (Puterman, 2014), providing a principled approach to sequential decision-making under uncertainty. An MDP is formally represented as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \rho, T)$, where:

- $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively;
- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ defines the transition dynamics, with $s' \sim P(\cdot|s, a)$ representing the probability of transitioning to state s' given current state s and action a;
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ assigns reward values to state-action pairs;
- $\rho : \mathcal{S} \to [0, 1]$ represents the initial state distribution;
- T denotes the finite horizon or episode length.

The central objective in the MDP framework is to identify a policy π_θ that maximizes the expected cumulative reward over the specified horizon T:

$$\max_\theta \mathbb{E}_{s_1 \sim \rho} \mathbb{E}_{a_{1:T} \sim \pi_\theta} \left[\sum_{t=1}^{T} r(s_t, a_t) \mid s_{t+1} \sim P(\cdot|s_t, a_t) \right]. \tag{1}$$

Specification in LLMs. When applying this framework to LLMs, we establish a direct correspondence between MDP components and LLM generation processes. Following (Li et al., 2024c; Rafailov et al., 2024b), a state s_t encompasses the sequence of previously generated tokens, represented as $s_t = (x, a_1, \ldots, a_{t-1})$, where x denotes the initial prompt. An action a_t corresponds to selecting a token from the vocabulary set $\mathcal{V}$. The initial state s_1 is simply the prompt x, with ρ reflecting the prompt distribution. We note that in this formulation, the action space is huge (e.g., $128K$ for Llama-3) and there are emerging works exploring more compact action spaces (Jia et al., 2025; Kim et al., 2025b).

A crucial distinction between LLM-based MDPs and traditional RL problems lies in the nature of state transitions. Unlike conventional RL settings where transitions are typically stochastic due to environmental uncertainty, the MDP formulation for LLMs features **deterministic transitions**—a property leveraged by Li et al. (2024c) to substantially simplify the RL optimization procedure. Specifically, the deterministic transition function P merely appends the current token to the history:

$$P(s_{t+1}|s_t, a_t) = \begin{cases} 1 & \text{if } s_{t+1} = (s_t, a_t) \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Here, (s_t, a_t) denotes the concatenation of the state s_t with action a_t, effectively creating the next state by extending the token sequence.

This determinism contrasts sharply with traditional RL challenges (Sutton & Barto, 2018), where executing identical actions from the same state can yield different outcomes due to environmental stochasticity. Stochastic transition RL problems are inherently difficult to solve (Azar et al., 2017), as this stochasticity lies beyond the learner's control, typically requiring advanced techniques such as value networks for variance reduction (Cai et al., 2020). The deterministic nature of LLM state transitions eliminates this significant source of complexity, enabling more efficient algorithms specifically tailored to language model fine-tuning, as we will review in subsequent sections.

In the context of LLMs, we evaluate complete responses holistically rather than assigning rewards to individual tokens. This approach requires adapting the traditional MDP framework by concentrating the entire reward

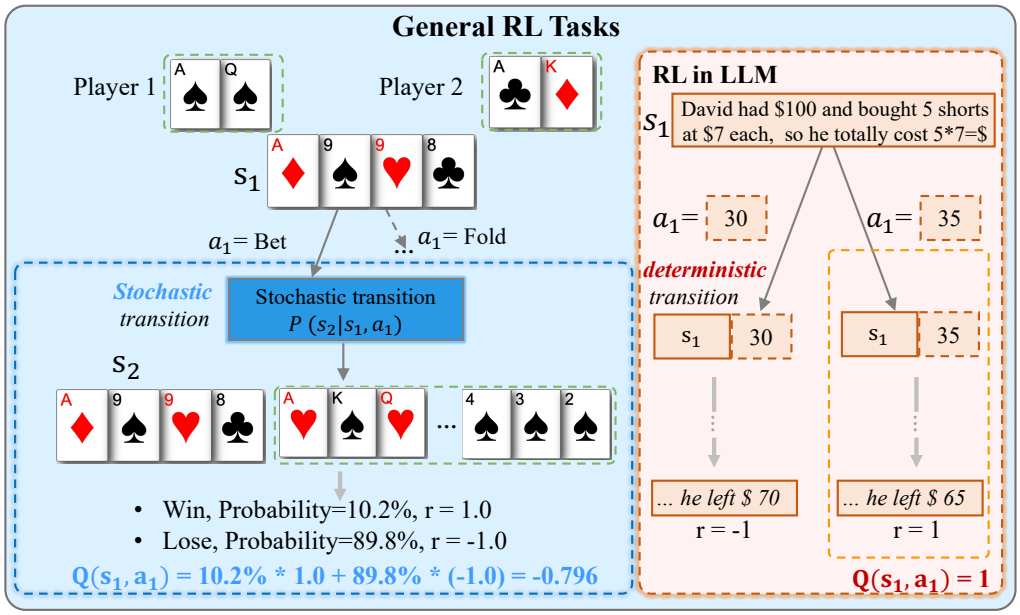


Figure 3: The fundamental distinction between general RL tasks and RL in LLMs. On the left, in a traditional RL poker setting, the state S_1 represents the current game situation, and an action a_1 (e.g., Bet) leads to a stochastic transition to a new state S_2, reflecting environmental uncertainty. The Q-value, $Q(s_1, a_1)$, calculates the expected future return (win rate) considering these probabilistic outcomes. On the right, in the LLM setting, the initial state S_1 is the prompt (x), and an action a_1 corresponds to selecting a token from the vocabulary. A crucial characteristic here is the deterministic transition: appending a chosen token to the current state always results in a unique, predictable next state. Additionally, LLMs typically employ a sparse reward function, where rewards (e.g., $r = 1$ or $r = -1$) are assigned only at the completion of a full response sequence, with intermediate token generation steps receiving zero reward.

signal at sequence completion while assigning zero rewards to all intermediate generation steps:

$$r(s_t, a_t) = \begin{cases} 0 & \text{if } t \neq T \\ r(x, a_{1:T}) & \text{otherwise.} \end{cases} \tag{3}$$

For instance, this sparse reward function might evaluate the helpfulness and harmlessness of an entire response in alignment tasks, or assess final answer accuracy in mathematical reasoning problems.

Under these specifications, the core optimization objective for LLMs becomes a straightforward reward maximization problem:

$$\max_{\theta} \mathbb{E}_{x\sim\rho}\mathbb{E}_{a_{1:T}\sim\pi_\theta}[r(x, a_{1:T})], \tag{4}$$

That is, we seek parameters θ that maximize the expected reward across the distribution of prompts and model-generated responses. When the context is clear, we simply exchange the notation of $a_{1:T}$ and the response $y_{1:T}$ for conciseness.

Note that our current formulation, particularly the definition of a single reward at the end of a fixed-length sequence (T), implicitly focuses on single-turn response generation. This is a common and foundational setting for many tasks, where the model receives an initial prompt (x) and generates one complete, self-contained response ($a_{1:T}$). We notice that the real-world deployment of LLMs frequently involves more complex, sequential interactions. In multi-turn conversational settings (see e.g., (Shani et al., 2024; Xiong et al., 2024)) or sophisticated agentic tasks (see e.g., (Singh et al., 2025; Dong et al., 2025; Xue et al., 2025)), the MDP framework requires extensions. Specifically, the state space ($\mathcal{S}$) would expand significantly. Instead of just the current prompt and partial response, s_t would need to encompass the entire conversational history up to that point. This introduces challenges related to context window management and long-term memory

(Yu et al., 2025a). Furthermore, the concept of an action (a_t) becomes multi-layered. While selecting an individual token from $\mathcal{V}$ remains the fundamental micro-action, the generation of an entire response for a turn could be considered a "macro action" within a higher-level decision process. The reward function (r) may typically shift from a single, end-of-sequence signal to potentially being received after each turn, or even more sparsely after multiple turns, depending on the dialogue's objective. This introduces the significant challenge of credit assignment over extended interaction sequences (Zeng et al., 2025a). Nevertheless, these extended scenarios often build upon the fundamental techniques established in single-turn settings, which remains the primary focus of this survey.

2.3 Formulations

In this section, we introduce three key formulations and tasks in post-training with SFT and RL, and connect them with the terminology defined before.

SFT. As its name suggests, SFT often involves fine-tuning a model with supervised data using the following likelihood-maximization objective:

$$\max_{\theta} \sum_{i=1}^{n} \log \pi_\theta(y^i_{1:T}|x^i) \tag{5}$$

where x^i represents the prompt and instruction, $y^i_{1:T}$ denotes the corresponding completion to be learned, and n is the number of training samples. Note that while x^i is also a sequence of tokens, we omit the timestep subscript for notational clarity.

What is the functional role of SFT and practical insights of the above likelihood-maximization? Answering this question requires examining why pre-trained LLMs struggle with downstream applications. Pre-trained models often exhibit problematic behaviors, including hallucinating facts, generating biased or toxic content, and failing to follow user instructions (Bender et al., 2021; Weidinger et al., 2021). This misalignment stems from a fundamental objective mismatch: pre-training optimizes next-token prediction on internet text, which differs substantially from the desired goal of "following user instructions helpfully and safely" (Ouyang et al., 2022). SFT addresses this misalignment by maintaining the same technical objective (next-token prediction) while fundamentally shifting the training distribution from general text corpora to curated instruction-response pairs and task-specific datasets (Raffel et al., 2020; Wei et al., 2021). This data shift represents a crucial step toward alignment, transforming the model's learned distribution to better match human expectations.

From an RL perspective, SFT serves a critical bridging function: it transforms model outputs into a form that human evaluators or reward models can effectively understand and assess, thereby enabling meaningful reward signals during subsequent RL training (Li et al., 2025g; Zeng et al., 2025b). While direct RL on pre-trained models without SFT is technically possible, as demonstrated by Guo et al. (2025), such approaches often result in models that develop problem-solving capabilities while producing confusing or incoherent outputs that are difficult for humans to interpret.

Why does SFT deserve recognition as a distinct approach? We note that SFT differs from classical supervised learning fundamentally. There could two key aspects from a machine learning perspective. First, in the optimization side, SFT leverages the extensive knowledge already acquired during pre-training, enabling effective adaptation with relatively few examples. This efficiency is further enhanced by the discovery that alignment primarily occurs in the first few tokens of responses (Qi et al., 2024b)—once aligned in these initial tokens, the model tends to maintain alignment throughout the remainder of the response. This characteristic positions SFT within established transfer learning and continual learning paradigms. Second, from a learning theory perspective, the limited size and coverage of SFT datasets position the learning process in an over-parameterized regime, necessitating careful attention to potential overfitting concerns (Xiao, 2024). For example, Li et al. (2025g) point out that the loss function in Equation (5) (actually the cross-entropy loss) can lead to response diversity reduction issues and propose implicit entropy regularization schemes. Another notable example is the alignment tax observed in (Ouyang et al., 2022), where models forget some knowledge and capabilities acquired during pre-training.

RLHF. The framework of RLHF in LLMs is proposed in (Stiennon et al., 2020; Ouyang et al., 2022). The target of RLHF is to further align the model with human preference based on a preference dataset $\mathcal{D}^{\text{pref}}$. The preference dataset has the format of $\mathcal{D}^{\text{pref}} = \{(x, y^w_{1:T}, y^l_{1:T})\}$, where the response $y^w_{1:T}$ is preferred over $y^l_{1:T}$. The RLHF framework typically involves two steps: reward learning and policy learning. To infer the reward, most existing works (Bai et al., 2022a; Ouyang et al., 2022; Rafailov et al., 2023) leverage the following Bradley-Terry (BT) model assumption (Bradley & Terry, 1952).

Assumption 1 (Bradley-Terry (BT) Model)**.** *The Bradley-Terry model defines the preference distribution as*

$$\mathbb{P}(y_{1:T} \succ y'_{1:T}|x) = \sigma(r^\star(x, y_{1:T}) - r^\star(x, y'_{1:T})),$$

where $\sigma(\cdot)$ denotes the sigmoid function and $r^\star$ represents the unknown true reward.

Under the BT model assumption, one can explicitly (Bai et al., 2022a; Ouyang et al., 2022) or implicitly (Rafailov et al., 2023) learn the reward by applying maximum likelihood estimation on the preference dataset.

$$\widehat{r} = \operatorname*{argmax}_r \mathbb{E}_{(x, y^w_{1:T}, y^l_{1:T}) \sim \mathcal{D}^{\text{pref}}} \left[\log\left(\sigma(r(x, y^w_{1:T}) - r(x, y^l_{1:T}))\right)\right].$$

This equation presents the most basic formulation for reward learning; for the latest advancements, please refer to (Lambert et al., 2024; Wang et al., 2024; Malik et al., 2025). Beyond this discriminative reward learning paradigm, an emerging research direction has focused on generative reward models (Zhang et al., 2024; Mahan et al., 2024; Liu et al., 2025e). These models leverage chain-of-thought (Wei et al., 2022b) for reward judgment, offering an alternative approach to traditional discriminative methods.

With the inferred reward model, RLHF further trains the policy model by applying various RL methods to maximize the (estimated) reward with a KL-regularization term.

$$\max_\theta \mathbb{E}_{x \sim \rho}\left[\mathbb{E}_{a_{1:T} \sim \pi_\theta(\cdot|x)}[\widehat{r}(x, a_{1:T})] - \beta D_{\text{KL}}(\pi_\theta(\cdot|x), \pi_{\text{ref}}(\cdot|x))\right].$$

Here $\beta > 0$ is the regularization coefficient and π_{ref} is a reference policy, typically chosen as the policy model after SFT. This KL-regularization term is included to mitigate the well-known reward hacking or reward overoptimization issue (Gao et al., 2022; Rafailov et al., 2024a). In particular, due to the limited size and coverage of the preference data, the learned reward model may deviate from the true reward (Zhu et al., 2023), especially in regions outside the preference data distribution. The subsequent policy learning step may overexploit the imperfect reward model, resulting in suboptimal true performance. This KL-regularization term can mitigate this issue by encouraging the learned policy to stay close to the reference policy in the training process.

