A SURVEY ON POST-TRAINING OF LARGE LANGUAGE MODELS

Guiyao Tie[1†]   **Zeli Zhao**[1]   **Dingjie Song**[2]   **Fuyang Wei**[3]   **Rong Zhou**[2]   **Yurou Dai**[2]
Wen Yin[1]   **Zhejian Yang**[4]   **Jiangyue Yan**[5]   **Yao Su**[6]   **Zhenhan Dai**[1]   **Yifeng Xie**[1]
Yihan Cao[7]   **Lichao Sun**[2]   **Pan Zhou**[1]   **Lifang He**[2]   **Hechang Chen**[4]   **Yu Zhang**[5]
Qingsong Wen[8]   **Tianming Liu**[9]   **Neil Zhenqiang Gong**[10]   **Jiliang Tang**[11]
Caiming Xiong[12]   **Heng Ji**[13]   **Philip S. Yu**[14]   **Jianfeng Gao**[15]

[1]**Huazhong University of Science and Technology**   [2]**Lehigh University**
[3]**The University of Hong Kong**   [4]**Jilin University**   [5]**Southern University of Science and Technology**
[6]**Worcester Polytechnic Institute**   [7]**LinkedIn Corporation**
[8]**Squirrel Ai Learning**   [9]**University of Georgia**   [10]**Duke University**
[11]**Michigan State University**   [12]**Salesforce Research**   [13]**University of Illinois Urbana-Champaign**
[14]**University of Illinois at Chicago**   [15]**Microsoft Research**

ABSTRACT

The emergence of Large Language Models (LLMs) has fundamentally transformed natural language processing, making them indispensable across domains ranging from conversational systems to scientific exploration. However, their pre-trained architectures often reveal limitations in specialized contexts, including restricted reasoning capacities, ethical uncertainties, and suboptimal domain-specific performance. These challenges necessitate advanced post-training language models (PoLMs) to address these shortcomings, such as OpenAI-o1/o3 and DeepSeek-R1 (collectively known as Large Reasoning Models, or LRMs). This paper presents the first comprehensive survey of PoLMs, systematically tracing their evolution across five core paradigms: **Fine-tuning**, which enhances task-specific accuracy; **Alignment**, which ensures ethical coherence and alignment with human preferences; **Reasoning**, which advances multi-step inference despite challenges in reward design; **Efficiency**, which optimizes resource utilization amidst increasing complexity; and **Integration and Adaptation**, which extend capabilities across diverse modalities while addressing coherence issues. Charting progress from ChatGPT's foundational alignment strategies in 2018 to DeepSeek-R1's innovative reasoning advancements in 2025, we illustrate how PoLMs leverage datasets to mitigate biases, deepen reasoning capabilities, and enhance domain adaptability. Our contributions include a pioneering synthesis of PoLM evolution, a structured taxonomy categorizing techniques and datasets, and a strategic agenda emphasizing the role of LRMs in improving reasoning proficiency and domain flexibility. As the first survey of its scope, this work consolidates recent PoLM advancements and establishes a rigorous intellectual framework for future research, fostering the development of LLMs that excel in precision, ethical robustness, and versatility across scientific and societal applications.

Keywords Post-training, Large Language Model, Fine-Tuning, Alignment, Reasoning, Efficiency.

[†]Guiyao Tie is the current corresponding author: tgy@hust.edu.cn
[‡]Latest Update: Mar., 2025.

Contents

1 Introduction

It is generally agreed upon that authentic intelligence equips us with reasoning capabilities, enables us to test hypotheses, and prepares for future eventualities.

Jean Khalfa – WHAT IS INTELLIGENCE? (1994)

Language models (LMs) [1, 2] represent sophisticated computational frameworks designed to model and generate human language. These models have revolutionized the field of natural language processing (NLP) [3] by enabling machines to understand, generate, and interact with human language in a manner that closely mimics human cognition. Unlike humans, who acquire language skills naturally through interaction and exposure to contextual environments, machines must undergo extensive, data-driven training to develop similar capabilities [4]. This presents a significant research challenge, as enabling machines to comprehend and generate human language, while engaging in natural, contextually appropriate dialogue, requires not only vast computational resources but also refined methodologies for model development [5, 6].

The emergence of Large Language Models (LLMs) such as GPT-3 [7], InstructGPT [8], and GPT-4 [9] has marked a transformative phase in the evolution of LMs. These models, distinguished by their extensive parameterization and advanced learning capabilities, are designed to capture complex linguistic structures, contextual relationships, and nuanced patterns within vast datasets. This enables LLMs not only to predict subsequent words but also to generate coherent, contextually relevant text across a wide range of tasks, including translation, question answering, and summarization. The development of LLMs has sparked significant academic interest [5, 6, 10], which can be divided into two main stages: *pre-training* and *post-training*.

Pre-training. The concept of pre-training originates from transfer learning in computer vision (CV) tasks [10]. Its primary goal is to develop a general model using extensive datasets, which facilitates easy fine-tuning for various downstream applications. A significant advantage of pre-training is its ability to utilize any unlabeled text corpus, thereby providing an abundant source of training data. However, early static pre-training methods, such as Neural Network Language Models (NNLM) [11] and Word2vec [12], struggled to accommodate different textual semantic environments, prompting the development of dynamic pre-training techniques like BERT [2] and XLNet [13]. BERT effectively addressed the limitations of static methods by leveraging the transformer architecture and employing self-attention mechanisms on large-scale unlabeled datasets. This study established the "pre-training and fine-tuning" learning paradigm, inspiring numerous subsequent studies that introduced diverse architectures, including GPT-2 [14] and BART [15].

Post-training. Post-training refers to the techniques and methodologies employed after a model has undergone pre-training, aiming to refine and adapt the model for specific tasks or user requirements. Following the release of GPT-3 [7], with its 175 billion parameters, the field of post-training experienced a significant surge in interest and innovation. Various approaches emerged to enhance model performance, including fine-tuning [16, 17], which adjusts model parameters using labeled datasets or specific task data; alignment strategies [18, 19, 20], which optimize models to better align with user preferences; knowledge adaptation techniques [21, 22], which enable models to incorporate domain-specific knowledge; and reasoning improvements [23, 24], which enhance a model's ability to make logical inferences and decisions. Collectively known as Post-training Language Models (PoLMs), these techniques have led to the development of models such as GPT-4 [9], LLaMA-3 [25], Gemini-2.0 [26], and Claude-3.5 [27], marking substantial progress in LLM capabilities. However, post-trained models often struggle to adapt to new tasks without retraining or significant parameter adjustments, making PTM development an area of active research.

As highlighted, pre-trained language models (PLMs) primarily aim to provide general knowledge and capabilities, while PoLMs focus on adapting these models to specific tasks and requirements. A notable example of this adaptation is the latest LLM, DeepSeek-R1 [28], which illustrates the evolution of PoLMs in enhancing reasoning abilities, aligning with user preferences, and improving adaptability across various domains [29]. Furthermore, the increasing availability of open-sourced LLMs (e.g., LLaMA [30], Gemma [31], and Nemotron [32]) and domain-specific large datasets (e.g., PromptSource [33] and Flan [34]) is driving a
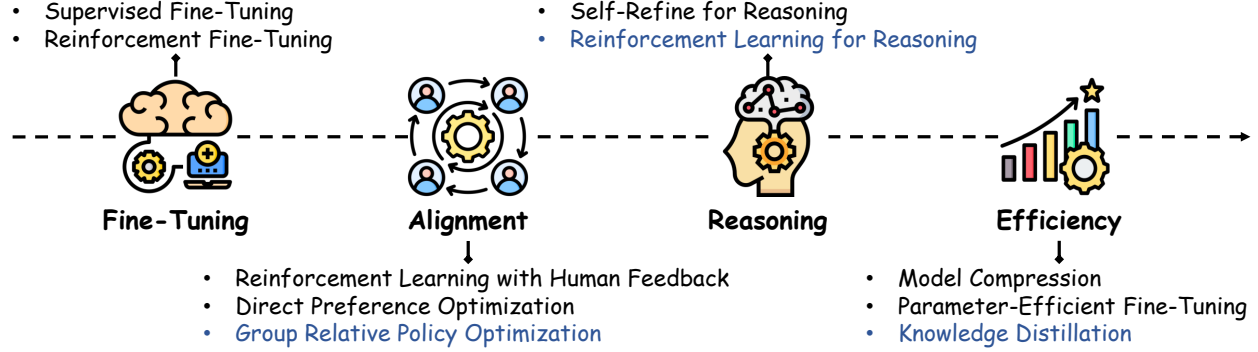
Figure 1: The evolution of post-training techniques for Large Language Models, delineating the progression from initial methodologies to advanced approaches, with emphasis on DeepSeek model contributions (highlighted in blue).

trend among academic researchers and industry practitioners to develop PoLMs. This trend underscores the growing recognition of the importance of tailored adaptations in the field of PoLMs.

In the existing literature, PLMs have been widely discussed and surveyed [10, 35, 36, 37], while PoLMs are seldom reviewed systematically. To advance these techniques, it is essential to thoroughly examine the existing body of research to identify key challenges, gaps, and opportunities for further refinement. This survey aims to fill this gap by providing a structured framework for the evolving research in post-training. As shown in **Fig. 1**, it explores multiple stages of post-training, with a particular focus on those employed **from ChatGPT to DeepSeek**. These techniques encompass a wide range of methodologies, including fine-tuning, LLM alignment, reasoning enhancement, and efficiency improvements. The blue section of the figure specifically highlights the set of post-training methods applied by DeepSeek, emphasizing the innovative strategies that have contributed to its success in adapting to user preferences and domain-specific needs.

1.1 Major Contributions

This paper represents the first comprehensive survey on PoLMs, providing a thorough, structured exploration of the latest advancements in the field. While previous surveys have typically focused on specific aspects of LLM development, such as preference alignment [38], parameter-efficient fine-tuning [39], and foundational techniques of LLMs [40], they have largely concentrated on narrow subtopics. In contrast, this survey takes a holistic approach, providing a complete review of the core techniques commonly employed during post-training and systematically categorizing them. Additionally, we investigate the datasets and real-world applications integral to these methods, as illustrated in **Fig. 2**, and identify open challenges and promising directions for future research. The main contributions of this survey are as follows:

- **Comprehensive Historical Synthesis.** We provide the first in-depth synthesis of PoLMs, tracing their evolution from ChatGPT's initial reinforcement learning from human feedback (RLHF) to DeepSeek-R1's innovative cold-start RL approach. This synthesis covers key techniques (i.e., Fine-tuning, Alignment, Reasoning, Efficiency, and Integration and Adaptation), analyzing their development and associated challenges, such as computational complexity and ethical considerations. By presenting this progression as a cohesive narrative, enriched with essential references, we provide researchers with a comprehensive overview of the evolution of post-training in recent years, serving as a foundational resource for the field.

- **Structured Taxonomy and Framework.** We introduce a structured taxonomy, depicted in **Fig. 2**, classifying post-training methods into five distinct categories and organizing datasets into seven types while framing applications across professional, technical, and interactive domains. This framework clarifies the interrelationships and practical implications of these methods, offering a systematic perspective on their development. By providing well-defined categories and analytical insights, we
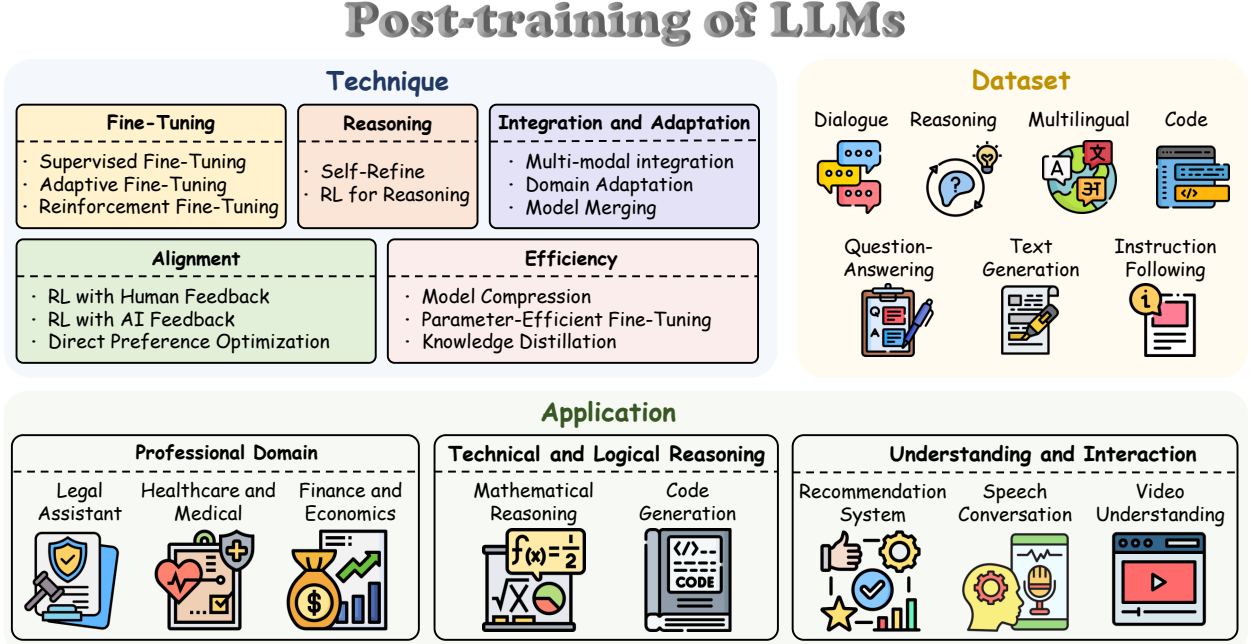
Figure 2: Structural overview of post-training techniques surveyed in this study, illustrating the organization of methodologies, datasets, and applications.

improve accessibility and comprehension for both novices and experts, establishing a comprehensive guide for navigating the complexities of post-training research.

- **Future Directions.** We highlight emerging trends, particularly the rise of Large Reasoning Models (LRMs) such as o1 [41] and DeepSeek-R1 [28], which harness large-scale reinforcement learning to push the boundaries of reasoning. We emphasize that ongoin advancements are crucial for further enhancing reasoning capabilities and domain adaptability. Our analysis identifies key challenges, including scalability constraints, ethical alignment risks, and multimodal integration obstacles. We propose research avenues such as adaptive RL frameworks and fairness-aware optimization. These directions aim to propel post-training forward, ensuring LLMs achieve heightened precision and trustworthiness to meet future demands.

1.2 Organization

This survey is systematically organized to comprehensively explore Post-training Language Models (PoLMs), spanning their historical evolution, methodologies, datasets, applications, and future trajectories. Section 2 provides a historical overview of PoLMs. Section 3 examines Fine-tuning, including Supervised Fine-Tuning (SFT) in Section 3.1 and Reinforcement Fine-Tuning (RFT) in Section 3.3. Section 4 addresses Alignment, covering Reinforcement Learning from Human Feedback (RLHF) in Section 4.1, Reinforcement Learning from AI Feedback (RLAIF) in Section 4.2, and Direct Preference Optimization (DPO) in Section 4.3. Section 5 focuses on Reasoning, with Self-Refinement Methods in Section 5.1 and Reinforcement Learning for Reasoning in Section 5.2. Section 6 surveys Efficiency-enhancing methods, including Model Compression in Section 6.1, Parameter-Efficient Fine-Tuning (PEFT) in Section 6.2, and Knowledge Distillation in Section 6.3. Section 7 investigates Integration and Adaptation, addressing multi-modal approaches, domain adaptation, and model merging. Section 8 reviews datasets used in post-training. Section 9 explores LLM applications. Section 10 evaluates open problems and future directions. Finally, Section 11 concludes with a summary and research outlook.

2 Overview

2.1 History of PoLMs

The advancement of LLMs constitutes a pivotal chapter in natural language processing (NLP), with post-training methods serving as critical catalysts in their evolution from generalized pre-trained architectures to specialized, task-adaptive systems. This section delineates the historical trajectory of Post-training Language Models (PoLMs), tracing their development from foundational pre-training milestones exemplified by BERT [2] and GPT [1] to the sophisticated post-training paradigms embodied in contemporary models such as o1 [41] and DeepSeek-R1 [28]. Illustrated in **Fig. 3**, this progression reflects a shift from establishing broad linguistic competence to enhancing task-specific adaptation, ethical alignment, reasoning sophistication, and multi-modal integration, marking a transformative journey in LLM capabilities.

The inception of modern PoLMs history aligns with the pre-training revolution in **2018**, heralded by the releases of BERT [2] and GPT [1], which redefined NLP benchmarks. BERT's bidirectional autoencoding framework, leveraging transformer architecture and self-attention, excelled in capturing contextual interdependencies for tasks like question answering, while GPT's autoregressive design prioritized generative coherence, setting a precedent for text generation. These models established the "pre-training and fine-tuning" paradigm, with subsequent refinements in **2019** via T5 [42], which unified diverse tasks under a text-to-text framework, fostering multi-task learning and laying a robust foundation for post-training advancements.

The landscape of PoLMs began to evolve substantially from **2020** onward, driven by a growing need to adapt pre-trained models efficiently to diverse tasks with limited data. Early innovations like prefix-tuning [43] and prompt-tuning [44] introduced lightweight adaptation strategies, enabling multi-task flexibility by modifying model inputs rather than retraining entire architectures, thus conserving computational resources while broadening applicability. This period also saw a pivotal shift toward user-centric optimization with the advent of Reinforcement Learning from Human Feedback (RLHF) in **2021** [45], a technique that leveraged human evaluations to align model outputs with subjective preferences, enhancing practical utility in conversational settings. By **2022**, RLHF matured with the adoption of Proximal Policy Optimization (PPO) [46], refining alignment stability and mitigating overfitting to noisy feedback. The release of ChatGPT in late 2022 [9] crystallized these advancements, showcasing RLHF's transformative potential in creating responsive, user-aligned LLMs and catalyzing a surge in PoLMs research. Concurrently, Chain-of-Thought (CoT) prompting [47] emerged as a reasoning enhancement strategy, encouraging models to articulate intermediate steps in complex tasks, thereby improving transparency and accuracy, particularly in logical inference and problem-solving domains.

Between **2022** and **2024**, PoLMs diversified to address domain specificity, ethical robustness, and multi-modal integration, reflecting an increasingly nuanced approach to LLM refinement. Domain adaptation techniques, such as Retrieval-Augmented Generation (RAG) [48], emerged to integrate external knowledge bases, enabling contextually enriched outputs for specialized fields without necessitating full retraining—a critical advancement for professional applications requiring up-to-date information. Ethical alignment efforts intensified, with Direct Preference Optimization (DPO) [49] in **2023** streamlining RLHF by directly optimizing model outputs against human preferences, bypassing intermediate reward modeling to enhance efficiency and robustness. Simultaneously, the pursuit of multi-modal capabilities gained traction, with models like PaLM-E [50] and Flamingo [51] pioneering vision-language integration, followed by BLIP-2 [52] and LLaVA [53], which extended these efforts into broader domains like medical imaging. Efficiency innovations paralleled these developments, notably through Mixture of Experts (MoE) architectures; Google's Switch-C Transformer [54] in 2022 introduced sparse activation of 1.6 trillion parameters across 2048 experts, while Mixtral [55] refined this paradigm, balancing scalability and performance. Reasoning enhancements during this period, such as self-play [56] and Monte Carlo Tree Search (MCTS) integration with CoT [57], further bolstered LLMs' decision-making capabilities by simulating iterative reasoning pathways, laying the groundwork for advanced inference-focused models.

A significant architectural advancement unfolded with the rise of Mixture of Experts (MoE) models, which diverge from traditional dense architectures by dynamically activating selective parameter subsets, thereby optimizing computational efficiency while accommodating expansive parameter scales. This paradigm was
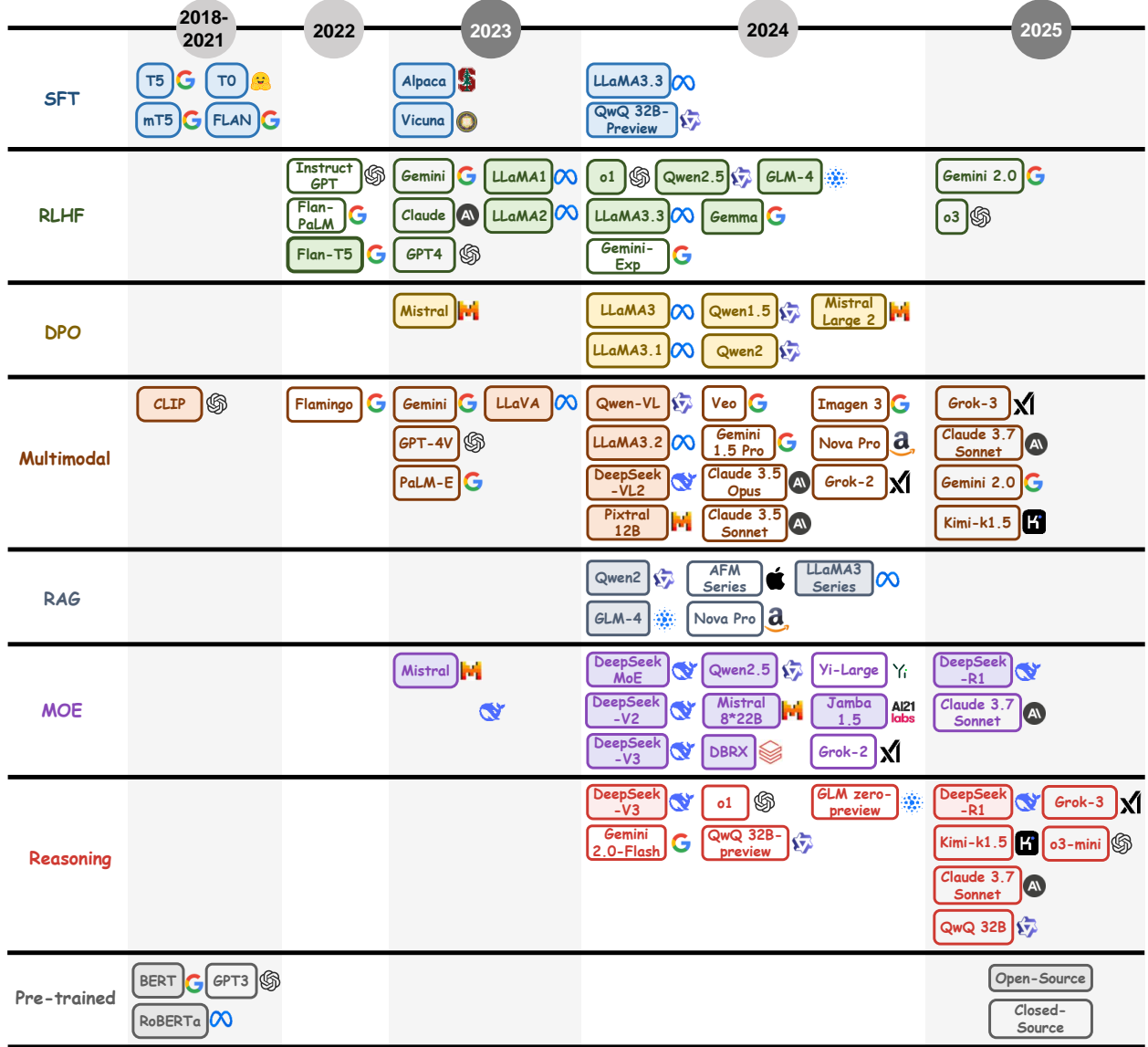
Figure 3: Timeline of post-training technique development for Large Language Models (2018–2025), delineating key milestones in their historical progression.

pioneered by Google's Switch-C Transformer [54] in **2022**, featuring 1.6 trillion parameters distributed across 2048 experts, a groundbreaking approach that balanced resource demands with performance gains. Subsequent iterations, such as Mixtral [55] and DeepSeek V2.5 [58]—the latter leveraging 236 billion total parameters with 21 billion active across 160 experts—further refined this framework, achieving state-of-the-art results on LMSYS benchmarks and demonstrating that sparse MoE architectures can rival dense models in both scalability and efficacy. These developments underscored a shift toward efficiency-focused PoLMs, enabling LLMs to handle complex tasks with reduced computational overhead, a critical step in broadening their practical applicability. By **2025**, DeepSeek-R1 [28] emerged as a landmark in PoLMs innovation, departing from conventional Supervised Fine-Tuning (SFT) reliance to embrace Chain-of-Thought (CoT) reasoning and exploratory RL strategies. Exemplified by DeepSeek-R1-Zero, which integrates self-verification, reflection, and extended CoT generation, this model validates RL-driven reasoning incentives within an open research paradigm, introducing distillation techniques [28] to transfer sophisticated reasoning patterns from larger to smaller architectures. This approach not only yields superior performance compared to standalone RL

training but also heralds a scalable, reasoning-centric paradigm for LLMs, poised to address the persistent challenges of computational efficiency and task adaptability in post-training methodologies.