Finally, we note that current RLHF methods can be categorized into two types: reward-model-based methods (Bai et al., 2022a; Ouyang et al., 2022) and reward-model-free methods (Rafailov et al., 2023; Azar et al., 2024), depending on whether they explicitly learn a reward model. Unlike the reward-model-free methods, reward-model-based methods can use the learned reward model to compute rewards on data (i.e., prompts and responses) beyond the preference dataset, which can be utilized for policy learning together with preference data (Li et al., 2024b).

Reinforcement Learning with Verifiable Reward. Recently, substantial advancements have been achieved in boosting the reasoning ability of LLMs especially for complex mathematical and coding tasks, including OpenAI-o1 (OpenAI, 2024), DeepSeek-R1 (Guo et al., 2025) and Kimi-1.5 (Kimi et al., 2025). A core method contributing to these advancements is Reinforcement Learning with Verifiable Reward (RLVR). In particular, RLVR trains the LLMs by applying RL to maximize a *rule-based outcome reward*.

$$\max_\theta \mathbb{E}_{x \sim \rho}\left[\mathbb{E}_{a_{1:T} \sim \pi_\theta(\cdot|x)}[r_{\text{rule}}(x, a_{1:T})]\right] - \beta D_{\text{KL}}(\pi_\theta(\cdot|x), \pi_{\text{ref}}(\cdot|x)).$$

This rule-based reward r_{rule} evaluates the correctness of the final answer based on certain golden rules. For mathematical tasks, one can compare the final answer in the response with the true answer. For coding tasks, one can leverage a compiler to evaluate whether the generated code can pass test cases. Notice that such a rule-based reward is an outcome reward by nature, which only cares about the outcome and does not evaluate the correctness of the intermediate reasoning steps.

2.4 Challenges of RL in LLMs

In this section, we highlight several unique challenges in applying RL to train LLMs, which calls for the development of more effective and efficient RL algorithms tailored for LLMs. We highlight two notable challenges below.

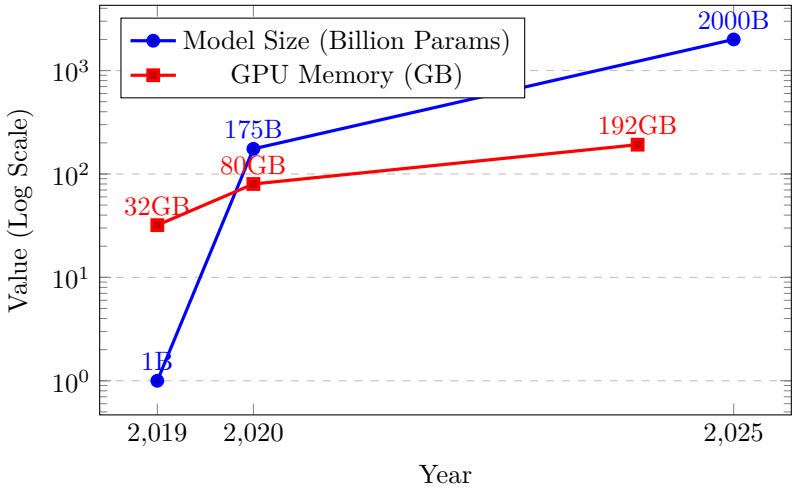


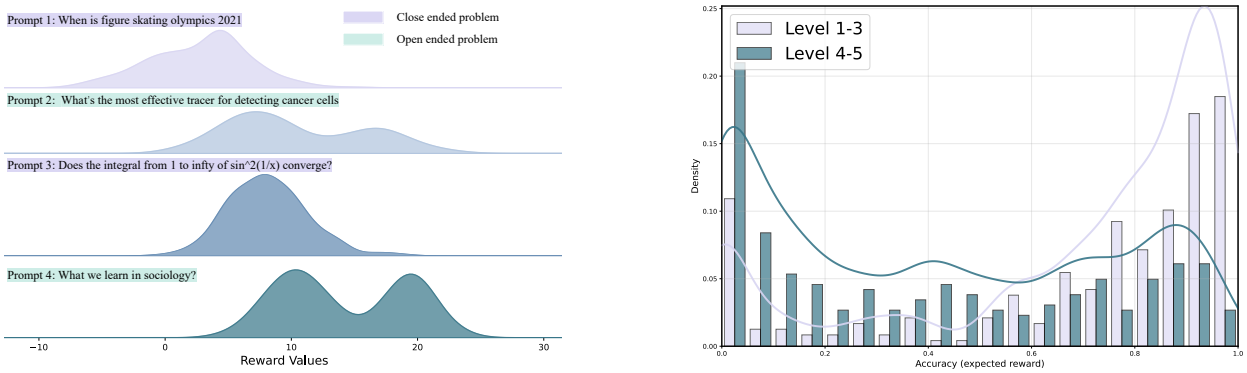
Figure 4: A comparison of LLM model size and single-GPU memory growth, plotted on a logarithmic scale. Model parameters have grown exponentially, from 1.5B (GPT-2, 2019) to 175B (GPT-3, 2020) and a projected 2,000B (Llama-4 Behemoth, 2025). In stark contrast, the memory of flagship GPUs has increased at a much slower, near-linear rate, from 32GB (V100, 2019) to 80GB (A100, 2020) and 384GB (GB200, 2024). This widening gap illustrates a critical hardware bottleneck that complicates the memory-intensive process of RL training for large-scale models.

Challenge: Huge Model Size. The first major challenge stems from the unprecedented scale of modern LLMs, driven by influential scaling laws (Kaplan et al., 2020) that demonstrate consistent performance improvements with increased model parameters across diverse tasks. Following this principle, LLMs have undergone dramatic expansion: GPT-2 contained 1 billion parameters in 2019 (Radford et al., 2019), GPT-3 (Brown et al., 2020) scaled to 175 billion parameters in 2020, and recent models like Llama-4 Behemoth have reached 2 trillion parameters by 2025 (Meta, 2025). This exponential growth in model size creates unique computational demands that distinguish LLM RL training from traditional RL applications.

RL training significantly amplifies these computational challenges compared to standard supervised learning. Unlike supervised training that requires only a single model instance, RL algorithms such as PPO (Schulman et al., 2017) typically maintain multiple model copies simultaneously—including actor networks, critic networks, and reference models for KL regularization—effectively multiplying memory requirements by approximately 2-3 times. For trillion-parameter models, this expanded memory footprint can exceed the capacity of even the most advanced GPU clusters. Moreover, RL algorithms require multiple forward and backward passes per training step, with PPO performing 4-8 gradient steps per batch of collected samples, substantially increasing computational overhead compared to single-pass supervised training.

This challenge is exacerbated by a growing divergence between model scaling and hardware capabilities. As illustrated in Figure 4, model parameter counts have grown exponentially while the memory of individual GPUs has scaled at a much slower, near-linear pace. For instance, flagship NVIDIA GPU memory increased 6-fold from 32GB (V100) to 192GB (B200) between 2019 and 2024. Over a comparable timeframe, state-of-the-art model sizes exploded by more than 1,000-fold (from 1.5B to a projected 2,000B parameters). This widening gap underscores the urgent need for memory-efficient RL algorithms (Rafailov et al., 2023; Li et al., 2024c) and novel distributed training infrastructures to make large-scale RL optimization feasible.

Challenge: Data Heterogeneity. Beyond computational constraints, the heterogeneous nature of training data presents significant challenges. LLMs are typically trained on diverse datasets encompassing multiple



(a) Alignment Task

(b) Math Reasoning Task

Figure 5: Heterogeneity in training data distribution across task types. (a) Alignment tasks show four distinct reward distributions corresponding to different prompt categories (Open ended problem vs. close ended problem), with reward values ranging from approximately -10 to 30. (b) Mathematical reasoning tasks exhibit high variability in expected rewards (accuracy) across different prompts, with accuracy ranging from 0.0 to 1.0. The bimodal distribution in panel (b) suggests two distinct performance regimes, with Level 1-3 tasks (light bars) generally achieving higher accuracy than Level 4-5 tasks (dark bars). The overlaid density curves illustrate the substantial heterogeneity in performance expectations across the prompt distribution for both task categories.

domains and task types, each exhibiting substantially different reward characteristics (Ouyang et al., 2022). This represents a one-model-for-many-tasks paradigm, contrasting sharply with the one-model-for-one-task approach common in classical RL applications (Mnih et al., 2015; Lillicrap et al., 2015).

This heterogeneity manifests across multiple dimensions of task complexity and domain specificity. In preference alignment tasks, reward distributions vary dramatically between prompt categories: open-ended creative prompts typically yield moderate, multi-modal distributions, while factual, closed-ended prompts such as "What is the capital of New Zealand?" often produce more extreme reward values due to their binary correct/incorrect nature (Song et al., 2023). Similarly, in RLVR for reasoning tasks, different mathematical problem difficulty levels exhibit vastly different success rates and corresponding reward distributions. The empirical evidence of this heterogeneity is illustrated in Figure 5, which shows distinct reward distribution patterns across alignment tasks (panel a) and mathematical reasoning tasks of varying difficulty levels (panel b). In alignment tasks, we observe four distinct reward distributions corresponding to different prompt categories, with reward values ranging from approximately -10 to 30. For mathematical reasoning tasks, the bimodal distribution reveals two distinct performance regimes: Level 1-3 tasks (light bars) generally achieve higher accuracy than Level 4-5 tasks (dark bars), with accuracy ranging from 0.0 to 1.0. These substantial distributional differences across task types create fundamental challenges for unified RL optimization.

This reward distribution heterogeneity poses critical challenges for RL training dynamics and model convergence (Li et al., 2024c). The varying scales and distributions can lead to gradient imbalance effects, where tasks with higher reward variance dominate gradient updates, potentially causing catastrophic forgetting in other domains or suboptimal allocation of learning capacity. Tasks with extreme rewards may receive disproportionate attention during optimization, while those with moderate but important improvements may be neglected. Furthermore, conflicting gradient signals from different task types can cause oscillatory training behavior or prevent proper convergence, necessitating specialized RL algorithms designed to handle multi-task gradient rebalancing and ensure stable optimization across heterogeneous reward landscapes.

The two challenges mentioned above directly hinder the scaling of RL for large models and datasets. Beyond these primary issues, RL training may also faces several additional complexities: sparse, delayed reward structures that complicate credit assignment (Lightman et al., 2023b; Cui et al., 2025a); difficult exploration and exploitation in extremely high-dimensional discrete action spaces (vocabularies of 50,000+ tokens) (Ramamurthy et al., 2022; Jia et al., 2025); reward hacking issues with imperfect reward evaluation (Lin et al.,

2023b; Skalse et al., 2022); and difficulties in accurate value function approximation over high-dimensional discrete text sequence spaces (Yuan et al., 2025b). While we cannot provide comprehensive discussion of each individual challenge, we note that some are inherited from classical RL and established wisdom may inspire viable solutions.

3 RL Algorithms

In this section, we review leading RL algorithms (Schulman et al., 2017; Li et al., 2024c; Ahmadian et al., 2024; Chen et al., 2024) for LLM training. We categorize these algorithms into two classes based on their interaction with the environment: *online* RL algorithms that iteratively refine through interaction with the environment, and *offline* RL algorithms that learn from a fixed dataset. Our review primarily focuses on response-level and outcome-based reward formulations. We intentionally limit our discussion of process reward methods (Lightman et al., 2023a), despite their impressive performance on certain tasks (Wang et al., 2023; Yuan et al., 2024; Cui et al., 2025a), as recent research suggests they face significant scalability challenges at the current stage of development (Guo et al., 2025).

3.1 Online Algorithms

At a high level, online RL algorithms for LLMs involve two fundamental conceptual processes: exploration and exploitation (Sutton & Barto, 2018; Agarwal et al., 2019). In the context of LLMs, exploration is the process of sampling diverse responses from the model to discover high-value outcomes and gather informative training data. Exploitation, in turn, involves learning from the collected data to improve the model's policy. This iterative process is illustrated in Algorithm 1.

Algorithm 1 Online RL Framework for LLM Training

Input: Language model π_θ, reward function/model $r : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$
for training iteration $1, 2, \ldots$ **do**
  Exploration: Sample multiple responses for a mini-batch of prompts from π_θ
  Evaluation: Calculate reward values $r(x, y)$ for each prompt-response pair
  Exploitation: Compute gradient estimator using collected data and rewards
  Update: Perform gradient ascent step to update policy parameters θ

The exploration component in LLMs is conceptually straightforward: it typically involves sampling multiple responses for each prompt according to the current language model distribution π_θ. This stochastic sampling naturally provides the diversity needed to explore the response space. However, the exploitation phase—effectively updating the model parameters using the collected data to maximize the performance—presents significant challenges (Ramamurthy et al., 2022; Li et al., 2024c). In this section, we focus primarily on this crucial aspect, discussing key insights for designing gradient estimators that enable stable and effective policy updates. To start with, we first review a classical theory from RL literature.

Theorem 1 (Policy Gradient Theorem Sutton (1988))**.** *The exact gradient of reward maximization in Equation* (4) *can be calculated with the following way:*

$$\nabla_\theta \mathbb{E}_{x\sim\rho}\mathbb{E}_{a_{1:T}\sim\pi_\theta}[r(x, a_{1:T})] = \mathbb{E}_{x\sim\rho}\sum_{a_{1:T}}[r(x, a_{1:T}) \cdot \nabla_\theta \pi_\theta(a_{1:T}|x)] \tag{6}$$

$$= \mathbb{E}_{x\sim\rho}\mathbb{E}_{a_{1:T}\sim\pi_\theta}[r(x, a_{1:T}) \cdot \nabla_\theta \log \pi_\theta(a_{1:T}|x)]. \tag{7}$$

Furthermore, a simple stochastic gradient estimator is

$$\widetilde{g}(\theta) = \sum_{i=1}^{n} r(x^i, a^i_{1:T}) \cdot \nabla_\theta \log \pi_\theta(a^i_{1:T}|x^i) \quad \textit{with } x^i \sim \rho, a^i_{1:T} \sim \pi_\theta(\cdot|x^i). \tag{8}$$

The key insight of the policy gradient theorem lies in the transition from Equation (6) to Equation (7), where the expectation shifts from a uniform distribution to the current model distribution π_θ. This transformation enables the accurate stochastic estimation formulation presented in Equation (8).