2.2 Formula Foundations of PoLMs

2.2.1 Principle of Policy Optimization

The Proximal Policy Optimization (PPO) algorithm [46] is a key reinforcement learning technique, particularly useful in settings such as Reinforcement Learning with Human Feedback (RLHF) [45], where maintaining stability and efficiency is paramount. PPO achieves these objectives by constraining the size of policy updates, ensuring that changes to the model's behavior are gradual and controlled, thus preventing catastrophic shifts in performance. This is especially important when fine-tuning large-scale language models, where drastic policy updates could lead to undesirable or unpredictable behavior.

Definition. In the context of PPO, the state $s_t \in \mathcal{S}$ represents the environment at time t, which includes all relevant information the model needs to make a decision. The action $a_t \in \mathcal{A}(s_t)$ denotes the choice the model makes given the state s_t. This action is part of a sequence of decisions made by the model. Upon executing an action, the agent receives a reward $r_t \in \mathbb{R}$, which serves as feedback from the environment, signaling the success or failure of the action taken. The advantage function $A^\pi(s, a)$ measures how advantageous it is to take action a in state s under the current policy π, compared to the expected value of actions in that state. It is formally defined as the difference between the action-value function $Q^\pi(s, a)$ and the state-value function $V^\pi(s)$, which are defined as:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s), \tag{1}$$

where $Q^\pi(s, a)$ represents the expected cumulative reward obtained by taking action a in state s and following policy π, and $V^\pi(s)$ is the expected cumulative reward starting from state s and following policy π. Both functions account for future rewards, discounted by a factor γ.

Policy Update. The PPO algorithm optimizes the policy π_θ by making incremental updates based on the advantage function. The policy update is performed using the clipped objective function:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}\left(r_t(\theta), 1 - \epsilon, 1 + \epsilon \right) \hat{A}_t \right) \right], \tag{2}$$

where $r_t(\theta)$ represents the ratio of the probability of taking action a_t under the current policy π_θ to that under the old policy $\pi_{\theta_{old}}$. This ratio is defined as:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}.$$

The term $\hat{A}_t$ is the estimated advantage at timestep t, and the clipping function $\text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)$ restricts the policy update to a safe range, controlled by the hyperparameter ϵ. This clipping mechanism ensures that updates do not diverge too much from the previous policy, thus maintaining stability during training.

Value Function Update. The value function V_ϕ estimates the expected cumulative reward from a given state s_t under the policy π_θ. To ensure that the value function provides accurate estimates, it is optimized by minimizing the mean squared error between the predicted value and the actual reward:

$$\phi_{k+1} = \arg \min_\phi \mathbb{E}_{s_t \sim \pi_{\theta_k}} \left[(V_\phi(s_t) - R(s_t))^2 \right], \tag{3}$$

where $R(s_t)$ is the actual cumulative reward obtained from state s_t, and $V_\phi(s_t)$ is the estimated value under the current policy. The goal is to adjust the parameters ϕ to minimize the discrepancy between predicted and actual rewards, improving the accuracy of the value function.

2.2.2 Principle of RLHF

Reinforcement Learning with Human Feedback (RLHF) is a crucial method for aligning models with human preferences by utilizing human-generated feedback in the learning process. This approach incorporates a

reward function that explicitly captures human input, enabling the model to better adapt to user preferences and real-world applications.

Definition. In RLHF, a language model ρ generates a probability distribution over sequences of tokens in the vocabulary Σ. The model ρ produces a sequence of tokens $x_0, x_1, \ldots, x_{n-1}$ from the input space $X = \Sigma^{\leq m}$, where each token is conditionally dependent on previous tokens. The model's output is defined by the following conditional probability distribution:

$$\rho(x_0 \cdots x_{n-1}) = \prod_{0 \leq k < n} \rho(x_k \mid x_0 \cdots x_{k-1}). \tag{4}$$

The model ρ is trained on a task defined by an input space X, a data distribution D over X, and an output space $Y = \Sigma^{\leq n}$. For example, in text summarization, as shown in [16], a GPT-2 model [14] is trained using RLHF, where the task involves predicting text summaries based on a dataset such as CNN/DailyMail [59] and TL;DR [60].

Objective Function. The policy π is a language model that shares the same structure as the original model ρ. Initially, the policy π is set equal to ρ. The objective is to maximize the expected reward $R(x, y)$ for input-output pairs (x, y) by optimizing the policy. The reward function $R(x, y) : X \times Y \to \mathbb{R}$ assigns a scalar value to each input-output pair, and the optimal policy π^* is obtained by solving the following maximization problem:

$$\pi^* = \max_\pi \mathbb{E}[R] = \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi(\cdot|x)}[R(x, y)]. \tag{5}$$

This objective function represents a standard RL problem, where the model learns to maximize the expected reward through interaction with the environment, guided by human feedback.

2.2.3 Principle of DPO

Direct Preference Optimization (DPO) builds upon RLHF by directly optimizing the model's outputs based on human preferences, which are often expressed in the form of pairwise comparisons. DPO eliminates the need for traditional reward functions, focusing instead on optimizing model behavior by maximizing preference-based rewards.

Objective Function. We begin with the same RL objective as in previous methods [61, 62, 63], under a general reward function r. The optimal solution to the KL-constrained reward maximization objective is given by:

$$\pi_r(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right), \tag{6}$$

where $Z(x)$ is the partition function that ensures the output is normalized across all possible actions. Even when utilizing a maximum likelihood estimate r_ϕ of the true reward r^*, the partition function $Z(x)$ can be approximated, simplifying the optimization process. This formulation allows for more efficient preference optimization by directly adjusting the policy based on human feedback.

Preference Model. Using the Bradley-Terry model, which models preferences between two outputs y_1 and y_2, the optimal policy π^* satisfies the following preference model:

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)}, \tag{7}$$

where $p^*(y_1 \succ y_2 \mid x)$ represents the probability that the human prefers output y_1 over y_2 given the input x. This approach effectively incorporates human preferences into the model's optimization process.

2.2.4 Principle of GRPO

The Group Relative Policy Optimization (GRPO) algorithm is a variant of the Proximal Policy Optimization (PPO) algorithm in reinforcement learning, first introduced in DeepSeek's previous work, *DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models* [64]. GRPO omits the critic model,

instead estimating the baseline using group scores, which significantly reduces training resource consumption compared to PPO.

Definition. The most significant difference between the GRPO and PPO algorithms lies in the method used to calculate the advantage function. As we can see from Equation 1 in Section 2.2.1, the value of the advantage function $A^\pi(s, a)$ in PPO is derived from the difference between the Q-value and the V-value.

Objective Function. Specifically, for each question q, GRPO samples a group of outputs $\{o_1, o_2, \ldots, o_G\}$ from the old policy $\pi_{\theta_{old}}$ and then optimizes the policy model by maximizing the following objective:

$$
\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(O|q)]
$$

$$
\frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left\{ \min \left[\frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})} \hat{A}_{i,t}, \text{clip} \left(\frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})}, 1 - \epsilon, 1 + \epsilon \right) \hat{A}_{i,t} \right] \right. \tag{8}
$$

$$
\left. - \beta D_{KL}[\pi_\theta \parallel \pi_{\text{ref}}] \right\},
$$

where ϵ and β are hyper-parameters, and $\hat{A}_{i,t}$ is the advantage calculated based on relative rewards of the outputs inside each group only, which will be detailed in the subsection 5.2.

3 PoLMs for Fine-Tuning

Fine-tuning constitutes a cornerstone of adapting pre-trained Large Language Models (LLMs) to specialized tasks, refining their capabilities through targeted parameter adjustments. This process leverages labeled or task-specific datasets to optimize performance, bridging the gap between general-purpose pre-training and domain-specific requirements. This chapter explores three principal fine-tuning paradigms: **Supervised Fine-Tuning** (§3.1), which employs annotated datasets to enhance task-specific accuracy; **Adaptive Fine-Tuning** (§3.2), which customizes model behavior via instruction tuning and prompt-based methods; and **Reinforcement Fine-Tuning** (§3.3), which integrates reinforcement learning to iteratively refine outputs based on reward signals, fostering continuous improvement through dynamic interaction.

3.1 Supervised Fine-Tuning

Supervised Fine-Tuning (SFT) [45] adapts pre-trained LLMs to specific tasks by leveraging task-specific labeled datasets. Distinct from instruction tuning, which relies on directive prompts, SFT directly adjusts model parameters using annotated data, yielding models that are both precise and contextually attuned while preserving broad generalization capabilities. SFT bridges the divide between the expansive linguistic knowledge encoded during pre-training and the nuanced demands of targeted applications [36]. Pre-trained LLMs, through exposure to vast corpora, acquire generalized language patterns, reducing reliance on extensive domain-specific data for fine-tuning. Model selection is pivotal: smaller models like T5 [42] excel in resource-constrained settings with limited datasets, whereas larger models, such as GPT-4 [9], leverage their superior capacity to excel in complex, data-rich tasks.

3.1.1 Dataset Preparation for SFT

Crafting a high-quality SFT dataset is a multi-faceted process critical to fine-tuning success.

SFT Dataset Construction. An SFT dataset is typically structured as $\mathcal{D} = \{(I_k, X_k)\}_{k=1}^N$, where I_k is an instruction and X_k is its corresponding instance. This pairing enables the LLM to discern task-specific patterns and generate relevant outputs. Methods like Self-Instruct [86] enrich diversity by synthesizing novel instruction-output pairs, with duplicates filtered using metrics such as ROUGE-L [87] to maintain variety.

SFT Dataset Screening. Screening ensures that only high-quality instruction–instance pairs remain in the final dataset. A screening function $r(\cdot)$ is used to evaluate the quality of each pair (I_k, X_k), yielding a curated subset $\mathcal{D}'$:

$$
\mathcal{D}' = \left\{ (I_k, X_k) \in \mathcal{D} \mid r(I_k, X_k) \geq \tau \right\}, \tag{9}
$$

Table 1: Summary of Pre-trained Large Language Model Releases by Various Organizations (2018–2025). This table details key models from Meta, DeepSeek, OpenAI, and other entities, including their parameter sizes, training data scales (where reported), open-source status, and release timelines. Open-source status is denoted by ✅ for models publicly accessible to the research community and ⊗ for closed-source proprietary models.