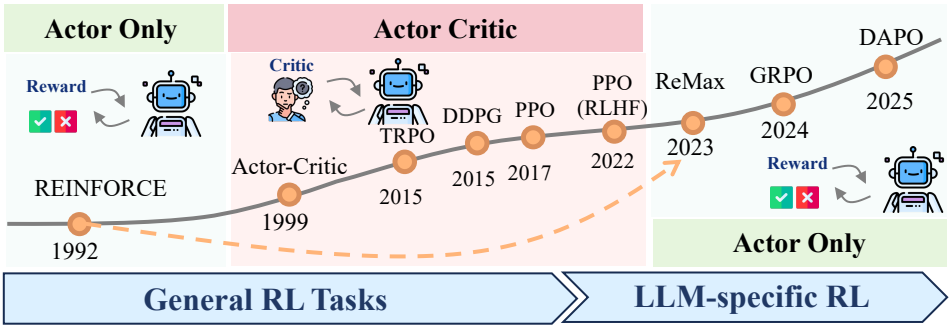


Figure 6: Evolution of RL algorithms. In the early development of RL, algorithms evolved from REINFORCE to actor-critic algorithms, and to PPO. However, with the growing application of RL in LLMs, there has been a shift away from PPO to value-model-free algorithms based on REINFORCE since they are 1) computationally cheap and easy to scale up; 2) does not suffer because no epistemic uncertainty.

At a high level, there are two main considerations in designing policy gradient estimators. One is *stability*, which primarily means seeking an estimator with low variance that enables reliable and stable performance improvements after gradient steps (Kakade, 2001; Schulman et al., 2015). Ensuring such stability is a minimal requirement for successful training, preventing issues like divergence during optimization. Following this foundational aspect, another important consideration is *regularization*, particularly *entropy regularization* (Haarnoja et al., 2018; Cui et al., 2025b). While entropy regularization can introduce a bias away from purely optimal policies, it incentivizes output diversity and enhances exploration. With these considerations in mind, we first review methods focused on improving stability by reducing variance. Subsequently, we discuss emerging studies that additionally incorporate entropy regularization.

History Overview. The first algorithm to use stochastic gradients for policy optimization is REINFORCE (Williams, 1992), introduced in the 1990s and widely referenced in the literature. Specifically, it directly builds on the stochastic gradient estimator in Equation (8). A key feature of REINFORCE is its simplicity of implementation, as it does not require a separate so-called value function. However, it has historically shown poor performance in classical deep RL applications (such as games and robotic control) due to the high variance of its direct stochastic gradient estimates (Sutton & Barto, 2018; Li et al., 2024c). Consequently, from the 2000s to the 2020s, the RL community developed actor-critic algorithms (Konda & Tsitsiklis, 1999; Lillicrap et al., 2015; Haarnoja et al., 2018) that provide lower-variance gradient estimations. The key idea behind these methods is that introducing a value model can accumulate learning from historical data, thereby reducing uncertainty and the variance of stochastic gradients. One such prominent algorithm is Proximal Policy Optimization (PPO) (Schulman et al., 2017). However, Li et al. (2024c) argue that this rationale for variance reduction, specifically concerning external environmental uncertainty, does not fully apply to LLMs due to the absence of epistemic uncertainty in the same way. This insight has spurred the development of more specialized RL algorithms (Shao et al., 2024; Yu et al., 2025b) designed for LLMs. Figure 6 provides an overview of this history. In the following parts, we provide a detailed review of these algorithms, starting with PPO.

PPO. Widely used in deep RL, PPO has proven successful in complex domains ranging from video games to robotics control. We will first review PPO's standard state-action formulation before adapting it to language generation tasks. Building upon natural policy gradient (Kakade, 2001) and trust-region-based policy optimization (Schulman et al., 2015), PPO optimizes a surrogate function:

$$\mathcal{L}_{\text{ppo}} = \mathbb{E}_{x\sim\rho}\mathbb{E}_{a_{1:T}\sim\pi_{\theta_{\text{old}}}}\left[\sum_{t=1}^{T}\widetilde{A}(s_t,a_t)\min\left\{\psi(s_t,a_t),\text{clip}\left(\psi(s_t,a_t),1-\delta,1+\delta\right)\right\}\right], \tag{9}$$

$$\psi(s_t,a_t) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, \tag{10}$$

$$\widetilde{A}(s_t, a_t) = \sum_{j=0}^{T-t} \lambda^j \texttt{advantage}_{t+j} = \sum_{j=0}^{T-t} \lambda^j [r(s_{t+j}, a_{t+j}) + \gamma V(s_{t+1+j}) - V(s_{t+j})]. \tag{11}$$

While this formulation appears complex, it comprises three key design elements:

- **Off-policy learning.** PPO optimizes the policy using data collected from an "old" policy $\pi_{\theta_{\text{old}}}$ (as seen in the expectation over $\pi_{\theta_{\text{old}}}$). However, the policy gradient theorem says that data should follow the distribution π_θ whenever we want to update π_θ. Therefore, directly updating using data from $\pi_{\theta_{\text{old}}}$ would introduce bias. To eliminate this bias, PPO employs importance sampling (Kroese et al., 2013), represented by the ratio $\psi(s_t, a_t)$, which adjusts the data ratio to ensure it is unbiased in expectation. This has a crucial practical implication: it enables *data reuse*. Even after updating policy to $\pi_\theta \neq \pi_{\theta_{\text{old}}}$, we can continue using data from $\pi_{\theta_{\text{old}}}$ rather than immediately collecting new data from π_θ. This approach significantly reduces data collection requirements (Liang et al., 2025).

  In practice, implementing data reuse and importance sampling involves two key details: splitting the data into multiple mini-batches and training them over multiple epochs.[1] Traditional deep RL methods advocate multi-epoch training to maximize data efficiency (Schulman et al., 2017; Huang et al., 2022). However, in language model fine-tuning, single-epoch training is often preferred (Li et al., 2024c; Shao et al., 2024) due to the instability risks posed by repeated updates (Yao et al., 2023b). Even within single-epoch training, the utility of splitting data into mini-batches remains debated. An alternative approach is full-batch gradient descent for on-policy updates, eliminating importance sampling altogether. This was first demonstrated by (Li et al., 2024c) in the context of LLM post-training, with subsequent studies (He et al., 2025; Chen et al., 2025d) suggesting that on-policy updates enhance stability and mitigate entropy collapse.

  We posit that these considerations fundamentally relate to the underlying data properties and offer theoretical insights from classical optimization learning theory (Bottou et al., 2018; Sun, 2019). Given a constant step size, splitting the data to mini-batches and perfoming Stochastic Gradient Descent (SGD) can converges faster than full-batch Gradient Descent (GD) when the data is *homogeneous*: in the extreme case where all samples are identical, SGD's gradient matches that of GD, but SGD processes n times more samples than GD, where n represents the total number of mini-batches. We conjecture that similar principles apply to RL optimization. If the data exhibits homogeneity, off-policy learning could prove more beneficial, provided identical learning rates are maintained. Conversely, with heterogeneous data, the advantages of off-policy learning may diminish substantially.

- **Clipping-based Regularization.** It is important to recognize that importance sampling, while effective, has inherent limitations: since samples from $\pi_{\theta_{\text{old}}}$ are finite and cannot encompass all possible scenarios, statistical estimation errors still exist even with importance sampling adjustments. To mitigate these errors, PPO implements probability ratios clipping as a mechanism to constrain excessive policy updates. Specifically, in the objective function above:

  - If $\widetilde{A}(s_t, a_t) > 0$ (positive advantage), the policy gradient naturally incentivizes increasing $\pi_\theta(a_t|s_t)$. However, when $\psi(s_t, a_t) > 1 + \delta$ (meaning $\pi_\theta(a_t|s_t) > (1 + \delta)\pi_{\theta_{\text{old}}}(a_t|s_t)$), further updates yield zero gradient.
  - Similarly, if $\widetilde{A}(s_t, a_t) < 0$ (negative advantage), the policy gradient incentivizes decreasing $\pi_\theta(a_t|s_t)$, but the clipping mechanism prevents it from falling below $(1 - \delta)\pi_{\theta_{\text{old}}}(a_t|s_t)$.

  As explained in (Schulman et al., 2017), this functions as a single-sided (applying separately to either positive or negative advantage) and point-wise (affecting individual elements in the distribution) regularization mechanism. This approach contrasts with the holistic and distributional KL regularization employed in TRPO (Schulman et al., 2015). Furthermore, PPO's regularization is post-hoc (applied after the policy probability ratio exceeds the threshold), while TRPO's KL-regularization is preventative (ensuring the updated policy never exceeds the boundary).

  Recent advances in the literature suggest that relative clipping may not be the optimal formulation. Two specific scenarios illustrate this limitation. First, in exploration scenarios involving potentially

[1] For instance, the former corresponds to the `ppo_mini_batches` hyperparameter, while the latter aligns with `ppo_epochs` in the training framework `verl` (Sheng et al., 2024).

beneficial actions with initially low probabilities, it may be advantageous to substantially increase these probabilities to exploit discovered rewards. Addressing this, Yu et al. (2025b) introduced asymmetric regularization coefficients with a larger δ value for positive advantage cases, allowing for more aggressive probability increases when warranted. Second, excessively increasing the probability of already high-likelihood actions with positive advantages (i.e., too much exploitation) can undermine exploration of other potentially beneficial actions. To address this concern, Li et al. (2025g) developed ranking-based probability flow regularization, operating on the insight that maintaining relative ranking within the distribution, rather than driving probabilities to extreme values, is sufficient to preserve action diversity while still enabling effective learning.

- **Generalized Advantage Estimation.** PPO employs an advanced method, Generalized Advantage Estimation (GAE) (Schulman et al., 2016), for estimating advantage, which represents a "centralized" version of reward; see Equation (11). We first clarify that this centralization preserves the optimal action ordering, thus maintaining unbiasedness. Furthermore, while centralization alone does not alter variance, when combined with stochastic sampling, it effectively reduces variance, as analyzed in (Dayan, 1991; Li et al., 2024c).

  The advantage calculation in PPO implements an Exponential Moving Average (EMA) of one-step advantage estimations. Specifically, the one-step advantage is computed as the difference between the immediate reward plus the discounted value at the subsequent state $V(s_{t+1})$ and the estimated value at the current state $V(s_t)$. The value function, defined as the expected long-term cumulative future rewards, must be *separately trained* using Temporal Difference (TD) learning methods (Sutton, 1988). As Li et al. (2024c) point out, this separate training effectively doubles the required computational resources and significantly extends the overall training time. The intricacies of value network learning extend beyond the scope of this survey; interested readers are directed to (Sutton & Barto, 2018) for a comprehensive treatment of this subject. Finally, we note two critical hyperparameters govern the calculation of GAE: $\gamma \in [0, 1]$ controls the discount rate for future rewards, while $\lambda \in [0, 1]$ regulates the discount rate for future advantages. We will exaime these hyper-parameters later.

To summarize, PPO operates iteratively through the following process: 1) collecting trajectories using the current policy; 2) estimating advantages using the value function; 3) updating the policy using the clipped surrogate objective with SGD-style algorithms (usually Adam (Kingma, 2014)); 4) simultaneously training the value function to better estimate returns same as the policy model does. This process continues until convergence, demonstrating PPO's online nature.

We clarify that algorithms that we will introduce later mainly differ PPO with the advantage estimation, with the first two ideas directly applicable to these algorithms. We further exclaim the third point below. We observe that the best practices for applying PPO to LLMs typically involve setting $\gamma, \lambda = 1$ (Yao et al., 2023b; Ahmadian et al., 2024). Under the further assumption of on-policy update, PPO effectively reduces to the REINFORCE estimator (Williams, 1992) with a baseline value (a point we will formally elaborate on later) with $V(s_t)$ being the baseline value (Li et al., 2024c). To put it in the context of LLM with outcome reward, PPO essentially reduces to

$$\mathcal{L}_{\text{ppo}}(\theta) = \mathbb{E}_{x\sim\rho}\mathbb{E}_{y_{1:T}\sim\pi_\theta}\left[\sum_{t=1}^{T} r(x, y_{1:T}) - V(x, y_{1:t})\right].$$

This simplification reveals an important insight for LLM applications: the value network in PPO primarily serves to provide a baseline value for gradient estimation, while GAE—despite its importance in traditional RL tasks—plays no significant role in the language model context.

Takeaways: PPO reduces to REINFORCE with Token-wise Baseline

- PPO requires training two separate networks: the language model itself and a value network, which effectively doubles both the computational resources and training time.
- When training LLMs with trajectory-level or outcome-based rewards, the value network primarily serves to provide a baseline for advantage estimation, while GAE and TD learning techniques—central to traditional RL—become unnecessary.

From PPO to REINFORCE. Recent research has focused on developing more efficient algorithms for LLM training. Li et al. (2024c) were the first to argue that PPO is overshot for LLMs and developed methods that leverage the special properties of language models. They highlighted that PPO was originally designed for general RL tasks where a value network serves a crucial purpose: in traditional RL environments, stochastic transitions create uncertainty beyond the agent's control, resulting in noisy return estimations. To address this challenge, the RL community developed value networks that could be iteratively trained on historical data to provide more accurate return estimates.

However, Li et al. (2024c) demonstrated that this core rationale for a value model largely dissipates in the context of LLMs. Crucially, token generation in language models is a deterministic process once the prompt and model parameters are fixed (assuming no stochastic sampling during inference). This fundamental difference implies that the "environment" (the LLM itself) does not introduce the same epistemic uncertainty that necessitates a complex value function for variance reduction. Consequently, there is no inherent need for a separate value model in the LLM setting.

From a computational efficiency perspective, eliminating the value model offers significant advantages. Specifically, the GPU memory traditionally allocated to the value network can be re-allocated to accelerate response sampling during the exploration phase, or to accommodate larger models/batches. Furthermore, the substantial training time and computational overhead associated with learning and updating the value network are completely saved, as noted in (Li et al., 2024c). Importantly, their research showed that performance comparable to PPO could be achieved without a value network by building upon the simpler REINFORCE algorithm, a finding that would be highly improbable in classical deep RL applications due to REINFORCE's historically high variance.

While this shift to a simpler framework is promising, the naive REINFORCE method suffers from well-known high variance issues, as documented in classical RL literature (Sutton & Barto, 2018) and reiterated for LLMs by Li et al. (2024c). However, this does not necessitate abandoning REINFORCE entirely. A well-established approach to address the variance problem is to augment it with a baseline, forming the REINFORCE with Baseline framework (Dayan, 1991; Sutton & Barto, 2018):

$$\widetilde{g}_{\text{REINFORCE-with-baseline}}(\theta) = \sum_{i=1}^{N} \left(r(x^i, y^i_{1:T}) - b^i \right) \cdot \nabla_\theta \log \pi_\theta(y^i_{1:T}|x^i) \quad \text{with } x^i \sim \rho, y^i_{1:T} \sim \pi_\theta(\cdot|x^i). \tag{12}$$

Here b^i represents the baseline value to be designed. That is, we introduce a baseline value for reference and the term $r(x^i, y^i_{1:T}) - b^i$ is also often referred to as "advantage". Thus, a reference point is introduced.

A key property is that this estimator remains unbiased, as demonstrated in (Williams, 1992). However, its variance can be substantially reduced (Dayan, 1991; Greensmith et al., 2004; Li et al., 2024c). Furthermore, a critical design choice for the baseline value is making it *prompt-dependent*: rather than using a single baseline value for all prompts, an adaptive baseline value for each prompt is essential for LLMs. This addresses the data heterogeneity challenges mentioned previously: rewards across different prompts behave quite differently. Without prompt-specific baselines to normalize these reward magnitudes, gradient updates become severely imbalanced across different prompt types, leading to training instability and potentially compromising model convergence.

Next, we review recent advances in designing baseline values. Most modern methods rely on direct Monte Carlo estimation, in contrast to the model-based estimation approach used in PPO. For clear comparison and understanding, we present the general form below:[2]

$$\widetilde{g}_{\text{REINFORCE-with-baseline}}(\theta) = \sum_{i=1}^{N} \sum_{j=1}^{K} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(y^{[i,j]}_t | x^{[i]}, y^{[i,j]}_{1:t-1}) \cdot \texttt{advantage}(x^{[i]}, y^{[i,j]}_{1:t-1})$$

Here K corresponds to the number of generated responses per prompt. For PPO, we have:

$$\texttt{advantage}_{\text{PPO}}(x^{[i]}, y^{[i,j]}_{1:t}) = r(x^{[i]}, y^{[i,j]}_{1:T}) - b_{\text{PPO}}(x^{[i]}, y^{[i,j]}_{1:t})$$

[2]We omit techniques such as off-policy learning and clipping regularization for clarity of presentation, though these can be implemented in practice.

$$b_{\text{PPO}}(x^{[i]}, y_{1:t}^{[i,j]}) = V(x^{[i]}, y_{1:t}^{[i,j]})$$

Note that the baseline value is calculated at the token level. This approach can be beneficial when the value estimation is accurate (i.e., when the value network is well-trained (Yuan et al., 2025b)), but may be detrimental when the value network is imperfectly trained.

Variants of REINFORCE. Several key design approaches have emerged:

- ReMax (Li et al., 2024c) selects the baseline value to the reward of greedy decoding response.

$$\texttt{advantage}_{\text{ReMax}}(x^{[i]}, y_{1:t}^{[i,j]}) = r(x^{[i]}, y_{1:T}^{[i,j]}) - b_{\text{ReMax}}(x^{[i]}) \tag{13}$$

$$b_{\text{ReMax}}(x^{[i]}) = r(x^{[i]}, \bar{y}_{1:T}), \text{ with } \bar{y}_t^{[i]} \in \operatorname{argmax} \pi_\theta(\cdot|x^{[i]}, \bar{y}_{1:t}^{[i]}) \tag{14}$$

  The rationale is that greedy decoding represents the mode (i.e., the highest probability) of the policy distribution, which provides compact information and is computationally efficient.

- GRPO (Shao et al., 2024) replaces the value network estimation with the direct reward value.

$$\texttt{advantage}_{\text{GRPO}}(x^{[i]}, y_{1:t}^{[i,j]}) = \frac{r(x^{[i]}, y_{1:T}^{[i,j]}) - b_{\text{GRPO}}(x^{[i]})}{\text{std}\{r(x^{[i]}, y_{1:T}^{[i,1]}), \ldots, r(x^{[i]}, y_{1:T}^{[i,K]})\} + \epsilon}$$

$$b_{\text{GRPO}}(x^{[i]}) = \text{mean}\{r(x^{[i,1]}, y_{1:T}^{[i,1]}), \ldots, r(x^{[i,K]}, y_{1:T}^{[i,K]})\}$$

  where ϵ is a small positive number (e.g., 10^{-6}) to avoid numerical division error. We note that this baseline value is actually depeds on both $x^{[i]}$ and $\{r(x^{(i,1)}, y_{1:T}^{(i,1)}), \ldots, r(x^{(i,K)}, y_{1:T}^{(i,K)})\}$, but we omit the later in the notation to emphasize that the baseline value is at the response-level and to avoid the notation cluster. This normalization is in part follows the normalization of advantage used in OpenAI's PPO implemention[3], with the exception that the normalization is done per prompt rather than across whole dataset.

- RLOO: Similar to GRPO, Ahmadian et al. (2024) leveraged the leave-one-out estimator in (Kool et al., 2019):

$$\texttt{advantage}_{\text{RLOO}}(x^{[i]}, y_{1:t}^{[i,j]}) = r(x^{[i]}, y_{1:T}^{[i,j]}) - b_{\text{RLOO}}(x^{[i]})$$

$$b_{\text{RLOO}}(x^{[i]}) = \frac{1}{K-1} \sum_{j=1:j\neq i}^{K} r(x^{[i]}, y_{1:T}^{[i,j]})$$

  This estimator leverages the leave-one-out principle, which calculates the baseline for each response by averaging the rewards of all other responses generated for the same prompt.

One notable feature of the above baselines is that ReMax leverages only one response for baseline estimation, while others use multiple responses for baseline estimation. This is largely because when ReMax was proposed, the training infrastructure did not support efficient sampling of multiple responses (actually supporting only 1 or 2 responses at that time), so using the greedy baseline value was the most efficient and effective approach in the context of limited sampling.

We note that the primary benefit of introducing a baseline lies in its variance reduction properties for stochastic gradients. Rigorous theoretical work by Dayan (1991) and Li et al. (2024c) demonstrates that appropriate baselines reduce gradient estimation variance. The underlying mechanism involves introducing a random variable $\nabla_\theta \log \pi_\theta(a_{1:T}|x) \cdot b(x)$ that positively correlates with $\nabla_\theta \log \pi_\theta(a_{1:T}|x) \cdot r(x, a_{1:T})$. This correlation reduces the second moment and consequently the variance—a technique known in statistical theory as a control variate (Kroese et al., 2013).

While alternative variance-reduction designs exist beyond REINFORCE with baseline, such as stochastic variance reduced gradient (Johnson & Zhang, 2013) approaches (demonstrated in (Papini et al., 2018) and (Xu et al., 2019)), these methods typically introduce computational overhead that exceeds the REINFORCE with baseline framework. Consequently, despite their theoretical merits, these alternative techniques have not seen widespread practical adoption.

[3] https://github.com/openai/baselines/blob/master/baselines/ppo2/model.py#L139

We also note that unlike ReMax and RLOO that can be proved to be unbiased, GRPO's gradient estimator has inherent biases. There are two distinct error sources. First, the standard deviation calculation introduces statistical correlation because it uses the same random samples, resulting in bias. Recent work by Liu et al. (2025c) has proposed eliminating the standard deviation component altogether to address these potential bias issues in GRPO. Second, the mean estimation introduces statistical bias, because it depends on random samples $(x^{[i]}, y^{[i,j]})$. However, Kool et al. (2019) proved that this bias is largely eliminated (up to a scaling constant) when applying double summation over j. We first clarify that the bias does not mean that this algorithm cannot converge. From the perspective of optimization theory, as long as the angle between update direction and the gradient direction is smaller than 90 degrees, the algorithm can be expected to converge asymptotically. We also note that from the perspective of optimization algorithms, the standard deviation correction may be viewed as "adaptive learning rates" with the insight that a smaller learning rate should be employed for high-variance stochastic gradients (see, e.g., the formal theoretical arguments in (Bottou et al., 2018)). In the context of GRPO, this means that GRPO uses the reward variance (a simple statistic as a proxy for reward noise) and adaptively adjusts the learning rate for each prompt. Thus, the bias introduced here could alleviate the variance issue and in some cases could be beneficial (Liu et al., 2025d).

The Optimal Baseline Value. Since we aim to minimize the variance of stochastic gradients, a natural question arises: what is the best design of baseline value in terms of variance reduction?

Proposition 1 ((Greensmith et al., 2004; Li et al., 2024c))**.** *The minimal-variance baseline value for Equation* (12) *is*

$$b^\star(x) = \frac{\mathbb{E}_{y_{1:T}\sim\pi_\theta}[r(x, y_{1:T}) \cdot \|\nabla_\theta \log \pi_\theta(y_{1:T}|x)\|_2^2]}{\mathbb{E}_{a_{1:T}\sim\pi_\theta}[r(x, y_{1:T})]}$$

This optimal baseline value incorporates the score function $\nabla_\theta \log \pi_\theta(y_{1:T}|x)$ into its calculation, as this component—in conjunction with the advantage function—directly influences gradient magnitude. Mechanistically, it uses the squared norm of the score function to re-weight reward values. However, this optimal baseline is computationally expensive to calculate, requiring additional forward and backward passes through the model. Consequently, there has been limited empirical investigation since it was first proposed. Recent work by Hao et al. (2025) has empirically studied this baseline value using various approximation techniques to make it more tractable in practice.

From a theoretical rigous perspective, it is important to note that only the optimal baseline value achieves global variance reduction (remaining effective across any policy π_θ), while baseline values employed in PPO, ReMax, GRPO, and RLOO achieve only local variance reduction. This fundamental distinction should be credited to the seminal work of Dayan (1991). For comprehensive theoretical developments on variance reduction techniques, readers are directed to both Dayan (1991) and the recent advances in Li et al. (2024c).[4]

Finally, we note that there are variants of policy optimization algorithms (Richemond et al., 2024; Gao et al., 2024; Kimi et al., 2025) that are derived not from the inner product objective of reward maximization as in Equation (4), but from other divergence functions. However, they share essentially the same insights as the above methods, so we do not provide detailed discussion here. Furthermore, there are emerging variants (see e.g., (Yu et al., 2025b; Yuan et al., 2025b; Chen et al., 2025b; Chu et al., 2025; Arnal et al., 2025; Zhang et al., 2025a)) of REINFORCE methods for LLM fine-tuning within specific contexts. To name a few, Hu et al. (2025) proposed global advantage normalization to improve training stability, while Zheng et al. (2025a) proposed group sequence policy optimization, which performs sequence-level importance-sampling-ratio correction to address training instability when training large-scale mixture-of-experts models. Additionally, GMPO (Zhao et al., 2025b) stabilizes training by optimizing the geometric mean of token rewards, and PKPO (Walder & Karkhanis, 2025) optimizes for pass@k performance to enhance sample diversity.

The Dynamics of Policy Optimization. While our previous discussion centered on the global design of policy gradient estimators—using techniques like baselines to reduce variance—we now shift our focus to the

[4]To avoid potential misunderstandings, we emphasize that local variance reduction does not compromise algorithmic convergence. Rather, it indicates that under specific circumstances, the variance of REINFORCE-with-baseline might exceed that of naive REINFORCE. However, in such regimes the absolute variance is typically already sufficiently small, making relative comparisons between methods inconsequential for overall performance.

granular dynamics of the learning process. This section examines the functional role of individual tokens and their interaction with policy entropy. A growing body of research suggests that understanding these token-level dynamics is crucial for effective policy optimization (Li et al., 2025g; Wang et al., 2025c; Zhu et al., 2025; Cui et al., 2025b; Yang et al., 2025d).

We first discuss the influence of high- and low-probability tokens. Recent studies highlight a fundamental tension in policy optimization. On one hand, maintaining high policy entropy is strongly correlated with better performance and exploration. For instance, Cui et al. (2025b) show that an LLM's exploratory capacity, quantifiable by its entropy, is a consistent indicator of success. Similarly, Wang et al. (2025c) reveal that tokens with high entropy are pivotal for guiding the model toward diverse reasoning paths. On the other hand, the optimization process can be disproportionately influenced by specific tokens. Yang et al. (2025d) find that low-probability tokens can dominate training due to their large gradient magnitudes. This creates a challenge: the very tokens that represent exploration can also destabilize learning. These findings collectively imply that training dynamics are critically sensitive to the gradients of certain tokens.

To better understand this phenomenon, we can visualize the interplay between a token's probability and the direction of its gradient update. As illustrated in Figure 7, this dynamic can be mapped onto a four-quadrant grid defined by two axes: *token probability* (low vs. high) and *gradient direction* (positive for reinforcement vs. negative for correction). Each quadrant represents a distinct learning scenario with a unique impact on policy entropy:

- Top-Left (Reinforcing a High-Probability Token): When a confident, correct prediction receives a positive gradient, the model's existing beliefs are reinforced. This sharpens the probability distribution and decreases entropy, potentially leading to overfitting—a scenario where regularization techniques like those in (Li et al., 2025g) become essential.
- Top-Right (Correcting a High-Probability Token): When a confident but incorrect prediction receives a negative gradient, the model is forced to correct a common mistake. This redistributes probability mass away from the incorrect token, naturally increasing entropy and encouraging exploration.
- Bottom-Left (Reinforcing a Low-Probability Token): In this key exploratory scenario, a novel action (a low-probability token) is rewarded with a positive gradient. This validates a new reasoning path and increases entropy by boosting a previously overlooked option. Successfully learning in this regime may require relaxing gradient update restrictions, as proposed by (Yu et al., 2025b).
- Bottom-Right (Correcting a Low-Probability Token): When an exploratory mistake receives a negative gradient, the model learns to prune an unproductive reasoning path. This removes a poor option from consideration and decreases entropy.