Company	Model	Size (B)	Data Scale	Open Resource	Release Time
Meta	LLaMA [65]	7B,13B,30B,65B	1.4T tokens	✅	Feb-2023
	LLaMA2 [66]	7B,13B,70B	2T tokens	✅	Jul-2023
	LLaMA3 [25]	8B,70B,405B	15T tokens	✅	Apr-2024
DeepSeek	DeepSeek-V1 [67]	7B,67B	2T tokens	✅	Dec-2023
	DeepSeek-V2 [58]	236B	8.1T tokens	✅	May-2024
	DeepSeek-V3 [68]	671B	14.8T tokens	✅	Dec-2024
	DeepSeek-R1 [28]	671B	-	✅	Jan-2025
Qwen	Qwen1.5 [69]	0.5B,1.8B,4B,7B,14B,72B	3T tokens	✅	Sep-2023
	Qwen2 [70]	0.5B,1.5B,7B,57B,72B	7T tokens	✅	Jul-2024
	Qwen2.5 [71]	0.5B,1.5B,3B,7B,14B,72B	18T tokens	✅	Dec-2024
Mistral	Mistral-7B [72]	7B	-	✅	Sep-2023
	Mistral-8X7B [55]	4.7B	-	✅	Dec-2023
	Mistral-8X22B [55]	14B	-	✅	Apr-2024
	Mistral-Large2 [73]	12.3B	-	✅	Jul-2024
Anthropic	Claude2 [74]	2B	-	⊗	Jul-2023
	Claude3 [27]	-	-	⊗	Mar-2024
	Claude3.5 [75]	175B	-	⊗	Oct-2024
Google	Gemini1.0 [76]	-	-	⊗	Dec-2023
	Gemini1.5 [77]	-	-	⊗	Mar-2024
	Gemini2.0 [26]	-	-	⊗	Dec-2024
OpenAI	GPT-1 [1]	0.01B	-	⊗	Jun-2018
	GPT-2 [14]	0.15B	-	⊗	Feb-2019
	GPT-3 [7]	17.5B	-	⊗	Jul-2020
	InstructGPT [45]	-	-	⊗	Mar-2022
	GPT-4 [9]	180B	-	⊗	Mar-2023
	GPT-4o [78]	200B	-	⊗	Oct-2024
	o1 [41]	300B	-	⊗	Dec-2024
	o3-mini [79]	-	-	⊗	Jan-2025
ZhiPuAI	GLM-4[80]	130B	-	✅	Jun-2024
Databricks	DBRX [81]	132B	-	✅	Mar-2024
01.AI	Yi-Large [82]	-	-	✅	Mar-2024
AI21 Labs	Jamba1.5 Large [83]	94B	-	✅	Aug-2024
Amazon	Nova Pro [84]	-	-	⊗	Dec-2024
MoonshotAI	Kimi-k1.5 [85]	-	-	⊗	Jan-2025

where τ is a user-defined quality threshold. For example, the *Instruction Following Difficulty* (IFD) metric [88] quantifies how effectively a given instruction guides the model toward generating the expected response. The IFD function is expressed as:

$$r_\theta(Q, A) = \frac{\sum_{i=1}^N \log P\left(w_i^A \mid Q, \, w_1^A, \, \ldots, \, w_{i-1}^A; \, \theta\right)}{\sum_{i=1}^N \log P\left(w_i^A \mid w_1^A, \, \ldots, \, w_{i-1}^A; \, \theta\right)}, \tag{10}$$

where Q denotes the instruction, A is the expected response, and θ represents the model's learnable parameters. This metric compares the likelihood of generating a response both with and without the instruction, thereby providing a normalized measure of how effectively the instruction facilitates response generation. In-

struction–instance pairs that do not meet the selected IFD threshold are excluded, resulting in a refined dataset $\mathcal{D}'$.

SFT Dataset Evaluation. Evaluating an SFT dataset involves selecting a high-quality subset, $\mathcal{D}_{\text{eval}}$, to serve as a benchmark for model performance. This subset can be sampled from the curated dataset $\mathcal{D}'$ or derived from an independent portion to ensure impartiality. Traditional methods for SFT evaluation, such as Few-Shot GPT [7] and Fine-tuning strategies [89], are resource-intensive, whereas Instruction Mining [90] offers a more efficient alternative. Instruction Mining uses linear quality rules and a set of metrics to measure dataset quality, such as response length and average reward model scores [65], to evaluate correlations between these metrics and overall dataset quality.
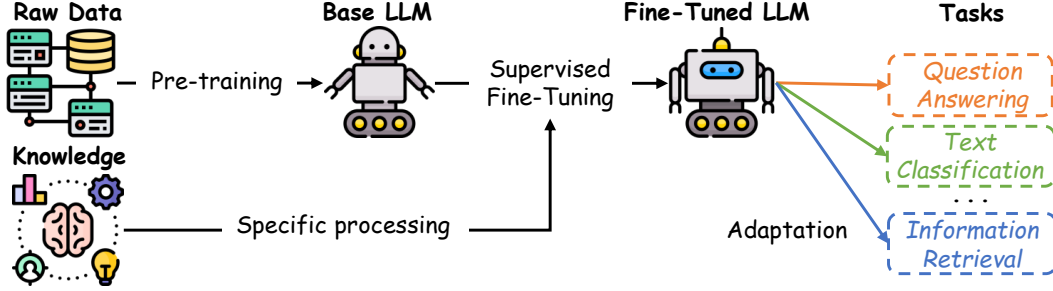

Figure 4: Process of Supervised Fine-Tuning.

3.1.2 Process of SFT

As depicted in **Fig. 4**, once the dataset is prepared, the fine-tuning process begins with a pre-trained LLM, typically obtained via unsupervised or self-supervised pre-training on large-scale raw datasets. The objective of this pre-training phase is to acquire general feature representations applicable across various tasks [36]. Subsequently, during the fine-tuning phase, the model's parameters are adjusted using task-specific annotated data, which aligns the model with the requirements of a given application. The objective function commonly used in this phase is the cross-entropy loss. For a classification task with N samples and C categories, it can be expressed as:

$$L_{\text{fine-tune}}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log P\big(y_j \mid x_i; \theta\big),\qquad(11)$$

where y_{ij} is the true label of sample i in category j, and $P\big(y_j \mid x_i; \theta\big)$ represents the model's predicted probability of sample i belonging to category j. Minimizing this loss function drives the model toward better alignment with the ground-truth labels, improving performance on the target task.

A prominent example is the BERT model [2], which undergoes extensive pre-training on broad linguistic corpora such as BooksCorpus and Wikipedia. During the fine-tuning phase, these broad representations are refined using task-specific data (e.g., the IMDB dataset [91] for sentiment analysis), allowing BERT to specialize in tasks such as sentiment classification and question answering.

3.1.3 Full-Parameter Fine-Tuning

Full-parameter fine-tuning refers to the process of adjusting all parameters of a pre-trained model, in contrast to parameter-efficient methods such as LoRA [92] or Prefix-tuning [43], which modify only a subset of parameters. Full-parameter tuning is often preferred for tasks requiring high precision, such as those in the medical and legal domains [93], but it entails substantial computational overhead. For example, fine-tuning a 65-billion-parameter model may require over 100 GB of GPU memory, creating challenges for resource-constrained environments. To mitigate such constraints, memory optimization techniques like LOMO [93] have been introduced, which reduce the memory footprint of gradient calculations and optimizer states. The model's parameters are updated according to the rule:

$$\theta_{t+1} = \theta_t - \eta\nabla_\theta L(\theta_t),\qquad(12)$$

14

where θ_t represents the model parameters at iteration t, η is the learning rate, and $\nabla_\theta L(\theta_t)$ denotes the gradient of the loss function. Memory optimization techniques, including Mixed Precision Training [94] and Activation Checkpointing [95], help reduce memory demands, enabling large models to be fine-tuned on systems with limited hardware resources.

GPT-3 to InstructGPT. A notable example of full-parameter fine-tuning is the transition from GPT-3 to InstructGPT [45], where the model's entire parameter set was fine-tuned using a dataset designed for instruction-following tasks. This approach leads to optimal performance but is computationally expensive due to the need to update all parameters.

3.2 Adaptive Fine-Tuning

Adaptive fine-tuning modifies a pre-trained model's behavior to better address user-specific needs and handle a broader range of tasks. This approach introduces additional cues to guide the model's output generation, offering a flexible framework for customizing the model's responses. Notable methods in adaptive fine-tuning include instruction tuning and prompt-based tuning, both of which significantly enhance the adaptability of LLMs by introducing task-specific guidance.
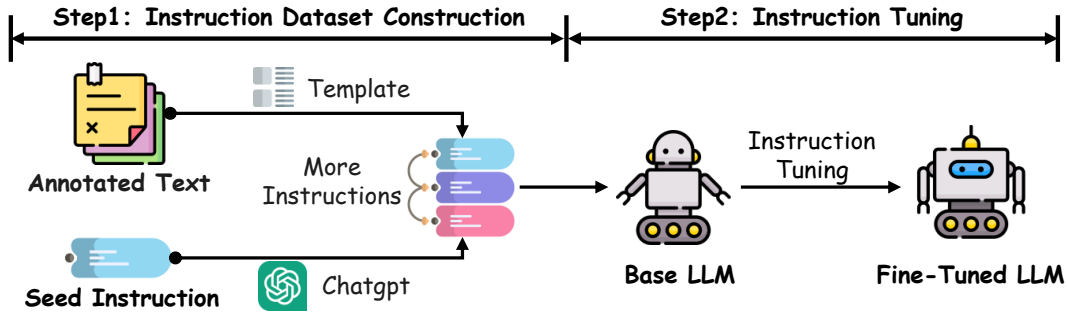

Figure 5: Workflow of Instruction Fine-tuning, illustrating the general pipeline for Instruction Dataset Construction and Instrction Tuning in Large Language Models.

3.2.1 Instruction Tuning

Instruction Tuning [96] is a technique that refines a base LLM by fine-tuning it on specially constructed instruction datasets. This method substantially boosts the model's ability to generalize across a variety of tasks and domains, improving its flexibility and accuracy. As shown in **Fig. 5**, the process begins by transforming existing NLP datasets (e.g., those for text classification, translation, and summarization) into natural language instructions that include task descriptions, input examples, expected outputs, and illustrative demonstrations. Techniques like Self-Instruct [86] further enhance the diversity of these datasets by automatically generating additional instruction–output pairs, expanding the model's exposure to a broader range of tasks. The fine-tuning procedure adapts the model's parameters to align with these task-specific instructions, resulting in an LLM that performs robustly across both familiar and previously unseen tasks. For instance, InstructGPT [45] and GPT-4 [7] have shown significant improvements in instruction-following capabilities across a wide array of applications.

The effectiveness of Instruction Tuning largely depends on the quality and breadth of the instruction datasets. High-quality datasets should encompass a wide range of languages, domains, and task complexities to ensure that the model remains broadly applicable [96]. Furthermore, the clarity and organization of instructions play a critical role in enabling the model to interpret and execute tasks effectively. Techniques such as integrating demonstration examples, including Chain-of-Thought prompting [47], can significantly improve performance on tasks requiring complex reasoning. Moreover, ensuring a balanced distribution of tasks during the fine-tuning phase is essential to avoid overfitting or diminishing model performance due to imbalanced task coverage. Techniques such as proportional task sampling or weighted loss functions are useful in addressing these issues, ensuring that each task contributes equitably to the fine-tuning procedure. Therefore, by con-

structing and managing instruction datasets meticulously, researchers can greatly enhance the generalization capabilities of fine-tuned LLMs, enabling them to excel across a wide range of tasks and domains [97].

3.2.2 Prefix-Tuning

Prefix-tuning [98] is a parameter-efficient fine-tuning method that involves adding a sequence of trainable prefix tokens (continuous vectors) to each Transformer layer in the language model, while keeping the core model parameters fixed. As depicted in **Fig. 6 (a)**, these prefix vectors are task-specific and function as virtual token embeddings. To optimize the prefix vectors, a reparameterization trick is used, wherein a small multi-layer perceptron (MLP) function is learned to map a smaller matrix to the prefix parameters instead of directly optimizing the prefix vectors. This method has been shown to stabilize the training process. Once the prefix vectors are optimized, the mapping function is discarded, and only the derived prefix vectors are retained for enhancing task-specific performance.
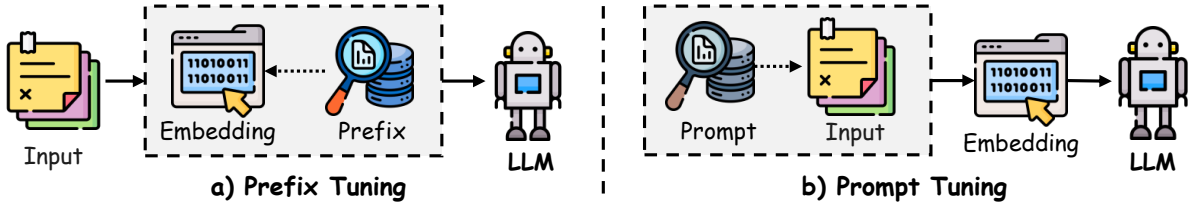

Figure 6: Comparison of Prefix Tuning and Prompt Tuning, delineating their distinct approaches to parameter fine-tuning: a) Prefix Tuning and b) Prompt Tuning.

By prepending a learned continuous prompt to the input sequence and utilizing layer-wise prompts, the model's behavior is steered toward task-specific outputs without requiring full-model fine-tuning. Since only the prefix parameters are adjusted, this results in a more parameter-efficient approach. Building upon this, P-Tuning v2 [99] incorporates layer-wise prompt vectors into the Transformer architecture specifically for natural language understanding tasks. This approach also leverages multi-task learning to optimize shared prompts across tasks, enhancing model performance across different parameter scales [43]. The potential of prefix-tuning to facilitate rapid and efficient adaptation of large language models for specific tasks is evident, making it a compelling strategy for applications requiring flexibility and efficiency.

3.2.3 Prompt-Tuning

Prompt-tuning [44, 100] is a method designed to adapt large language models efficiently by optimizing trainable vectors at the input layer rather than modifying the model's internal parameters. As shown in **Fig. 6 (b)**, this technique builds on discrete prompting methods [101, 102] by introducing soft prompt tokens, which can be structured either in an unrestricted format [44] or as a prefix [100]. These learned prompt embeddings are combined with the input text embeddings before being processed by the model, thereby guiding the model's output while keeping the pre-trained weights frozen. Two notable implementations of prompt-tuning are P-tuning [44], which uses a flexible method to combine context, prompt, and target tokens, making it suitable for both understanding and generation tasks. This method enhances the learning of soft prompt representations through a bidirectional LSTM architecture. In contrast, standard prompt-tuning [100] employs a simpler design, wherein prefix prompts are prepended to the input, and only the prompt embeddings are updated during training based on task-specific supervision.

Research has shown that prompt-tuning can match the performance of full-parameter fine-tuning across many tasks, while requiring significantly fewer trainable parameters. However, its success is closely tied to the underlying language model's capacity, as prompt-tuning only modifies a small number of parameters at the input layer [44]. Building on these advancements, newer approaches such as P-Tuning v2 [99] have demonstrated that prompt-tuning strategies can scale effectively across various model sizes, handling complex tasks previously thought to require full fine-tuning. These findings establish prompt-tuning as a highly efficient alternative to traditional fine-tuning, offering comparable performance with reduced computational and memory costs.

3.3 Reinforcement Fine-Tuning

Reinforcement Fine-Tuning (ReFT) [103] represents an advanced technique that integrates RL with SFT to enhance the model's ability to solve complex, dynamic problems. Unlike traditional SFT, which typically uses a single CoT annotation for each problem, ReFT enables the model to explore multiple valid reasoning paths, thereby improving its generalization capacity and problem-solving skills. The ReFT process begins with the standard SFT phase, where the model is initially trained on labeled data to learn fundamental task-solving abilities through supervised annotations. Following this initial fine-tuning, the model undergoes further refinement using RL algorithms, such as Proximal Policy Optimization (PPO) [46]. During the reinforcement phase, the model generates multiple CoT annotations for each problem, exploring different potential reasoning paths. These generated paths are evaluated by comparing the model's predicted answers to the true answers, with rewards assigned for correct outputs and penalties for incorrect ones. This iterative process drives the model to adjust its policy, ultimately improving its reasoning strategy.
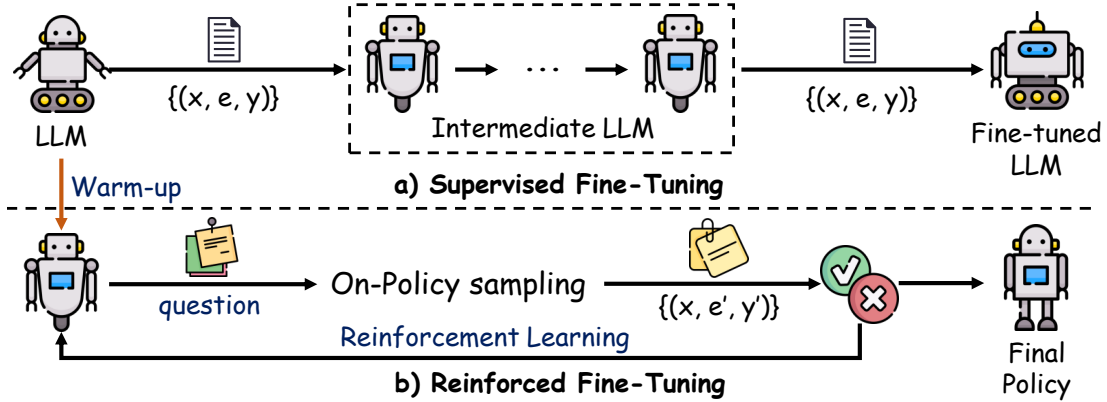

Figure 7: Process of Reinforcement Fine-Tuning (ReFT), depicting the iterative Supervised Fine-Tuning (SFT) warm-up followed by RL training on identical datasets.

As shown in **Fig. 7**, the ReFT process is executed in two stages. The upper section represents the SFT phase, where the model iterates over the training data to learn the correct CoT annotation for each problem over several epochs. In the lower section, the ReFT phase is introduced: starting from the SFT-trained model, the model generates alternative CoT annotations (e') based on its current policy and compares its predicted answers (y') with the true answers (y). Positive rewards are given for correct answers, and negative rewards for incorrect answers, driving the model to improve its performance. These reward signals are then used to update the model's policy through reinforcement learning, enhancing its ability to generate accurate and diverse CoT annotations.

Recent studies have demonstrated that ReFT significantly outperforms traditional SFT approaches [103]. Moreover, the integration of inference-time strategies, such as majority voting and re-ranking, can further enhance performance, allowing the model to refine its outputs after training. Notably, ReFT achieves these improvements without requiring additional or augmented training data, learning solely from the existing dataset used during the SFT phase. This highlights the model's superior generalization ability, as it learns more efficiently and effectively from the available data.

4 PoLMs for Alignment

Alignment in LLMs involves guiding model outputs to conform to human expectations and preferences, particularly in safety-critical or user-facing applications. This chapter discusses three major paradigms for achieving alignment: **Reinforcement Learning with Human Feedback** (§4.1), which employs human-labeled data as a reward signal; **Reinforcement Learning with AI Feedback** (§4.2), which leverages AI-generated feedback to address scalability issues; and **Direct Preference Optimization** (§4.3), which learns directly from pairwise human preference data without requiring an explicit reward model. Each paradigm

offers distinct advantages, challenges, and trade-offs in its pursuit of robust alignment. A concise comparison of these and related methods is summarized in **Tab. 2**.

Table 2: Comparative Overview of Alignment Methods for Large Language Models (2022–2024). This table evaluates prominent alignment techniques across eight metrics: **RM1** (Explicit or Implicit Reward Model), **RM2** (Point Reward or Preference Probability Model), **RM3** (Response- or Token-level Reward), **RM4** (Positive or Negative Reward Model), **F** (Feedback Type: Human or AI), **RL1** (Reference Model or Reference Model-Free RL), **RL2** (On-policy or Off-policy RL), and **O** (Online/Iterative or Offline/Non-iterative Optimization).

Methods	RM1	RM2	RM3	RM4	F	RL1	RL2	O	Release Time
InstructGPT [45]	Explicit	Point	Response	Positive	Human	Reference	On	Offline	Mar-2022
RLHF: Anthropic [104]	Explicit	Point	Response	Positive	Human	Reference	Off	Hybrid	Apr-2022
RLAIF-Anthropic [105]	Explicit	Point	Response	Positive	AI	Reference	Off	Offline	Dec-2022
RRHF [106]	–	–	–	–	Human	Free	Off	Offline	Apr-2023
DPO [49]	Implicit	Point	Response	Positive	Human	Reference	Off	Offline	May-2023
PRO [107]	Explicit	Point	Response	Positive	Human	Free	Off	Offline	Jun-2023
RLAIF-Google [108]	Explicit	Point	Response	Positive	AI	Reference	Off	Offline	Sep-2023
CRINGE [109]	Implicit	Point	Response	Positive	AI	Reference	Off	Online	Dec-2023
CPO [110]	Implicit	Point	Response	Negative	Human	Reference	Off	Offline	Jan-2024
Iterative DPO [111]	Implicit	Point	Response	Positive	AI	Reference	Off	Online	Jan-2024
RLOO [112]	Explicit	Point	Response	Positive	Human	Free	Off	Offline	Feb-2024
LiPO [113]	Implicit	Point	Response	Positive	Human	Reference	Off	Offline	Feb-2024
GRPO [114]	Implicit	Point	Response	Positive	AI	Reference	Off	Online	Feb-2024
NN [115]	Implicit	Point	Response	Negative	Human	Reference	On	Offline	Mar-2024
R-DPO [116]	Implicit	Point	Response	Positive	Human	Reference	Off	Offline	Mar-2024
DNO [117]	–	Preference	Response	Positive	Human	Reference	Hybrid	Offline	Apr-2024
DPO: from r to Q [118]	Implicit	Point	Token	Positive	Human	Reference	Off	Offline	Apr-2024
TDPO [119]	Implicit	Point	Token	Positive	Human	Reference	Off	Offline	Apr-2024
NPO [120]	Implicit	Point	Response	Negative	Human	Reference	Off	Offline	Apr-2024
SIMPO [121]	–	–	–	–	Human	Free	Off	Offline	May-2024
UNA [122]	Implicit	Preference	Response	Positive and Negative	Human and AI	Free	Off	Offline	Sep-2024
Aligner [123]	Implicit	Preference	Response	Positive and Negative	AI	free	Off	Offline	Nov-2024

4.1 Reinforcement Learning with Human Feedback

Supervised Fine-Tuning (SFT) [45] has served as a foundational technique for guiding LLMs to follow human instructions. Nevertheless, the diversity and quality of annotated data in purely supervised scenarios can be uneven, and the ability of supervised models to capture more nuanced or adaptive human preferences is often limited. In response, reinforcement learning (RL)-based fine-tuning has been proposed to address these shortcomings. Among RL methods, Reinforcement Learning from Human Feedback (RLHF) [104] stands out as one of the earliest and most impactful RL-based post-training approaches for alignment.

As illustrated in **Fig. 8**, RLHF first aggregates human feedback in the form of preference labels or reward signals, then uses that information to train a reward model. Guided by this reward model, the policy is iteratively adjusted to better match human preferences. Compared with SFT, RLHF incorporates continuous, preference-driven updates, leading to stronger alignment outcomes. Notably, modern LLMs such as GPT-4 [9], Claude [27], and Gemini [76] have benefited from these mechanisms, showcasing improvements in instruction-following, factual consistency, and user relevance. Below, we discuss the major components of RLHF, including feedback mechanisms, reward modeling, and policy-learning strategies.

4.1.1 Feedback Mechanisms of RLHF

Human feedback lies at the core of RLHF, informing the reward model about user preferences and guiding policy updates. This subsection adopts the taxonomy of [124] to categorize common forms of human feedback. **Tab. 3** presents these feedback types across dimensions such as granularity, involvement level, and explicitness. Each feedback modality contributes to distinct aspects of model optimization, offering varying levels of interpretability, scalability, and noise tolerance.
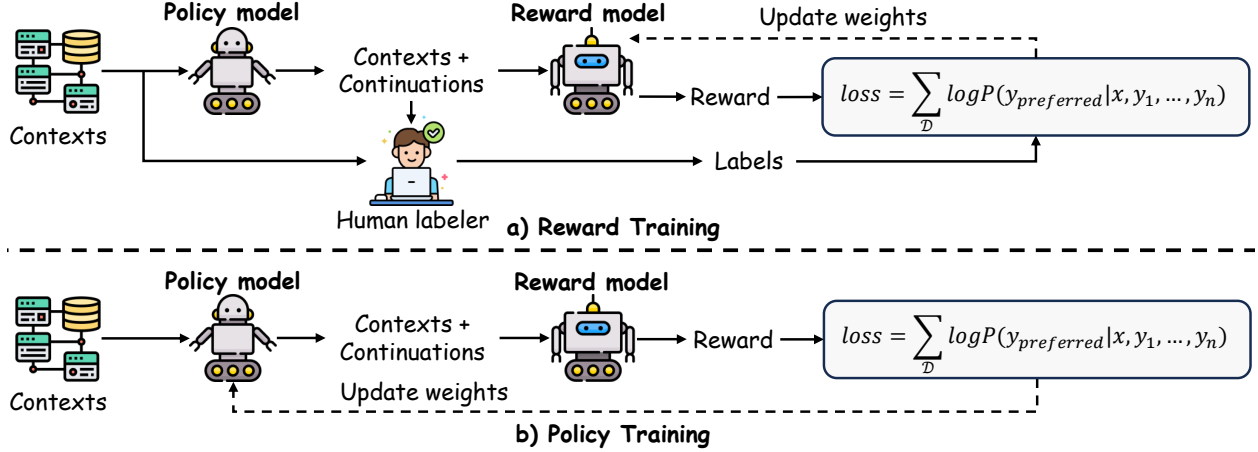
Figure 8: Workflow of Reinforcement Learning from Human Feedback (RLHF), delineating the overall training process for aligning Large Language Models with human preferences.