Crucially, in both scenarios involving low-probability tokens (the left two quadrants), the gradients have a large magnitude (Li, 2025). This makes their optimization a delicate balancing act, requiring careful management to harness the benefits of exploration without destabilizing the training process.

3.2 Offline Algorithms

There is another class of algorithms called offline algorithms, with the representive work of Direct Preference Optimization (DPO) (Rafailov et al., 2023), which updates the LLM with a fixed dataset. We clarify that offline RL is extensively studied in the classical RL setting, but usually from a randomly initialized network and addressed challenges like distribution shift and over-estimation issues (Fujimoto et al., 2019; Kumar et al., 2020). They have not been successfully in LLM in large scale, therefore we do not review them; please refer to (Levine et al., 2020; Prudencio et al., 2023) for references.

Compared with the general RL framework, DPO solves a specific classes of RL problems: KL-regularized reward-maximization problems:

$$\pi^\star \in \operatorname*{argmax}_{\pi} \mathbb{E}_{y\sim\pi(\cdot|x)}[r(x,y)] - \beta D_{\mathrm{KL}}(\pi, \pi_{\mathrm{ref}}).$$

Here $\beta > 0$ is a hyper-parameter controls the strength of KL regularization, and π_{ref} is a reference model (usually the base model or SFT model) for reference. By introducing the KL regularization, there is a

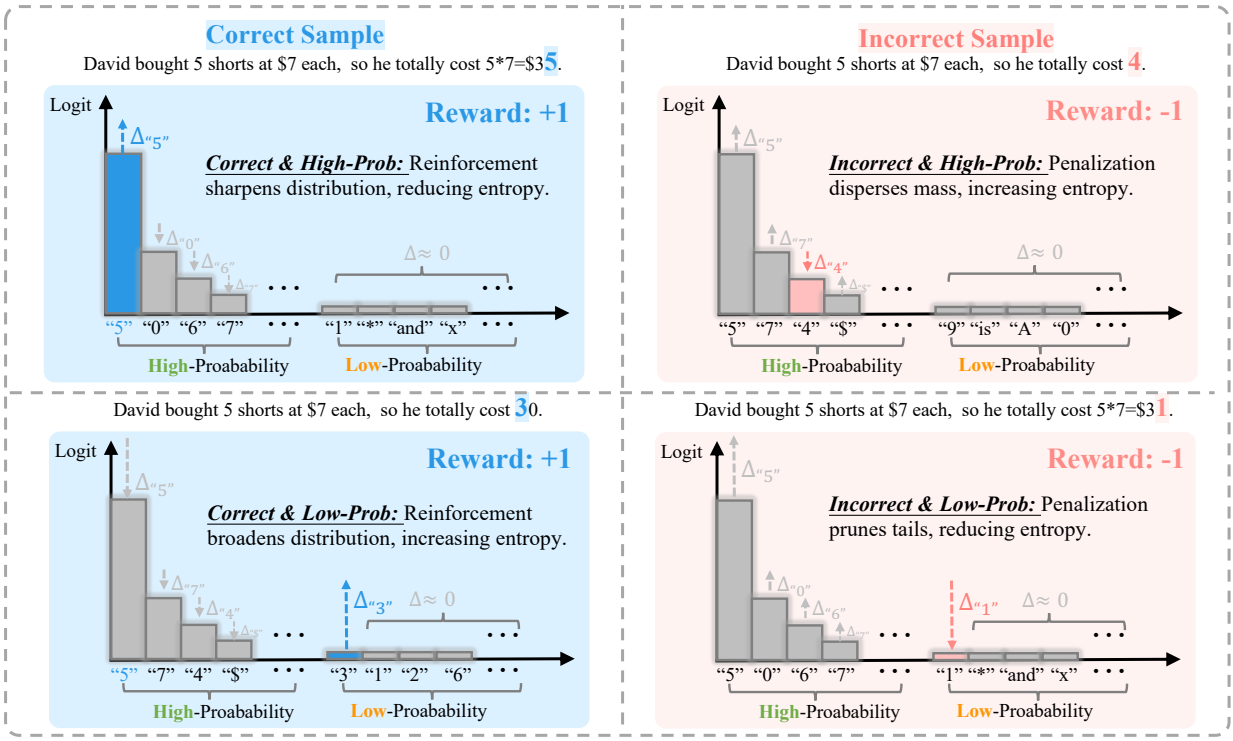


Figure 7: The four quadrants of gradient dynamics in policy optimization. This figure illustrates how token updates affect policy entropy based on the token's initial probability (x-axis) and the direction of the gradient (y-axis). Updates that increase entropy (fostering exploration) occur when correcting a high-probability token or reinforcing a low-probability one. Conversely, updates that decrease entropy (promoting exploitation) occur when reinforcing a high-probability token or correcting a low-probability one.

closed-form solution:

$$\pi^{\star}(y|x) = \frac{1}{Z(x)}\pi_{\text{ref}}(y|x)\exp\left(\frac{r(x,y)}{\beta}\right).$$

where $Z(x) = \sum_y \pi_{\text{ref}}(y|x)\exp(r(x,y)/\beta)$. This allows a one-to-one mapping between the reward function and the optimal policy $\pi^{\star}$. Hence, we can parameter $\pi^{\star}$ by direct parameterization of r. This insight is leveraged in (Rafailov et al., 2023). They studied preference learning from Bradley-Terry model loss, the policy optimization objection can be derived from the close-form solution:

$$\mathcal{L}_{\text{DPO}}\left(\pi_\theta;\pi_{\text{ref}}\right) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta\left(y_w\mid x\right)}{\pi_{\text{ref}}\left(y_w\mid x\right)} - \beta\log\frac{\pi_\theta\left(y_l\mid x\right)}{\pi_{\text{ref}}\left(y_l\mid x\right)}\right)\right] \tag{15}$$

The key insight is that LLMs are viewed as an implicit reward model. For preference data, the model's own log probabilities can be interpreted as indicators of how preferable one response is over another. During policy optimization, DPO directly adjusts the log-likelihoods to increase the probability of preferred responses relative to depreferred ones.

DPO simplifies the training pipeline by eliminating the need for an explicit reward model, making it appealing for practical applications. For example, LLaMA 3 (Grattafiori et al., 2024) replaces traditional RLHF with DPO optimization. In practice, various implementation tricks are often employed to improve stability and performance (Park et al., 2024; Pang et al., 2024; Ethayarajh et al., 2024; Wu et al., 2024a; Meng et al., 2024; Pal et al., 2024; Kim et al., 2025a; Lou et al., 2025). For instance, Park et al. (2024) studied the length bias issue of DPO and proposed a length-normalized variant. Pang et al. (2024) observed that the likelihood of positive responses can decrease and proposed adding a next-token-prediction loss to mitigate this issue. SimPO (Meng et al., 2024) incorporates the average log probability, effectively aligning with model generation while removing the dependency on a reference model, thereby improving computational

and memory efficiency. Xiao et al. (2024) pointed out that RLHF algorithms, including DPO, cannot recover the true preference policy and proposed entropy regularization to address this limitation.

Since the introduction of DPO, there has been increasing interest in extending it beyond the Bradley–Terry framework, as it cannot address the issue of conflicting preferences. Please refer to (Azar et al., 2024; Wu et al., 2024b; Swamy et al., 2024; Liu et al., 2025a) for recent advances and related work.

Connection with Online RL Algorithms. Offline RL algorithms have fundamental gaps with online RL algorithms (Li et al., 2024b; Nika et al., 2024). Xu et al. (2024) have presented comprehensive empirical results to support this claim. The core advantage of online RL algorithms lies in their "online optimization" characteristic, meaning the model can generate new responses and learn from this new data produced by the current policy. This contrasts with DPO's reliance on a human-crafted static dataset. The work by (Li et al., 2024b) emphasizes that the improvement potential of online RL algorithms is best realized when online updates are sufficient, including an adequate supply of model-generated prompts and responses. This implies that offline methods could be enhanced by introducing some form of "online" data collection and utilization mechanism, even if simulated or model-driven. For instance, introducing iterative DPO (Xiong et al., 2023; Pang et al., 2024), by generating new responses using the current model and then utilizing a learned reward model to label this new data for the next round of DPO training. This process introduces elements analogous to online learning, wherein the model learns from data generated by its own (or an updated) policy, thereby partially bridging purely offline learning with the dynamics of online learning and exhibiting some robustness to OOD issues.

4 Large-scale RL Training Framework

The preceding section focused on the algorithmic foundations of RL, particularly gradient estimation and optimization. These algorithmic advancements, when applied to LLMs, necessitate corresponding innovations in large-scale training infrastructure. We now transition from theory to practice, examining the significant systems engineering challenges required to support these algorithms. A primary challenge in implementing RL for LLMs stems from the need to concurrently support both model training and response sampling on GPUs. Specifically, while LLMs based on Transformers rely on KV-cache for accelerated inference (response sampling), this mechanism, designed for efficient generation, often introduces complexities when directly combined with the demands of model training on the same hardware. This inherent incompatibility in GPU resource management has spurred the development of new training infrastructure since.

This section specifically focuses on practical system designs for the online RL framework presented in Algorithm 1, drawing from recent advancements in the field (Yao et al., 2023b; Xiao et al., 2023; Mei et al., 2024; Hu et al., 2024; Sheng et al., 2024; Shen et al., 2024; Fu et al., 2025; Wu et al., 2025; Wang et al., 2025d). Through an analysis of these techniques, we aim to provide practical guidance for implementing large-scale infrastructure and offer insights into improving the computational efficiency of RL systems, thereby enabling further scaling of RL applications for LLMs.

As mentioned before, RL training is an iterative process alternating between two key phases. The first is exploration, where the model generates diverse responses to find optimal solutions; from a systems perspective, this is the response sampling phase. The second is exploitation, where the model updates its parameters via gradient descent (i.e., backpropagation) to improve itself based on the collected data; this corresponds to the model updating phase. Accordingly, we will separately review response sampling and model updating before discussing their system integration.

4.1 Model Updating

In the algorithmic framework, the model update process is conceptually straightforward: model parameters are updated via gradient descent on a predefined loss function. In practice, however, implementation presents significant challenges. For instance, model sizes often exceed the memory capacity of a single GPU, while large datasets create computational bottlenecks that slow down training iterations.

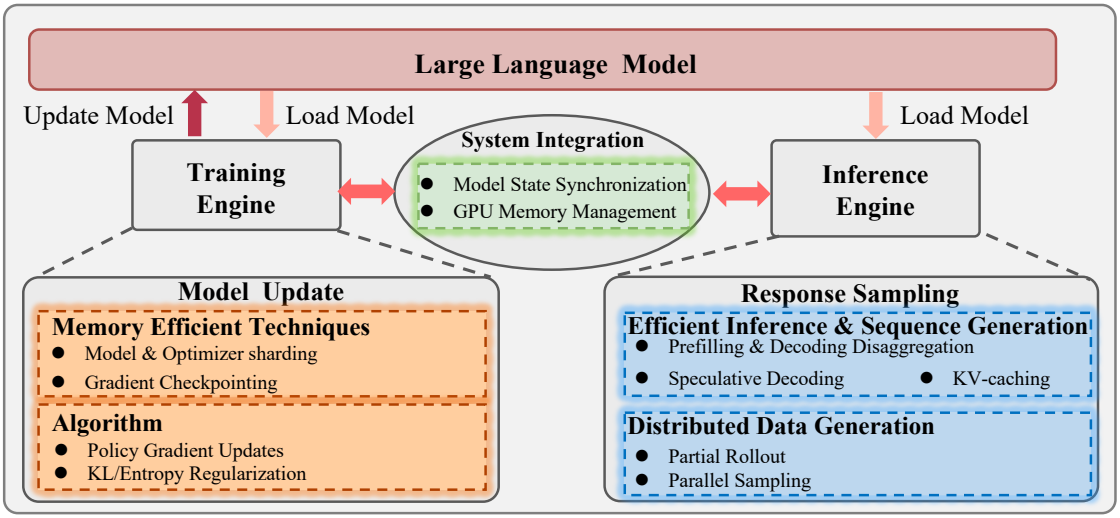


Figure 8: Architecture of a large-scale reinforcement learning (RL) system for large language models. The framework separates the Model Update and Response Sampling pipelines, which are coordinated through a central System Integration module. The Model Update pipeline, executed by the Training Engine, refines model parameters via memory-efficient techniques (e.g., model and optimizer sharding, gradient checkpointing) and RL algorithms (e.g., policy-gradient updates, KL/entropy regularization). In parallel, the Response Sampling pipeline, executed by the Inference Engine, efficiently generates new data using methods such as prefilling and decoding disaggregation, speculative decoding, KV-caching, partial rollout, and parallel sampling. System Integration synchronizes model states and manages GPU memory across the training and inference loops, enabling continuous, low-latency data generation and model optimization.

Distributed training systems are the standard solution. Frameworks like FSDP (Zhao et al., 2023), DeepSpeed (Rajbhandari et al., 2020), and Megatron (Narayanan et al., 2021) overcome these hurdles using a combination of parallelism strategies. These strategies can be broadly categorized:

- Data Parallelism (Li et al., 2020) addresses large datasets by distributing data batches across multiple workers.
- Model Parallelism (e.g., tensor (Shoeybi et al., 2019) and pipeline (Narayanan et al., 2021) parallelism) addresses large models by partitioning the model itself—sharding components like MLPs and attention layers—across several GPUs.

We refer readers to existing surveys for a deeper technical dive (Verbraeken et al., 2020; Duan et al., 2024).