Table 3: Classification of Feedback Types in Post-training Methods for Large Language Models. This table provides an overview of common feedback classes and their defining attributes across six metrics: **Granularity** (scope: episode, segment, or step), **Involvement** (engagement: observed, active, or co-generative), **Arity** (instance count: single, multiple, or ternary), **Abstraction** (target: feature or instance), **Intent** (purpose: evaluative, descriptive, or literal), and **Explicitness** (directness: explicit or implicit).

Feedback	Method	Granularity	Involvement	Arity	Abstraction	Intent	Explicitness
	Critique [125]	-	Observed	Single	-	Evaluative	Explicit
	Comparisons [126]	-	Observed	Multiple	-	Evaluative	Explicit
	Inter-Temporal [127]	Segment	Observed	Single	-	Evaluative	Explicit
Primary	**Proxy Rewards** [128]	Episode	Observed	-	Feature	Descriptive	Explicit
	Social Behavior [129]	Segment	Observed	Single	Instance	Literal	Implicit
	Improvements [130]	Episode	Co-generative	Single	Instance	-	-
	Natural Language [131]	-	Observed	Single	-	Descriptive	Explicit
Representation	**E-Stops** [132]	Episode	Observed	Single	Instance	Literal	Implicit
	Importance [133]	-	Observed	-	-	Descriptive	Explicit
Supplementary	**Feature Traces** [134]	Segment	Active	Single	Instance	Descriptive	Explicit
	Similarity Queries [135]	-	Observed	Ternary	-	Descriptive	Explicit

Primary Feedback. This category comprises feedback types that most directly shape reward models in RLHF. For example, Critique [125] focuses on explicit human assessments of agent behavior, often refined via binary or multi-label annotations to mitigate noise. Comparisons [126] allow evaluators to compare multiple outputs or trajectories; while larger choice sets can offer richer signals, they may also lead to causal confusion. Inter-Temporal Feedback [127] refines trajectory assessment by providing judgments at different time steps, whereas Proxy Rewards [128] incorporate approximate reward functions that direct the model toward a user-defined goal. Social Behavior [129] harnesses implicit cues (e.g., facial expressions) to align agent objectives with user sentiment. Improvements [130] emphasize real-time human interventions for incremental policy refinement. Finally, Natural Language Feedback [131] leverages textual information to convey preferences and suggestions for improvement.

Supplementary Feedback. In addition to primary feedback, two classes further strengthen the reward modeling process. Emergency stops (e-stops) [132] allow humans to intervene in an agent's behavior by halting its trajectory without suggesting alternatives. This feedback is characterized by implicit involvement and a singular focus on preventing undesirable behavior. In contrast, importance labels [133] indicate the significance of specific observations for achieving objectives, providing explicit feedback that does not directly alter

behavior. This feedback varies by context and serves as supplementary input, reinforcing the overall learning process for the reward model.

Representation-Specific Feedback. Certain feedback types primarily enhance representation learning rather than directly shaping the reward function. Feature Traces [134] prompt human operators to demonstrate monotonic changes in a given feature, thus enabling dynamic expansion of feature sets. Similarity Queries [135] compare triplets of trajectories, guiding representation learning via pairwise distances in trajectory space. By leveraging these representation-specific feedback forms, RLHF can achieve more robust generalization to new tasks and contexts.

4.1.2 Reward Model of RLHF

The true reward function $r(x, y)$ is often unknown, making it necessary to construct a learnable reward model $r_\theta(x, y)$ based on human-provided preferences. This model predicts the degree to which a candidate output y aligns with human expectations for a given input x. To obtain the training data for $r_\theta(x, y)$, human evaluators compare or label output pairs according to their relative suitability, and the model is typically trained using a cross-entropy loss on these comparisons. To discourage the policy π from straying too far from the initial model ρ, a penalty term controlled by the hyperparameter β is introduced into the reward function:

$$r_\theta(x, y) = r(x, y) - \beta \log \frac{\pi(y \mid x)}{\rho(y \mid x)}, \tag{13}$$

where $\pi(y \mid x)$ is the probability that the fine-tuned policy π produces output y given input x, and $\rho(y \mid x)$ is the corresponding probability under the original model ρ. This term ensures that, while π adapts to human feedback, it remains constrained by prior knowledge captured in ρ.

Evaluating the reward function $r_\theta(x, y)$ is critical, as it directly influences learning effectiveness and policy performance. Accurately assessing this function helps identify suitable reward structures for aligning model outputs with human preferences. However, in safety-sensitive domains, standard rollout methods [136, 137] and off-policy evaluations [138, 139] may be infeasible because of risks related to online interactions, biases, and the need for ground-truth rewards. To address these challenges, two prominent approaches are commonly adopted:

Distance Functions. Recent research has focused on reward evaluation distance functions that account for potential transformations, such as potential shaping. For example, EPIC [140] measures reward function equivalences under various transformations, while DARD [141] refines canonicalization to ensure that evaluations remain grounded in feasible transitions. EPIC-like distances [142] generalize EPIC's methodology by allowing variability in canonicalization, normalization, and metric functions, and STARC [143] preserves EPIC's theoretical properties while offering additional flexibility.

Visual and Human Inspection. Other methods rely on interpretability and curated datasets to gauge the validity of learned reward functions. PRFI [144] uses a preprocessing step to simplify reward functions while retaining equivalence, thus enhancing their transparency. Meanwhile, CONVEXDA and REWARDFUSION [145] propose datasets designed to test how consistently reward models respond to semantic variations in prompts. Together, these techniques contribute to more reliable evaluations of reward functions, reinforcing the alignment of Large Language Models with human preferences.

4.1.3 Policy Learning of RLHF

Policy learning for RLHF, as shown in **Fig. 9**, involves optimizing policies through human feedback in both online and offline settings.

Online Learning. In online RLHF, systems gather real-time human preferences on newly generated model trajectories. Algorithms like DPS [146] use Bayesian updates to manage the dueling process, while PPS and PEPS [147] integrate dynamic programming and bandit ideas to refine policy behavior. In LPbRL [148], feature embeddings capture evolving reward structures, and PbOP [149] integrates least-squares estimates for both transition dynamics and preference signals. More recently, PARL [150] targets data-collection efficiency by treating feedback acquisition as an integral part of policy optimization.
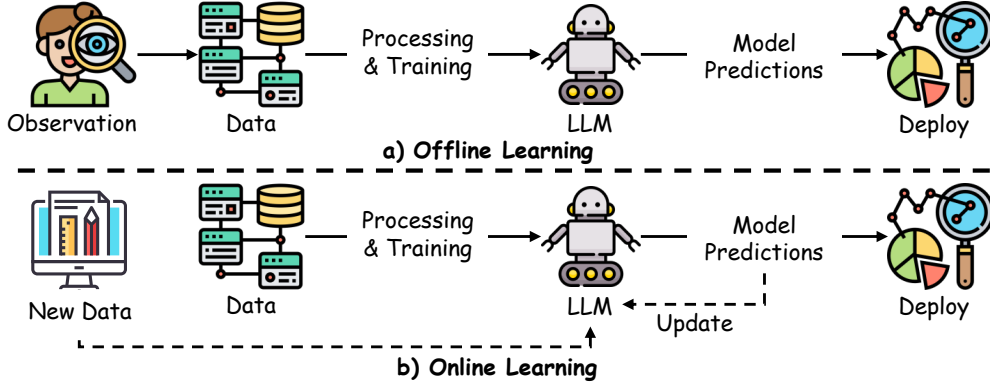
Figure 9: Comparison of Online and Offline RLHF, illustrating continuous feedback collection during policy execution in online RLHF versus pre-collected trajectory utilization in offline RLHF.

Offline Learning. In offline RLHF, previously gathered preference-labeled trajectories are used to learn or refine a policy. For instance, [151] study pessimistic maximum likelihood estimation for policy learning with pairwise comparison data, establishing bounds on performance. Extensions like FREEHAND [152] and DCPPO [153] generalize to unknown preference models, exploring the interplay between offline data coverage and policy generalization. Moreover, [154] address overfitting in Boltzmann models for pairwise comparisons, while DCPPO [153] further studies the dynamic discrete choice model for improved feedback efficiency.

Blending Online and Offline Learning. Hybrid methods combine offline pretraining with online preference aggregation, capitalizing on pre-collected data while still incorporating real-time updates. PFERL [155] adopts a two-phase approach to minimize human queries, whereas PERL [156] explores optimistic least-squares strategies for active exploration. Dueling RL [148] and its extensions (e.g., REGIME in PRPRL [152]) reduce human labeling requirements by carefully partitioning data acquisition from feedback collection, thus optimizing trade-offs among sample efficiency, annotation cost, and policy performance.

4.2 Reinforcement Learning with AI Feedback

Reinforcement Learning with AI Feedback (RLAIF) extends the RLHF paradigm by employing LLMs to generate feedback signals. This approach can complement or replace human feedback, providing more scalable, lower-cost preference data in tasks where human annotations are scarce, costly, or inconsistent.

4.2.1 RLAIF vs RLHF

A major challenge in applying RLHF at scale lies in its reliance on human-generated preference labels, which necessitate considerable resources for gathering, curating, and labeling data. The process of annotating data is both time-intensive and costly, and human evaluators may introduce inconsistencies, thereby complicating large-scale, consistent labeling across all model outputs. These constraints significantly limit the scalability and efficiency of RLHF. To address these challenges, RLAIF was proposed by [105], which combines human feedback with AI-generated feedback for training models via reinforcement learning. By leveraging LLMs as the source of feedback, RLAIF reduces reliance on human annotators, offering a viable alternative to traditional RLHF. This approach enables continuous feedback generation, significantly enhancing scalability while preserving the flexibility of human-guided model optimization.

As depicted in **Fig. 10**, the key distinction between RLHF and RLAIF lies in the source of feedback: RLHF relies on human-generated preferences, while RLAIF uses AI-generated feedback to guide policy updates. Empirical studies, such as those by [157], have demonstrated that RLAIF can achieve performance comparable to or even superior to RLHF, as evaluated by human raters. Notably, RLAIF not only surpasses traditional supervised fine-tuning baselines but does so with an LLM preference labeler of the same scale as the policy model, underscoring the approach's efficiency.
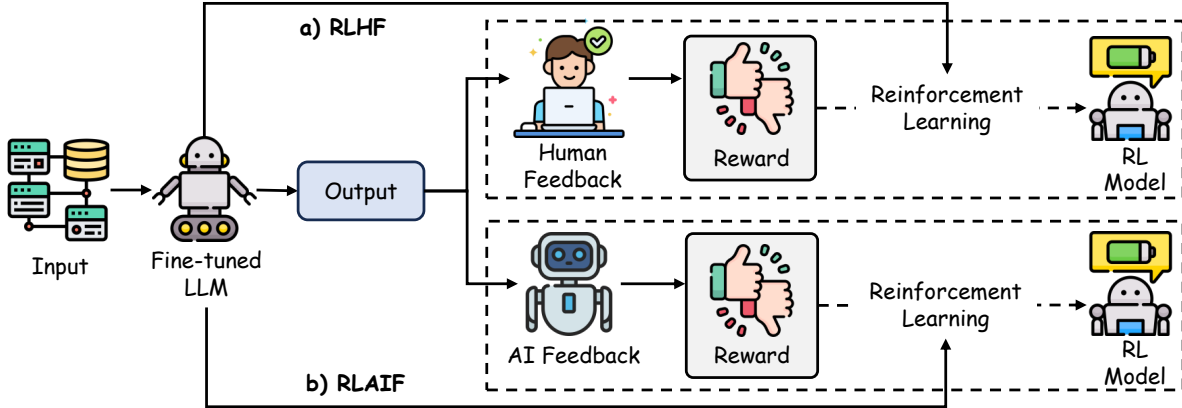
Figure 10: Comparison of RLHF and RLAIF Approaches, delineating their distinct methodologies for preference alignment in Large Language Models.

4.2.2 RLAIF Training Pipeline

The RLAIF training pipeline follows several key stages wherein AI-generated feedback is utilized to iteratively refine the model's behavior. The pipeline facilitates the alignment of LLM outputs with human expectations in a manner that scales across various tasks, as detailed by [108]. The stages are as follows:

AI Feedback Collection. In this phase, the AI system generates feedback based on predefined criteria, which may include task-specific metrics, correctness of responses, or appropriateness of the model's outputs. Unlike human feedback, which requires interpretation and manual annotation, AI feedback can be consistently generated across a broad range of model outputs. This characteristic enables AI feedback to be continuously provided, scaling the feedback loop significantly.