These strategies apply to pre-training, SFT and RL, since all involve gradient-based model updates. However, compared with SFT or pre-training on fixed datasets, RL training presents additional computational challenges. For instance, using mini-batches from a replay buffer creates an off-policy setting that requires careful hyperparameter design, particularly for learning rates (Hilton et al., 2022). Furthermore, properly setting the number of iterations or model updates for regularization schemes (e.g., KL-regularization) is challenging at scale (Gao et al., 2022). While pre-training addresses such issues through techniques like hyperparameter scaling laws (Bjorck et al., 2024; Li et al., 2025c; Kaplan et al., 2020) and μ-Parameter transfer (Yang et al., 2022), efficient parameter setting in large-scale RL remains largely unexplored territory. We believe these research questions are of practical important as we scale model and compute during RL training and believe there are huge research opportunities.

4.2 Response Sampling

Beyond model updates, RL training requires continuous response sampling to explore and collect high-quality data. While conceptually straightforward—involving standard auto-regressive sampling—the practical

implementation presents significant challenges. The interactive nature of RL demands low-latency response generation throughout training, which consumes substantial GPU memory and necessitates specialized acceleration techniques.

The computational complexity differences between sampling and training are particularly striking. During sampling, the auto-regressive nature of Transformers is inherently sequential, resulting in $\Theta(T)$ time complexity to generate T tokens, as it requires T separate forward passes. This contrasts sharply with model updates, where the gradients for all T tokens can be computed in parallel with a single forward pass, achieving $\Theta(1)$ complexity in terms of passes. We review several key optimization directions below.

- **Caching and Memory Management.** KV caching is indispensable for accelerating generation, as it stores previously computed key–value pairs to avoid redundant computation (Pope et al., 2023). However, as memory demands scale, efficient cache management becomes increasingly critical. Modern inference systems such as vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024b) employ techniques like paged attention (Kwon et al., 2023) to improve cache scheduling and utilization. Recent work further explores prefilling/decoding disaggregation (Qin et al., 2024; Chen et al., 2025c), which distributes prefilling and decoding across separate instances to enhance overall throughput.
- **Algorithmic Acceleration.** As context lengths grow, algorithmic improvements for Transformer inference become increasingly vital. Notable advances include speculative decoding (Chen et al., 2023; Xia et al., 2025), which leverages smaller models (or auxiliary prediction heads) to propose candidate tokens verified by larger models, and lookahead decoding (Fu et al., 2024), which reformulates generation as a nonlinear system solved with Jacobian iteration, enabling simultaneous decoding of multiple tokens.
- **Improved Architectures.** To mitigate the sequential bottleneck of auto-regressive generation, researchers are exploring parallel sampling approaches inspired by diffusion models (Lou et al., 2023; Song et al., 2025; Deepmind, 2025). Within Transformer components, attention remains the primary computational bottleneck for long contexts, motivating more efficient designs. Examples include parameter-sharing methods like grouped query attention (Ainslie et al., 2023), dimensionality-reduction techniques such as multi-head latent attention (Liu et al., 2024) and linear attention (Katharopoulos et al., 2020; Li et al., 2025a), and sparsity-aware approaches that exploit the inherent structure of attention weights (Yuan et al., 2025a; Lu et al., 2025).

Beyond module-specific optimizations, algorithm-hardware co-design approaches exemplified by FlashAttention (Dao et al., 2022) leverage hardware properties such as memory hierarchy to achieve substantial speedups, inspiring broader hardware-aware optimization strategies (Zhao et al., 2025a). We acknowledge that the above review covers only a subset of active research directions in inference acceleration. For comprehensive coverage of this rapidly evolving field, we refer readers to recent surveys (Zhou et al., 2024; Li et al., 2024a; Park et al., 2025).

4.3 System Integration

The previous sections examined techniques for model updating and response sampling in isolation, each employing distinct systems and optimization strategies. However, integrating these components into a unified large-scale system presents substantial coordination challenges that require sophisticated distributed training infrastructure.

Centralized Orchestration. Effective integration demands a centralized orchestration framework capable of scheduling model updates and response sampling while dynamically managing GPU memory allocation across the distributed system. Furthermore, it also manages the model parameter synchronization between training engine and inference engine. Modern implementations such as OpenRLHF (Hu et al., 2024), Verl (Sheng et al., 2024), and ReaL (Mei et al., 2024) demonstrate this architectural approach, providing coordinated resource management and workload distribution.

Limitations of Synchronous Training. The systems described above primarily employ synchronous training paradigms, where model updating and response sampling proceed in strict sequential order. This approach is sub-optimal in practice due to inherent variability in sampling times across different prompts,

leading to resource underutilization and synchronization bottlenecks. The problem becomes particularly acute for complex tasks such as code generation, where reward computation may involve executing multiple unit tests, significantly extending the feedback loop and exacerbating idle GPU time (Luo et al., 2025).

Asynchronous Training Paradigms. To address these inefficiencies, asynchronous training architectures have emerged as a promising alternative, enabling partial rollouts, concurrent execution of response sampling and reward calculation (Fu et al., 2025; Kimi et al., 2025; Xia et al., 2025). By decoupling these processes, asynchronous systems can eliminate synchronization bubbles and achieve higher resource utilization.

However, this architectural shift introduces fundamental algorithmic challenges. Asynchronous training inherently creates off-policy learning scenarios where the policy generating training data differs from the policy being updated, potentially leading to training instability and convergence issues. Addressing these challenges requires drawing from established techniques in classical RL literature, including methods for off-policy learning (Mnih et al., 2016), safe policy updates that bound deviation from reference policies (Munos et al., 2016), and divergence control mechanisms that prevent catastrophic policy degradation (Wang et al., 2019; Liang et al., 2025). The tension between computational efficiency and algorithmic stability in asynchronous RL systems represents a rapidly evolving research frontier, with significant opportunities for innovation spanning both system architecture and algorithmic design.

5 Discussion and Future Directions

Throughout this survey, we primarily examine the algorithmic and computational aspects of RL for LLMs. We delve into several important discussions not extensively covered in previous sections. The first topic is the foundational capability of LLMs trained by RL and its scaling law. The second topic addresses the high-level features and objectives we aim to achieve through RL training. The final topic explores advanced capabilities that we seek to incorporate for real-world applications.

What can RL actually teach a pretrained language model? Based on well-pretrained LLMs, we know that RL can fine-tune models to enhance cognitive abilities such as chain-of-thought reasoning (Wei et al., 2022b), tool usage (Li et al., 2025e), and self-reflection in test-time scaling (OpenAI, 2024; Guo et al., 2025). Crucially, by fine-tuning, we refer to using a relatively small compute budget (e.g., less than 1%) compared with pre-training. Two fundamental questions emerge: whether we can further scale this post-training compute? and what new behaviors RL enables that differ from pre-training alone?

Regarding this fundamental question, Zhou et al. (2023) were the first to investigate this and proposed the famous “superficial alignment hypothesis." They argued that “a model’s knowledge and capabilities are learned almost entirely during pre-training, while alignment teaches it which subdistribution of formats to use when interacting with users." Their compelling evidence came from demonstrating that a model trained via SFT with approximately 1,000 carefully structured samples could generate human-like responses—suggesting that models quickly learn stylistic conventions while drawing their substantive knowledge from pretraining (Lin et al., 2023a).

This hypothesis found further empirical support in mathematical reasoning domains. Studies such as DeepSeekMath (Shao et al., 2024) revealed an interesting pattern: while RL training did not increase the model’s "ceiling performance" (defined by pass@K), meaning it could not solve fundamentally harder problems than before, it dramatically improved pass@1 rates, indicating that models became significantly better at selecting their optimal response on the first attempt. This suggested that rather than expanding raw problem-solving capabilities, RL functioned as a sophisticated confidence calibrator, teaching models to distill their best reasoning into their default response and making greedy generation more reliable. This phenomenon has been systematically investigated by Yue et al. (2025).

However, we note that these “negative results" have limitations, as indicated by emerging studies. First, treating pass@K based on final answer correctness as the sole metric of ceiling performance has inherent flaws. This is because a response may involve process errors (Zheng et al., 2024a) or yield the correct outcome merely by chance. Thus, pass@K could be overestimated when used to measure "ceiling performance." An interesting study by Wen et al. (2025) showed that RL can, in fact, improve fundamental capabilities. For

instance, they demonstrated that RL not only improves outcome correctness but also reduces process errors, despite receiving only outcome-based rewards. Second, existing "negative results" often operate within a small compute regime. This "small" regime encompasses several factors: limited data size (usually 10K-100K samples), restricted exploration (sampling 8, 16, or 32 responses per prompt), and few training iterations (usually several epochs). Collectively, these factors suggest that current RL training paradigms are often insufficient. Several recent works (Liu et al., 2025b) have extensively expanded computational resources for RL and observed promising results. Despite this, we would like to remark that existing RL algorithms also have limitations in exploration, preventing the model's full capabilities from being unlocked. To illustrate, hard tasks (e.g., winning in the game of Go, writing a competition-level math solution, or solving an open question in a scientific domain) may require hundreds or thousands of trials and errors to find a correct solution. If limited sampling is used, the received rewards are often all negative, leading to zero training gradients. Furthermore, current infrastructure often does not adequately support such extensive exploration, thereby limiting the capabilities that RL can unlock. In summary, we expect enhanced RL to unlock new capabilities in the post-training process by interacting with environments and receiving feedback not covered by pre-training. This calls for algorithmic, training infrastructure, and evaluation advances in this direction.

Algorithmic Features for Future Development. Following the previous discussion, as RL algorithms become more stable and training infrastructure improves, several key directions emerge for algorithmic advancement. We highlight two critical areas:

- **Efficient and Effective Exploration.** Current RL methods for LLMs often suffer from inefficient exploration of the vast action space defined by token sequences. Consequently, the trials attempted may receive entirely negative outcome feedback, preventing the model from learning anything. To address this, extensive exploration strategies are crucial. Future algorithms must develop more advanced exploration techniques that can navigate the high-dimensional discrete space of language generation while maintaining training stability. This could be achieved through compact state/action space representations (Barrault et al., 2024; Jia et al., 2025), leveraging more structured exploration spaces (e.g., tree-based approaches) (Zheng et al., 2025b; Li et al., 2025f), or exploiting the unique properties of LLMs with hint-guided exploration techniques (Huang et al., 2025b; Zhang et al., 2025b).
- **Learning from Offline Data.** While online RL shows promise, the expensive cost of online exploration (both computational and monetary, due to large model sizes and slow simulation) makes it prohibitively expensive in practice. Furthermore, online RL algorithms often do not fully exploit the data they collect; for instance, algorithms like PPO and GRPO typically train on samples once and then discard them. Given that the online RL process can be viewed as a data collection process where mixed-quality data is accumulated, it becomes valuable to learn from this kind of experience data (Levine et al., 2020; Sinha et al., 2022; Li et al., 2023). There has been no extensive investigation of this approach in LLMs yet, with SFT (which typically involves imitation from high-quality, curated data) remaining dominant. However, leveraging such data will be crucial for future developments in the field (Silver & Sutton, 2025).
- **Off-policy Learning.** Existing RL approaches for LLMs primarily rely on on-policy learning, where data is collected directly from the current policy. Breaking this limitation offers several significant advantages. First, as discussed, effectively learning from offline data inherently requires off-policy capabilities to maximize the utility of past experiences. Second, from a training system perspective, asynchronous training—which we previously noted can improve GPU utilization efficiency—naturally leads to off-policy data collection. While off-policy learning has historically introduced instability issues (Kumar et al., 2019; Hilton et al., 2022), these challenges could potentially be mitigated by leveraging the unique characteristics of LLMs (Fu et al., 2025).

Emerging Capabilities for Real-World Applications Pre-trained LLMs are primarily based on web text corpora and, as a result, cannot possibly encompass all scenarios encountered in physical and interactive environments, for which pre-training provides limited coverage. This scenario requires LLMs to leverage external tools and receive more external feedback, rendering RL increasingly necessary. We highlight several topics below.

- **Tool Usage and External Integration.** For complex tasks (such as fixing the issue in an Github repo (Jimenez et al., 2023)), solving problems solely within the token space can be both challenging and computationally prohibitive for LLMs. A promising strategy involves offloading specific subproblems to external tools, thereby abstracting low-level reasoning steps into tool calls and reducing the model's reasoning complexity (Shen, 2024; Jiang et al., 2024). This approach not only simplifies the training process but also enhances performance by mitigating hallucinations and producing more reliable outputs (Béchard & Ayala, 2024; Gao et al., 2023; Huynh & Lin, 2025). However, LLMs do not inherently learn effective tool invocation strategies through pretraining alone. Post-training techniques, particularly RL, offer a pathway to teach models efficient tool usage, enabling them to learn when and how to delegate subtasks appropriately. This requires LLMs not only to recognize their own limitations but also to strategically delegate subtasks to appropriate external tools or resources. Recent efforts have focused on training LLMs to interact with tools through long-chain-of-thought frameworks using RL, which help mitigate spurious or inefficient calls during complex problem-solving (Jin et al., 2025; Chen et al., 2025a; Li et al., 2025b;e; Zheng et al., 2025c; Xu & Peng, 2025). However, current explorations remain in early stages, often validating effectiveness within isolated, single-tool scenarios. How to leverage diverse tool ecosystems to solve more general and complex problems remains an open and important research question.

- **Multi-Agent Coordination and Collaboration.** The future of RL-enhanced LLMs extends beyond individual agent capabilities to multi-agent coordination. As multiple AI agents begin collaborating on complex tasks, RL becomes essential for learning effective communication protocols, task decomposition strategies, and coordination mechanisms. This includes learning how to negotiate, delegate responsibilities, resolve conflicts, and maintain coherent joint objectives across multiple interacting agents. Multi-agent RL for LLMs presents unique challenges, including the need for agents to develop shared communication languages, understand others' capabilities and limitations, and adapt their behavior based on the evolving dynamics of the agent collective (Cemri et al., 2025; Yang et al., 2025c; Tan et al., 2025; Tran et al., 2025). These capabilities are crucial for deploying LLM-based systems in real-world scenarios where multiple AI agents must work together or interact with human users in collaborative settings.

- **Multi-Modal RL and Physical World Interaction.** The integration of multi-modal capabilities represents a critical frontier for RL-enhanced LLMs (Huang et al., 2025a). Image and video understanding enable models to perceive and interact with the physical world in ways that pure text-based systems cannot achieve (Liu et al., 2023; Bai et al., 2023). Audio and speech processing further enhance LLMs' ability to engage seamlessly with humans, fostering more natural and responsive interfaces (Zeng et al., 2024; Zhang et al., 2025c). Through RL training on multi-modal tasks, LLMs can learn to ground their language understanding in visual and spatial contexts, developing more robust real-world reasoning capabilities (Yang et al., 2025b). Multi-modal RL presents unique challenges in action space design, reward specification, and cross-modal representation learning (Qi et al., 2024a; Saeed et al., 2025; Yang et al., 2025b; Shen et al., 2025; Wang et al., 2025a). As this field rapidly evolves, advances in multi-modal RL continue to push the boundaries of embodied AI.

6 Conclusion

In this survey, we have traced the evolution of reinforcement learning for large language models across three key dimensions. We examined the unique algorithmic challenges posed by LLMs' discrete, high-dimensional nature and the specialized methods developed to address them. We explored the computational frameworks that have made large-scale RL training feasible, enabling the transition from proof-of-concept to production systems. Finally, we engaged with the fundamental debate about post-training's true capabilities: whether RL teaches new knowledge or refines the expression of existing knowledge, and how this understanding shapes our approach to capability development.

As we stand at this inflection point, the field faces exciting challenges around scaling laws, sample efficiency, and evaluation frameworks that capture RL's full impact. We hope this survey serves as both a reference for

newcomers and a catalyst for existing researchers to explore the many open questions that remain. The story of RL for LLMs is still being written, and its most transformative chapters may well lie ahead.

References

Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 32:96, 2019.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, et al. A survey on data selection for language models. *arXiv preprint arXiv:2402.16827*, 2024.

Anthropic. System card: Claude opus 4 & claude sonnet 4. *https://www-cdn.anthropic.com/4263b940cabb546aa0e3283f35b686f4f3b2ff47.pdf*, 2025.

Charles Arnal, GaĂŤtan Narozniak, Vivien Cabannes, Yunhao Tang, Julia Kempe, and Remi Munos. Asymmetric reinforce for off-policy reinforcement learning: Balancing positive and negative rewards. *arXiv preprint arXiv:2506.20520*, 2025.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *International conference on machine learning*, pp. 263–272. PMLR, 2017.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pp. 4447–4455. PMLR, 2024.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Loïc Barrault, Paul-Ambroise Duquenne, Maha Elbayad, Artyom Kozhevnikov, Belen Alastruey, Pierre Andrews, Mariano Coria, Guillaume Couairon, Marta R Costa-jussà, David Dale, et al. Large concept models: Language modeling in a sentence representation space. *arXiv preprint arXiv:2412.08821*, 2024.

Patrice Béchard and Orlando Marquez Ayala. Reducing hallucination in structured outputs via retrieval-augmented generation. *arXiv preprint arXiv:2404.08189*, 2024.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pp. 610–623, 2021.

Alex Beutel, Kai Xiao, Johannes Heidecke, and Lilian Weng. Diverse and effective red teaming with auto-generated rewards and multi-step reinforcement learning. *arXiv preprint arXiv:2412.18693*, 2024.

Johan Bjorck, Alon Benhaim, Vishrav Chaudhary, Furu Wei, and Xia Song. Scaling optimal lr across token horizons. *arXiv preprint arXiv:2409.19913*, 2024.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

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

Qi Cai, Zhuoran Yang, Chi Jin, and Zhaoran Wang. Provably efficient exploration in policy optimization. In *International Conference on Machine Learning*, pp. 1283–1294. PMLR, 2020.

Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

Stephen Casper, Jack Davies, Ziming Shi, Thomas Haber, Abdullah Almaatouq, Mark Riedl, Thomas L Gilbert, Jenny Zheng, David Manheim, Alessandro Flint, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, et al. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, Fan Yang, et al. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025a.

Peter Chen, Xiaopeng Li, Ziniu Li, Xi Chen, and Tianyi Lin. Spectral policy optimization: Coloring your incorrect reasoning in grpo. *arXiv preprint arXiv:2505.11595*, 2025b.

Siyu Chen, Yifei Wang, Yaliang Zhao, Zhen Li, and Yu Zhang. Grpo: Gradient-regularized preference optimization for sample-efficient rlhf. *arXiv preprint arXiv:2402.03300*, 2024.

Xing Chen, Rong Shi, Lu Zhao, Lingbin Wang, Xiao Jin, Yueqiang Chen, and Hongfeng Sun. Disaggregated prefill and decoding inference system for large language model serving on multi-vendor gpus. *arXiv preprint arXiv:2509.17542*, 2025c.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Acereason-nemotron: Advancing math and code reasoning through reinforcement learning. *arXiv preprint arXiv:2505.16400*, 2025d.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in neural information processing systems*, volume 30, 2017.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. Gpg: A simple and strong reinforcement learning baseline for model reasoning. *arXiv preprint arXiv:2504.02546*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025a.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025b.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.

Peter Dayan. Reinforcement comparison. In *Connectionist Models*, pp. 45–51. Elsevier, 1991.

Google Deepmind. Google diffusion. url, 2025. URL `https://deepmind.google/models/gemini-diffusion/`.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025.

Jiangfei Duan, Shuo Zhang, Zerui Wang, Lijuan Jiang, Wenwen Qu, Qinghao Hu, Guoteng Wang, Qizhen Weng, Hang Yan, Xingcheng Zhang, et al. Efficient training of large language models on distributed infrastructures: a survey. *arXiv preprint arXiv:2407.20018*, 2024.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *arXiv preprint arXiv:2505.24298*, 2025.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. Break the sequential dependency of llm inference using lookahead decoding. *arXiv preprint arXiv:2402.02057*, 2024.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Leo Gao, Aman Madaan, Shuyan Zhou, Daniel Kang, Aditya Samudre, Vishakh Padmakumar, Catherine Pryor, Nathan Soboroff, Eric Xi, Yiming Du, et al. Scaling laws for reward model overoptimization. *arXiv preprint arXiv:2210.10760*, 2022.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

Zhaolin Gao, Jonathan Chang, Wenhao Zhan, Owen Oertell, Gokul Swamy, Kianté Brantley, Thorsten Joachims, Drew Bagnell, Jason D Lee, and Wen Sun. Rebel: Reinforcement learning via regressing relative rewards. *Advances in Neural Information Processing Systems*, 37:52354–52400, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530, 2004.

Melody Y Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Heylar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, et al. Deliberative alignment: Reasoning enables safer language models. *arXiv preprint arXiv:2412.16339*, 2024.

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

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Yaru Hao, Li Dong, Xun Wu, Shaohan Huang, Zewen Chi, and Furu Wei. On-policy rl with optimal reward baseline. *arXiv preprint arXiv:2505.23585*, 2025.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, et al. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*, 2025.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Jacob Hilton, Karl Cobbe, and John Schulman. Batch size-invariance for policy optimization. *Advances in Neural Information Processing Systems*, 35:17086–17098, 2022.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Jian Hu, Xibin Wu, Zilin Zhu, Weixun Wang, Dehao Zhang, Yu Cao, et al. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. Reinforce++: An efficient rlhf algorithm with robustness to both prompt and reward models. *arXiv preprint arXiv:2501.03262*, 2025.

Shengyi Huang, Rousslan Fernand Julien Dossa, Antonin Raffin, Anssi Kanervisto, and Weixun Wang. The 37 implementation details of proximal policy optimization. *The ICLR Blog Track 2023*, 2022.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025a.

Zeyu Huang, Tianhao Cheng, Zihan Qiu, Zili Wang, Yinghui Xu, Edoardo M Ponti, and Ivan Titov. Blending supervised and reinforcement fine-tuning with prefix sampling. *arXiv preprint arXiv:2507.01679*, 2025b.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Nam Huynh and Beiyu Lin. Large language models for code generation: A comprehensive survey of challenges, techniques, evaluation, and applications. *arXiv preprint arXiv:2503.01245*, 2025.

Chengxing Jia, Ziniu Li, Pengyuan Wang, Yi-Chen Li, Zhenyu Hou, Yuxiao Dong, and Yang Yu. Controlling large language model with latent actions. *arXiv preprint arXiv:2503.21383*, 2025.

Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*, 2024.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Dahyun Kim, Yungi Kim, Wonho Song, Hyeonwoo Kim, Yunsu Kim, Sanghoon Kim, and Chanjun Park. sdpo: Don't use your data all at once. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pp. 366–373, 2025a.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025b.

Team Kimi, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! 2019.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Dirk P Kroese, Thomas Taimre, and Zdravko I Botev. *Handbook of Monte Carlo Methods*. John Wiley & Sons, 2013.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025a.

Chengpeng Li, Zhengyang Tang, Ziniu Li, Mingfeng Xue, Keqin Bao, Tian Ding, Ruoyu Sun, Benyou Wang, Xiang Wang, Junyang Lin, et al. Cort: Code-integrated reasoning within thinking. *arXiv preprint arXiv:2506.09820*, 2025b.

Houyi Li, Wenzhen Zheng, Jingcheng Hu, Qiufeng Wang, Hanshan Zhang, Zili Wang, Shijie Xuyang, Yuantao Fan, Shuigeng Zhou, Xiangyu Zhang, et al. Predictable scale: Part i–optimal hyperparameter scaling law in large language model pretraining. *arXiv preprint arXiv:2503.04715*, 2025c.

Jinhao Li, Jiaming Xu, Shan Huang, Yonghua Chen, Wen Li, Jun Liu, Yaoxiu Lian, Jiayi Pan, Li Ding, Hao Zhou, et al. Large language model inference acceleration: A comprehensive hardware perspective. *arXiv preprint arXiv:2410.04466*, 2024a.

Qian Li, Ziniu Li, Tian Ding, and Ruoyu Sun. Position: Transformers have the potential to achieve agi. 2025d. URL `https://openreview.net/forum?id=vMTijVnXQ8`.

Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, et al. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*, 2020.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. Torl: Scaling tool-integrated rl. *arXiv preprint arXiv:2503.23383*, 2025e.

Yingru Li. Logit dynamics in softmax policy gradient methods. *arXiv preprint arXiv:2506.12912*, 2025.

Yizhi Li, Qingshui Gu, Zhoufutu Wen, Ziniu Li, Tianshun Xing, Shuyue Guo, Tianyu Zheng, Xin Zhou, Xingwei Qu, Wangchunshu Zhou, et al. Treepo: Bridging the gap of policy optimization and efficacy and inference efficiency with heuristic tree-based modeling. *arXiv preprint arXiv:2508.17445*, 2025f.

Ziniu Li, Tian Xu, Zeyu Qin, Yang Yu, and Zhi-Quan Luo. Imitation learning from imperfection: Theoretical justifications and algorithms. *Advances in Neural Information Processing Systems*, 36:18404–18443, 2023.

Ziniu Li, Tian Xu, and Yang Yu. When is rl better than dpo in rlhf? a representation and optimization perspective. In *The Second Tiny Papers Track at ICLR 2024*, 2024b.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *International Conference on Machine Learning*, 2024c.

Ziniu Li, Congliang Chen, Tian Xu, Zeyu Qin, Jiancong Xiao, Zhi-Quan Luo, and Ruoyu Sun. Preserving diversity in supervised fine-tuning of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025g.

Jing Liang, Hongyao Tang, Yi Ma, Jinyi Liu, Yan Zheng, Shuyue Hu, Lei Bai, and Jianye Hao. Squeeze the soaked sponge: Efficient off-policy reinforcement finetuning for large language model. *arXiv preprint arXiv:2507.06892*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harri Edwards, Bowen Baker, Stephanie Lin, Jack Parker-Holder, Isaac Betts, Phoenix Bridge, Tom Vred, et al. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023a.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023b.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. The unlocking spell on base llms: Rethinking alignment via in-context learning. *arXiv preprint arXiv:2312.01552*, 2023a.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang, Wenbin Hu, Hanning Zhang, et al. Mitigating the alignment tax of rlhf. *arXiv preprint arXiv:2309.06256*, 2023b.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Kaizhao Liu, Qi Long, Zhekun Shi, Weijie J Su, and Jiancong Xiao. Statistical impossibility and possibility of aligning llms with human preferences: From condorcet paradox to nash equilibrium. *arXiv preprint arXiv:2503.10990*, 2025a.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025b.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. `https://github.com/sail-sg/understand-r1-zero`, 2025c.

Zihe Liu, Jiashun Liu, Yancheng He, Weixun Wang, Jiaheng Liu, Ling Pan, Xinyu Hu, Shaopan Xiong, Ju Huang, Jian Hu, et al. Part i: Tricks or traps? a deep dive into rl for llm reasoning. *arXiv preprint arXiv:2508.08221*, 2025d.

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling. *arXiv preprint arXiv:2504.02495*, 2025e.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

Xingzhou Lou, Junge Zhang, Jian Xie, Lifeng Liu, Dong Yan, and Kaiqi Huang. Sequential preference optimization: Multi-dimensional preference alignment with implicit reward modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 27509–27517, 2025.

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, Weiran He, Enming Yuan, Yuzhi Wang, et al. Moba: Mixture of block attention for long-context llms. *arXiv preprint arXiv:2502.13189*, 2025.

Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepcoder: A fully open-source 14b coder at o3-mini level. `https://pretty-radio-b75.notion.site/DeepCoder-A-Fully-Open-Source-14B-Coder-at-O3-mini-Level-1cf81902c14680b3bee5eb349a512a51`, 2025. Notion Blog.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative reward models. *arXiv preprint arXiv:2410.12832*, 2024.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A Smith, Hannaneh Hajishirzi, and Nathan Lambert. Rewardbench 2: Advancing reward model evaluation. *arXiv preprint arXiv:2506.01937*, 2025.