Reward Model Training. The AI-generated feedback is subsequently used to train or refine a reward model. This model maps input-output pairs to corresponding rewards, aligning the model's output with the desired outcomes as dictated by the feedback. While traditional RLHF relies on direct human feedback to evaluate outputs, RLAIF utilizes AI-generated labels, which, although potentially introducing issues related to consistency and bias, offer advantages in scalability and independence from human resources.

Policy Update. The final stage involves updating the model's policy based on the reward model trained in the previous step. Reinforcement learning algorithms are employed to adjust the model's parameters, optimizing the policy to maximize cumulative reward across a variety of tasks. This process is iterative, with the reward model guiding the model's outputs towards higher alignment with the intended objectives.

The principal advantage of RLAIF lies in its ability to scale the feedback loop without requiring continual human intervention. By substituting human feedback with AI-generated feedback, RLAIF facilitates the continuous improvement of LLMs across multiple tasks, alleviating the bottleneck posed by human labeling efforts.

4.3 Direct Preference Optimization

As previously discussed, RLHF [45] typically consists of three stages: Supervised Fine-Tuning [17, 86], Reward Modeling, and Reinforcement Learning (usually implemented via Proximal Policy Optimization, PPO) [46]. Despite its effectiveness, RLHF can be complex and prone to instability, particularly in the stages where a reward model is fitted and then used to fine-tune a large language model. The difficulty lies in creating a reward model that accurately reflects human preferences and in the challenge of fine-tuning the language model to optimize this estimated reward while staying close to the original model. To address these issues, Direct Preference Optimization (DPO) [49] has been introduced as a more stable and computationally efficient alternative. DPO simplifies the reward optimization process by directly linking the reward function to the optimal policy. It treats the reward maximization problem as a single-stage policy training problem

based on human preference data, thus avoiding the complexities of reward model fitting and the dependencies of the Bradley-Terry model [158].

4.3.1 Foundation of DPO

RLHF involves training a reward model (RM) and fine-tuning a language model (LM) via reinforcement learning. DPO simplifies this process by training the LM directly with human preference data, implicitly capturing the reward model within the policy itself.

KL-Regularized Reward Maximization Objective. DPO begins with the well-established KL-regularized reward maximization framework, as shown in the following objective:

$$\pi^* \;=\; \arg\max_{\pi} \; \mathbb{E}_{x \sim \mathcal{D}, \, y \sim \pi(\cdot|x)} \Big[r(x,y) \;-\; \beta \, \mathrm{KL}\Big(\pi(\cdot \mid x) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid x) \Big) \Big], \tag{14}$$

where $r(x,y)$ represents the reward function, $\beta > 0$ is a coefficient controlling the degree of proximity to the reference policy π_{ref}, and $\mathrm{KL}(\cdot\|\cdot)$ denotes the Kullback-Leibler divergence. Here, $x \sim \mathcal{D}$ represents the input drawn from the data distribution, and $y \sim \pi(\cdot \mid x)$ denotes the output sampled from the policy.

Deriving the Optimal Policy. Under appropriate assumptions, the solution to Eq. (14) is derived in the form of a Boltzmann distribution [61, 62, 63]:

$$\pi^*(y \mid x) \;=\; \frac{1}{Z(x)} \, \pi_{\mathrm{ref}}(y \mid x) \exp\Big(\tfrac{1}{\beta} \, r(x,y) \Big), \tag{15}$$

where the partition function

$$Z(x) \;=\; \sum_{y} \pi_{\mathrm{ref}}(y \mid x) \exp\Big(\tfrac{1}{\beta} \, r(x,y) \Big) \tag{16}$$

acts as a normalization term ensuring that π^* remains a valid probability distribution (i.e., that its probabilities sum to 1).

Reparameterizing the Reward. Taking the natural logarithm of both sides of Eq. (15), we can relate the reward $r(x,y)$ to the optimal policy π^*. This yields:

$$r^*(x,y) \;=\; \beta \Big[\log \pi^*(y \mid x) \;-\; \log \pi_{\mathrm{ref}}(y \mid x) \Big] \;+\; \beta \log Z(x), \tag{17}$$

where $\beta \log Z(x)$ is a constant that does not affect pairwise comparisons of rewards. If the optimal policy π^* is known, the true reward $r^*(x,y)$ can be determined up to this constant.

Bradley–Terry Preferences. Under the Bradley-Terry model [158], human preferences between two outputs y_1 and y_2 are governed by the difference in their reward values. The probability of preferring y_1 over y_2 is given by

$$p^*\big(y_1 \succ y_2 \mid x\big) \;=\; \frac{\exp\big(r^*(x,y_1)\big)}{\exp\big(r^*(x,y_1)\big) \;+\; \exp\big(r^*(x,y_2)\big)}. \tag{18}$$

Substituting Eq. (17) into Eq. (18), we obtain the final preference model:

$$p^*\big(y_1 \succ y_2 \mid x\big) \;=\; \frac{1}{1 \;+\; \exp\Big(\beta \Big[\log \frac{\pi^*(y_2|x)}{\pi_{\mathrm{ref}}(y_2|x)} \;-\; \log \frac{\pi^*(y_1|x)}{\pi_{\mathrm{ref}}(y_1|x)} \Big] \Big)}. \tag{19}$$

This expression links the pairwise human preference probability to the ratio of the optimal policy π^* and reference policy π_{ref}.

Objective of DPO. DPO sidesteps explicit reward modeling by learning a policy directly from preference data. Given a dataset of preference triplets $\{(x, y_w, y_l)\}$, where y_w is the preferred output and y_l is the less preferred output for a prompt x, DPO maximizes the likelihood of observed preferences. Formally, DPO adopts the following objective:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) \;=\; -\mathbb{E}_{(x,\, y_w,\, y_l) \sim \mathcal{D}}\left[\log \sigma\Big(\beta\Big[\log \tfrac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}\Big] \;-\; \beta\Big[\log \tfrac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\Big]\Big)\right], \qquad (20)$$

where $\sigma(\cdot)$ is the logistic sigmoid function, and $\beta \log \tfrac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$ represents a reparameterized reward difference between π_θ and the reference policy π_{ref}. By maximizing $\mathcal{L}_{\text{DPO}}$, the policy π_θ aligns with human preferences without requiring a separate reward model. Because the DPO objective inherits a KL-regularized formulation from RLHF, it preserves essential theoretical guarantees—such as consistency under well-defined preference assumptions [159]—while unifying the training procedure into a single stage. Consequently, DPO facilitates a more direct path for aligning language models with human evaluations, reducing system complexity and enhancing training stability.

4.3.2 Training Details of DPO

The DPO framework builds upon two core models: a reference policy π_{ref} and a target policy π_{tar}. The reference policy, typically a pre-trained and supervised fine-tuned language model, remains fixed throughout training. By contrast, the target policy is initialized from π_{ref} and iteratively updated using preference-based feedback, thereby improving alignment with human judgments. **Fig. 11** depicts this overall pipeline.
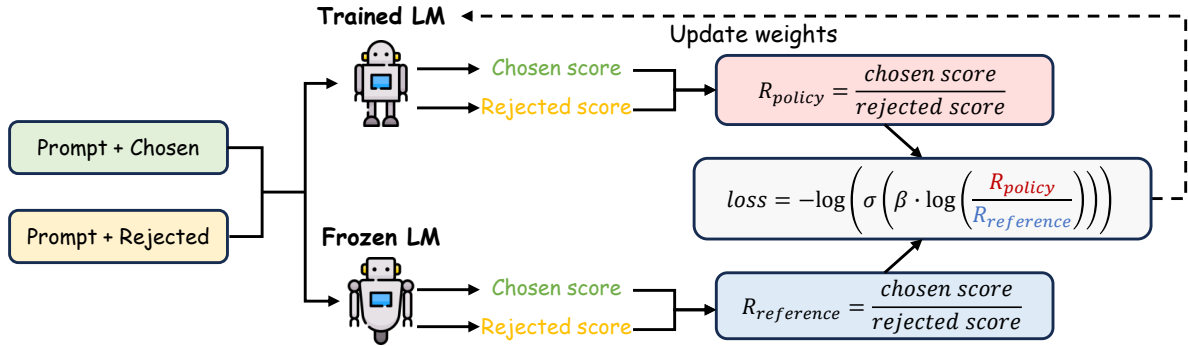

Figure 11: Workflow of Direct Preference Optimization (DPO), illustrating the training pipeline for optimizing Large Language Model outputs based on human preferences.

Data Collection and Preparation. DPO relies on a curated preference dataset obtained by sampling multiple candidate responses from π_{ref} for each prompt x. Human annotators then compare or rank these responses based on coherence, relevance, and clarity, among other criteria. The resulting preference labels serve as the core training signals for optimizing π_{tar}.

Training Procedure. The target policy is refined through a series of gradient-based updates aimed at minimizing the loss L_{DPO}. Specifically, 1) Generation: π_{ref} produces candidate outputs for each prompt x. 2) Annotation: human annotators compare the generated outputs, determining their relative preference. 3) Optimization: using these pairwise preferences, π_{tar} is iteratively updated to better emulate human-favored outputs. Throughout this process, π_{ref} remains unchanged, providing a stable baseline against which to measure improvements.

Practical Considerations. Selecting a robust reference policy is often critical to initializing DPO effectively. SFT typically yields a well-performing baseline for π_{ref}, ensuring that subsequent preference-driven updates can focus on refinement rather than fundamental skill acquisition. Additionally, preference data must be sufficiently diverse to capture variations in user expectations, thereby promoting model adaptability and preventing overfitting to narrowly defined tasks.

4.3.3 Variants of DPO

Multiple variants of DPO have emerged to address specific alignment challenges and optimize different aspects of text generation. **Tab. 2** contains an overview of these methods, which range from token-level generation optimizations to controlling verbosity and handling listwise or negative preferences.

DPO for Optimizing Generation. Token-level and iterative DPO strategies facilitate finer-grained or continuous alignment with human preferences. Reformulated as a bandit problem, token-level DPO [118] adopts a Markov Decision Process (MDP) defined by (S, A, f, r, ρ_0). This approach mitigates challenges such as excessive KL divergence for dispreferred tokens. TDPO [119] applies sequential forward KL divergence instead of reverse KL, improving both alignment and diversity preservation in text generation. Iterative DPO [111] adopts a multi-round approach to continuously refine outputs through repeated preference evaluations, often performed by the model itself. Pairwise Cringe Optimization (PCO) [109] extends binary feedback to a pairwise setting, using a soft margin to balance exploration and exploitation. Step-wise DPO [160] partitions the preference dataset and applies iterative updates, using the updated policy from each round as the baseline for the next.

Controllable and Flexible DPO. Some DPO variants aim to manage verbosity and reduce the need for a fixed reference policy. R-DPO [116] penalizes output length through a regularization term in the objective function, addressing overly verbose or redundant responses. SimPO [121] eliminates the requirement for a reference policy by normalizing response length and streamlining the loss function to handle both desirable and undesirable outputs. RLOO [112] leverages the REINFORCE algorithm without training a value model, substantially reducing computational overhead. It treats the entire response as a single action and learns from sparse rewards, simplifying implementation compared to traditional PPO-based methods.

Listwise DPO. Rather than limiting preference data to pairwise comparisons, Listwise DPO approaches optimize over sets of outputs. Listwise Preference Optimization (LiPO) [113] applies Learning-to-Rank techniques directly on ranked lists of candidate responses, improving efficiency relative to repeated pairwise comparisons. RRHF [106] incorporates preference alignment into SFT, eliminating the need for a separate reference model. PRO [107] breaks down listwise preferences into simpler binary tasks, simplifying alignment during SFT.

Negative DPO. Certain tasks require learning from undesired or harmful outputs: Negating Negatives (NN) [115] discards positive responses and maximizes divergence from less preferred outputs. Negative Preference Optimization (NPO) [120] employs gradient ascent on negative preferences, effectively reducing harmful outputs and mitigating catastrophic collapse.