Zhiyu Mei, Wei Fu, Kaiwei Li, Guangju Wang, Huanchen Zhang, and Yi Wu. Real: Efficient rlhf training of large language models with parameter reallocation. *arXiv preprint arXiv:2406.14088*, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024.

AI Meta. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. *https://ai.meta. com/blog/llama-4-multimodal-intelligence/, checked on*, 4(7):2025, 2025.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PmLR, 2016.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Deepak Narayanan, Amar Phanishayee, Kaiyu Shi, Xie Chen, and Matei Zaharia. Memory-efficient pipeline-parallel dnn training. In *International Conference on Machine Learning*, pp. 7937–7947. PMLR, 2021.

Andi Nika, Debmalya Mandal, Parameswaran Kamalaruban, Georgios Tzannetos, Goran Radanović, and Adish Singla. Reward model learning vs. direct policy optimization: A comparative analysis of learning from human preferences. *arXiv preprint arXiv:2403.01857*, 2024.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

OpenAI. Learning to reason with LLMs. OpenAI Blog, Feb 2024. `https://openai.com/index/learning-to-reason-with-llms`.

OpenAI. Gpt-5 system card. *https://openai.com/index/gpt-5-system-card/*, 2025.

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

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*, 2024.

Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization. In *Advances in Neural Information Processing Systems 38*, 2024.

Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In *International conference on machine learning*, pp. 4026–4035. PMLR, 2018.

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization. *arXiv preprint arXiv:2403.19159*, 2024.

Sihyeong Park, Sungryeol Jeon, Chaelyn Lee, Seokhun Jeon, Byung-Soo Kim, and Jemin Lee. A survey on inference engines for large language models: Perspectives on optimization and efficiency. *arXiv preprint arXiv:2505.01658*, 2025.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5:606–624, 2023.

Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

Ji Qi, Ming Ding, Weihan Wang, Yushi Bai, Qingsong Lv, Wenyi Hong, Bin Xu, Lei Hou, Juanzi Li, Yuxiao Dong, et al. Cogcom: A visual language model with chain-of-manipulations reasoning. *arXiv preprint arXiv:2402.04236*, 2024a.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. *arXiv preprint arXiv:2406.05946*, 2024b.

Ruoyu Qin, Zheming Li, Weiran He, Mingxing Zhang, Yongwei Wu, Weimin Zheng, and Xinran Xu. Mooncake: A kvcache-centric disaggregated architecture for llm serving. *arXiv preprint arXiv:2407.00079*, 2024.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sushil Sikchi, Joey Hejna, Brad Knox, Chelsea Finn, and Scott Niekum. Scaling laws for reward model overoptimization in direct alignment algorithms. *Advances in Neural Information Processing Systems 37*, pp. 126207–126242, 2024a.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From r to $q*$: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*, 2024b.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16. IEEE, 2020.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. *arXiv preprint arXiv:2210.01241*, 2022.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Pierre Harvey Richemond, Yunhao Tang, Daniel Guo, Daniele Calandriello, Mohammad Gheshlaghi Azar, Rafael Rafailov, Bernardo Avila Pires, Eugene Tarassov, Lucas Spangher, Will Ellsworth, et al. Offline regularised reinforcement learning for large language models alignment. *arXiv preprint arXiv:2405.19107*, 2024.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Shaheer U Saeed, Yipei Wang, Veeru Kasivisvanathan, Brian R Davidson, Matthew J Clarkson, Yipeng Hu, and Daniel C Alexander. Reasoning in machine vision: learning to think fast and slow. *arXiv preprint arXiv:2506.22075*, 2025.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the 4th International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Lior Shani, Aviv Rosenberg, Asaf Cassel, Oran Lang, Daniele Calandriello, Avital Zipori, Hila Noga, Orgad Keller, Bilal Piot, Idan Szpektor, et al. Multi-turn reinforcement learning from preference human feedback. *arXiv preprint arXiv:2405.14655*, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Gerald Shen, Zhilin Wang, Olivier Delalleau, Jiaqi Zeng, Yi Dong, Daniel Egert, Shengyang Sun, Jimmy Zhang, Sahil Jain, Ali Taghibakhshi, et al. Nemo-aligner: Scalable toolkit for efficient model alignment. *arXiv preprint arXiv:2405.01481*, 2024.

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, et al. Vlm-r1: A stable and generalizable r1-style large vision-language model. *arXiv preprint arXiv:2504.07615*, 2025.

Zhuocheng Shen. Llm with tools: A survey. *arXiv preprint arXiv:2409.18807*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

David Silver and Richard S Sutton. Welcome to the era of experience. *Google AI*, 1, 2025.

Joykirat Singh, Raghav Magazine, Yash Pandya, and Akshay Nambi. Agentic reasoning and tool integration for llms via reinforcement learning. *arXiv preprint arXiv:2505.01441*, 2025.

Samarth Sinha, Ajay Mandlekar, and Animesh Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pp. 907–917. PMLR, 2022.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471, 2022.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

Ziang Song, Tianle Cai, Jason D Lee, and Weijie J Su. Reward collapse in aligning large language models. *arXiv preprint arXiv:2305.17608*, 2023.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021, 2020.

Ruoyu Sun. Optimization for deep learning: theory and algorithms. *arXiv preprint arXiv:1912.08957*, 2019.

Richard Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44, 1988.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Gokul Swamy, Christoph Dann, Rahul Kidambi, Zhiwei Steven Wu, and Alekh Agarwal. A minimaximalist approach to reinforcement learning from human feedback. *arXiv preprint arXiv:2401.04056*, 2024.

Zhi-Hao Tan, Zi-Chen Zhao, Hao-Yu Shi, Xin-Yu Zhang, Peng Tan, Yang Yu, and Zhi-Hua Zhou. Learnware of language models: Specialized small language models can do big. *arXiv preprint arXiv:2505.13425*, 2025.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, volume 30, 2017.

Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenburg, Tim Verbelen, and Jan S Rellermeyer. A survey on distributed machine learning. *Acm computing surveys (csur)*, 53(2):1–33, 2020.

Christian Walder and Deep Karkhanis. Pass@ k policy optimization: Solving harder reinforcement learning problems. *arXiv preprint arXiv:2505.15201*, 2025.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*, 2024.

Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. Vl-rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *arXiv preprint arXiv:2504.08837*, 2025a.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023.

Peng-Yuan Wang, Tian-Shuo Liu, Chenyang Wang, Yi-Di Wang, Shu Yan, Cheng-Xing Jia, Xu-Hui Liu, Xin-Wei Chen, Jia-Cheng Xu, Ziniu Li, et al. A survey on large language models for mathematical reasoning. *arXiv preprint arXiv:2506.08446*, 2025b.

Qing Wang, Yingru Li, Jiechao Xiong, and Tong Zhang. Divergence-augmented policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025c.

Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, Yancheng He, Ju Huang, Jiaheng Liu, Zhendong Li, Xiaoyang Li, et al. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library. *arXiv preprint arXiv:2506.06122*, 2025d.

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

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022a.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022b.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.

Xumeng Wen, Zihan Liu, Shun Zheng, Zhijian Xu, Shengyu Ye, Zhirong Wu, Xiao Liang, Yang Wang, Junjie Li, Ziming Miao, et al. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*, 2025.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Bo Wu, Sid Wang, Yunhao Tang, Jia Ding, Eryk Helenowski, Liang Tan, Tengyu Xu, Tushar Gowda, Zhengxing Chen, Chen Zhu, et al. Llamarl: A distributed asynchronous reinforcement learning framework for efficient large-scale llm trainin. *arXiv preprint arXiv:2505.24034*, 2025.

Junkang Wu, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. *beta*-dpo: Direct preference optimization with dynamic *beta*. *Advances in Neural Information Processing Systems*, 37:129944–129966, 2024a.

Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. *arXiv preprint arXiv:2405.00675*, 2024b.

Bingquan Xia, Bowen Shen, Dawei Zhu, Di Zhang, Gang Wang, Hailin Zhang, Huaqiu Liu, Jiebao Xiao, Jinhao Dong, Liang Zhao, et al. Mimo: Unlocking the reasoning potential of language model–from pretraining to posttraining. *arXiv preprint arXiv:2505.07608*, 2025.

Jiancong Xiao, Ziniu Li, Xingyu Xie, Emily Getzen, Cong Fang, Qi Long, and Weijie J Su. On the algorithmic bias of aligning large language models with rlhf: Preference collapse and matching regularization. *arXiv preprint arXiv:2405.16455*, 2024.

Lechao Xiao. Rethinking conventional wisdom in machine learning: From generalization to scaling. *arXiv preprint arXiv:2409.15156*, 2024.

Youshao Xiao, Zhenglei Zhou, Fagui Mao, Weichang Wu, Shangchun Zhao, Lin Ju, Lei Liang, Xiaolu Zhang, and Jun Zhou. An adaptive placement and parallelism framework for accelerating rlhf training. *arXiv preprint arXiv:2312.11819*, 2023.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. *arXiv preprint arXiv:2312.11456*, 2023.

Wei Xiong, Chengshuai Shi, Jiaming Shen, Aviv Rosenberg, Zhen Qin, Daniele Calandriello, Misha Khalman, Rishabh Joshi, Bilal Piot, Mohammad Saleh, et al. Building math agents with multi-turn iterative preference learning. *arXiv preprint arXiv:2409.02392*, 2024.

Pan Xu, Felicia Gao, and Quanquan Gu. Sample efficient policy gradient methods with recursive variance reduction. *arXiv preprint arXiv:1909.08610*, 2019.

Renjun Xu and Jingwen Peng. A comprehensive survey of deep research: Systems, methodologies, and applications. *arXiv preprint arXiv:2506.12594*, 2025.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study. *arXiv preprint arXiv:2404.10719*, 2024.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Greg Yang, Edward J Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466*, 2022.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652, 2024b.

Tianyun Yang, Yunwen Li, Ziniu Li, Zhihang Lin, Ruoyu Sun, and Tian Ding. Bridging formal language with chain-of-thought reasoning to geometry problem solving. *arXiv preprint arXiv:2508.09099*, 2025b.

Yingxuan Yang, Huacan Chai, Shuai Shao, Yuanyi Song, Siyuan Qi, Renting Rui, and Weinan Zhang. Agentnet: Decentralized evolutionary coordination for llm-based multi-agent systems. *arXiv preprint arXiv:2504.00587*, 2025c.

Zhihe Yang, Xufang Luo, Zilong Wang, Dongqi Han, Zhiyuan He, Dongsheng Li, and Yunjian Xu. Do not let low-probability tokens over-dominate in rl for llms. *arXiv preprint arXiv:2505.12929*, 2025d.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023a.

Zhewei Yao, Reza Yazdani Aminabadi, Olatunji Ruwase, Samyam Rajbhandari, Xiaoxia Wu, Ammar Ahmad Awan, Jeff Rasley, Minjia Zhang, Conglong Li, Connor Holmes, et al. Deepspeed-chat: Easy, fast and affordable rlhf training of chatgpt-like models at all scales. *arXiv preprint arXiv:2308.01320*, 2023b.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, et al. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent. *arXiv preprint arXiv:2507.02259*, 2025a.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025b.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, YX Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025a.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025b.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Aohan Zeng, Zhengxiao Du, Mingdao Liu, Kedong Wang, Shengmin Jiang, Lei Zhao, Yuxiao Dong, and Jie Tang. Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot. *arXiv preprint arXiv:2412.02612*, 2024.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, and Mingyi Hong. Reinforcing multi-turn reasoning in llm agents via turn-level credit assignment. *arXiv preprint arXiv:2505.11821*, 2025a.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025b.

Kaichen Zhang, Yuzhong Hong, Junwei Bao, Hongfei Jiang, Yang Song, Dingqian Hong, and Hui Xiong. Gvpo: Group variance policy optimization for large language model post-training. *arXiv preprint arXiv:2504.19599*, 2025a.

Kaiyi Zhang, Ang Lv, Jinpeng Li, Yongbo Wang, Feng Wang, Haoyuan Hu, and Rui Yan. Stephint: Multi-level stepwise hints enhance reinforcement learning to reason. *arXiv preprint arXiv:2507.02841*, 2025b.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*, 2024.

Xueyao Zhang, Yuancheng Wang, Chaoren Wang, Ziniu Li, Zhuo Chen, and Zhizheng Wu. Advancing zero-shot text-to-speech intelligibility across diverse domains via preference alignment. *arXiv preprint arXiv:2505.04113*, 2025c.

Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Huazuo Gao, Jiashi Li, Liyue Zhang, Panpan Huang, Shangyan Zhou, Shirong Ma, et al. Insights into deepseek-v3: Scaling challenges and reflections on hardware for ai architectures. *arXiv preprint arXiv:2505.09343*, 2025a.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–38, 2024.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, et al. Geometric-mean policy optimization. *arXiv preprint arXiv:2507.20673*, 2025b.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*, 2024a.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025a.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Livia Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in Neural Information Processing Systems*, 37:62557–62583, 2024b.

Tianyu Zheng, Tianshun Xing, Qingshui Gu, Taoran Liang, Xingwei Qu, Xin Zhou, Yizhi Li, Zhoufutu Wen, Chenghua Lin, Wenhao Huang, et al. First return, entropy-eliciting explore. *arXiv preprint arXiv:2507.07017*, 2025b.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025c.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021, 2023.

Zixuan Zhou, Xuefei Ning, Ke Hong, Tianyu Fu, Jiaming Xu, Shiyao Li, Yuming Lou, Luning Wang, Zhihang Yuan, Xiuhong Li, et al. A survey on efficient inference for large language models. *arXiv preprint arXiv:2404.14294*, 2024.

Banghua Zhu, Michael Jordan, and Jiantao Jiao. Principled reinforcement learning with human feedback from pairwise or k-wise comparisons. In *International Conference on Machine Learning*, pp. 43037–43067. PMLR, 2023.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

Barret Zoph and John Schulman. Chatgpt and the art of post-training. Google Docs, 2025. URL `https://docs.google.com/presentation/d/11KWCKUORnPpVMSY6vXgBeFSWo7fJcuGQ9yuR6vC1pzE/edit?usp=sharing`.