5 PoLMs for Reasoning

Reasoning constitutes a central pillar for enabling LLMs to tackle tasks involving multi-step logic, intricate inference, and complex decision-making. This chapter examines two core techniques for enhancing model reasoning capabilities: **Self-Refine for Reasoning** (§5.1), which guides the model to autonomously detect and remedy errors in its own reasoning steps; and **Reinforcement Learning for Reasoning** (§5.2), which employs reward-based optimization to improve the consistency and depth of the model's chain-of-thought. These approaches collectively enable more robust handling of long-horizon decision-making, logical proofs, mathematical reasoning, and other challenging tasks.

5.1 Self-Refine for Reasoning

Reasoning remains a core challenge in optimizing LLMs for tasks that demand intricate logical inference and context-dependent decision-making. In this context, self-refine emerges as a powerful mechanism to iteratively pinpoint and correct errors during or after text generation, substantially improving both reasoning depth and overall reliability. As shown in **Fig. 12**, self-refine methods can be divided into four categories: Intrinsic Self-refine, which relies on the model's internal reasoning loops; External Self-refine, which incorporates external feedback resources; Fine-tuned Intrinsic Self-refine, which iteratively updates the model's reasoning processes based on self-generated corrections; and Fine-tuned External Self-refine, which harnesses external signals and fine-tuning to refine reasoning in a more adaptive, long-term manner. **Tab. 4** further illustrates how each category fortifies LLM reasoning capacity across various tasks.

Intrinsic Self-Refine. Intrinsic self-refine methods focus on empowering the model itself to detect and fix errors internally without resorting to outside tools. For instance, RCI Prompting [190] only triggers corrections when a contradiction or error is identified, avoiding overreactions to minor uncertainties. CAI Revi-
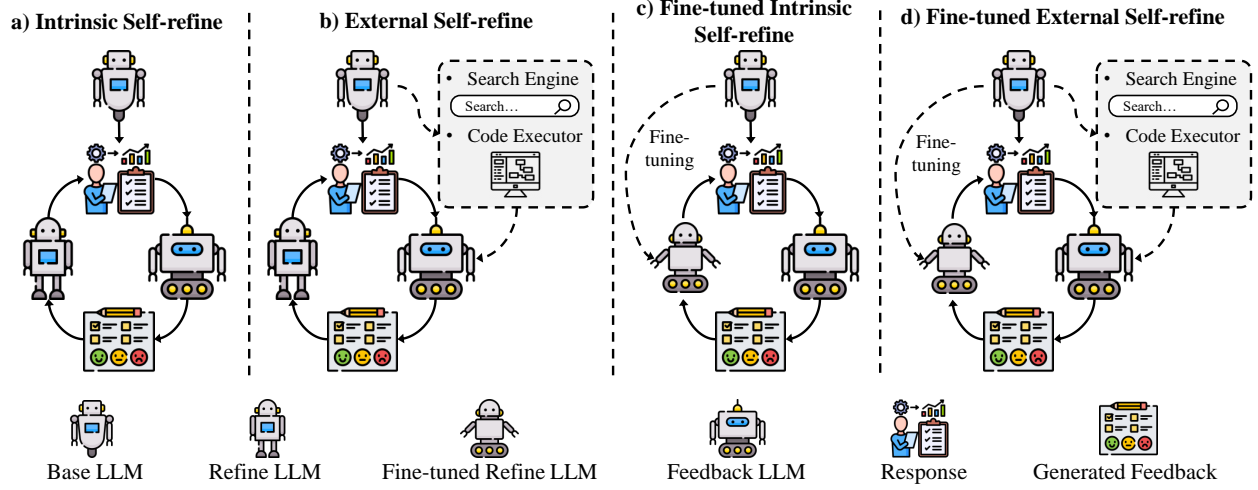
Figure 12: Taxonomy of Self-Refine methods, delineating architectural variations for enhancing reasoning in Large Language Models.

sions [105] corrects undesirable outputs (e.g., offensive text) while teaching the model to self-moderate its responses. Similarly, Self-Refine [164] leverages the transition from lower-quality prompts to high-fidelity instructions, refining the intermediate logic to boost consistency. CoVe [169] addresses multi-answer questions by dividing them into subtasks, each verified individually to ensure precision and consistency across the entire reasoning chain. Weak-to-Strong Generalization (W2SG) approaches leverage advanced algorithms to enable strong student models to learn effectively from noisy demonstrations produced by less capable teacher models [191]. This framework has seen several key developments and applications across different domains. Recent research has enhanced W2SG through various innovations. For instance, ensemble learning techniques have been successfully applied to improve the robustness and effectiveness of W2SG methods [192]. [193] adopt weak-to-strong extrapolation to enhance LLMs alignment.

External Self-Refine. These methods involve extrinsic feedback sources or computational tools to guide and correct the model's reasoning. CRITIC [177] systematically checks step-by-step outputs, enhancing the reliability of complex reasoning tasks. Reflexion [172] and Self-Debug [173] compare generated answers to reference solutions or few-shot exemplars, respectively, iteratively refining the logic. Techniques like FLARE [170] and Logic-LM [171] incorporate references from external documents or symbolic solvers, thereby minimizing logical missteps. RARR [165] and SelfEvolve [166] show that verifying intermediate states (e.g., compiler messages or relevant knowledge sources) is a powerful way to prune erroneous paths early and refine the model toward a correct solution. [194] proposes Iterative Preference Learning from Human Feedback, which includes an iterative version of the Direct Preference Optimization (DPO) algorithm for online settings, and a multi-step rejection sampling strategy for offline scenarios. PIT [195] implicitly learns the improvement goal from human preference data.

Fine-Tuned Intrinsic Self-Refine. By fine-tuning the base model specifically for internal revision, these approaches systematically strengthen the LLM's self-correction loops. Self-Critique [161] aims to improve summarization via self-review, while SelFee [174] uses iterative feedback loops to ensure higher levels of logical consistency. Volcano [180] reduces multimodal hallucinations by fine-tuning a dedicated corrector module within the LLM's architecture, and RL4F [167] harnesses RL-based critique loops to raise performance by an average of 10% on benchmarks requiring in-depth reasoning. REFINER [176] similarly concentrates on intermediate reasoning paths without changing the model's original generation process, demonstrating that consistent improvements can be achieved by training the model to carefully re-examine its partial outputs. Additionally, the concept of easy-to-hard generalization has emerged as a promising variant of W2SG, where models are initially trained on easily verifiable examples before tackling more complex tasks [196]. One notable implementation of this approach involves training a strong reward model on human-verifiable examples, which then guide the supervision of more capable models on challenging tasks [197]. In addition, the

Table 4: Overview of Self-Refine Methods in Large Language Models (2022–2025). This table summarizes prominent self-refinement techniques, detailing their primary LLMs, tasks, and release timelines across three metrics: **ET.** (External Tools: ✓ denotes usage, ⊗ denotes absence), **FT.** (Fine-Tuning: ✓ indicates application, ⊗ indicates non-application), and **SR.** (Self-Refine Type: IS for Intrinsic Self-refine, ES for External Self-refine, IF for Intrinsic Fine-tuning, EF for External Fine-tuning).

Methods	Main LLMs	Main Tasks	ET.	FT.	SR.	Release Time
Self-Critique [161]	InstructGPT	Topic Summarization	⊗	✓	IF	Jun-2022
CodeRL [162]	GPT-3.5	Program Synthesis	✓	✓	EF	Nov-2022
CAI Revisions [105]	52B (no details)	Detoxification	⊗	⊗	IS	Dec-2022
Baldur [163]	Minerva 8B, 62B	Proof Generation	✓	✓	EF	Mar-2023
Self-Refine [164]	GPT-3.5, GPT-4	Math, Coding, Dialogue	⊗	⊗	IS	May-2023
RARR [165]	Palm 540B	NQ, SQA, QReCC	✓	⊗	ES	May-2023
SelfEvolve [166]	InstructGPT, GPT-4	DS-1000, HumanEval	✓	⊗	ES	Jun-2023
RL4F [167]	GPT-3	Action Plan, Topic	⊗	✓	IF	Jul-2023
Self-Edit [168]	CodeGen, GPT-3.5	Code Generation	✓	✓	EF	Jul-2023
CoVe [169]	PaLM-540B	Multiple Answers	⊗	⊗	IS	Sep-2023
FLARE [170]	GPT-3.5	StrategyQA, ASQA	✓	⊗	ES	Oct-2023
Logic-LM [171]	GPT-3.5, GPT-4	PrOntoQA, Logic Reasoning	✓	⊗	ES	Oct-2023
Reflexion [172]	GPT-4	Games, Coding, HotpotQA	✓	⊗	ES	Oct-2023
Self-Debug [173]	GPT-3.5, GPT-4	Text-to-Code	✓	⊗	ES	Oct-2023
SelFee [174]	LLaMA-7B, 13B	MT-Bench	⊗	✓	IF	2023
RCI [175]	GPT-3.5-Turbo	Computer Taks, CSQA	⊗	⊗	IS	2023
REFINER [176]	GPT-3.5	Math, Logic, Moral Stories	⊗	✓	IF	Feb-2024
CRITIC [177]	GPT-3, LLaMA2-70B	GSM8k, SVAMP, HotpotQA	✓	⊗	ES	Feb-2024
ProMiSe [178]	FLAN-T5, LLaMA2-13B	MultiDoc2Dia, QuAC	✓	⊗	ES	Feb-2024
PREFER [179]	InstructGPT, GPT-4	NLI, NLC	⊗	⊗	IS	Mar-2024
Volcano [180]	GPT-3.5	Visual Reasoning	⊗	✓	IF	Apr-2024
CYCLE [181]	CodeGen, StarCoder	HumanEval, MBPP-S, APPS	⊗	✓	IF	Apr-2024
SRIT [182]	LLaMA2-7B, GPT-3.5	SIQA, PIQA, CSQA, OBQA	⊗	⊗	IS	May-2024
MCTSr [183]	LLaMA3-8B, GPT-4	GSM8K, GSM Hard, MATH	⊗	⊗	IS	Jun-2024
Self-Contrast [184]	GPT-3.5, L-70B	GSM8K, SVAMP, CommonMT	⊗	⊗	IS	Jun-2024
TEaR [185]	GPT-3.5, Claude-2	WMT22, WMT23	⊗	⊗	IS	Jun 2024
LLMRefine [186]	PaLM (Bison)	MQM, ASQA, Summ	⊗	✓	IF	Jun-2024
Self-Bias [187]	GPT-4, DeepSeek	Flores-200, MQM	✓	⊗	ES	Aug-2024
Exp-Refiner [188]	GPT-3.5, GPT-4	e-SNLI, QASC, WorldTree	⊗	⊗	IS	Oct-2024
Self-corrective [189]	GPT-2	GSM8k, SVAMP, Detoxification	⊗	✓	IF	Jan-2025

effectiveness of W2SG extends beyond LLMs, with successful applications demonstrated in computer vision tasks as well [198].

Fine-Tuned External Self-Refine. In scenarios where long-term improvements are crucial, the model's parameters are updated via external feedback mechanisms. For example, Self-Edit [168] regenerates code outputs based on execution results, leading to iterative improvements in correctness. Baldur [163] strengthens theorem proving by adding or modifying context, while CodeRL [162] employs test-based critics to verify functional accuracy in program synthesis tasks. Together, these techniques demonstrate that combining external resources with targeted fine-tuning fosters reliable, stepwise advancements in the model's overall reasoning performance.

5.2 Reinforcement Learning for Reasoning

In Subsection 5.1, we explored self-refine methods, a widely used approach to improve LLM reasoning through local tuning and optimization. This technique is typically applied to single-step tasks or output refinement, such as text generation and question answering, offering quick inference gains. However, it struggles with complex, long-term reasoning tasks requiring multi-step logic. The release of OpenAI's o1 series [41]

highlights reinforcement learning (RL) as a powerful alternative, training LLMs for advanced reasoning by refining long internal CoT through reward-based feedback. This significantly boosts performance in complex tasks like mathematical proofs and strategic planning. The o1 success has spurred research into large-scale RL, with models like QwQ-32B-Preview [199] excelling in mathematics and programming, and DeepSeek-R1 [28] matching o1's capabilities. This subsection examines RL's role in enhancing reasoning, focusing on DeepSeek-R1 and DeepSeek-R1-Zero, the leading open-source models.

5.2.1 Formulating Reasoning as an MDP

Reasoning within LLMs can be elegantly modeled as a sequential decision-making process, wherein the model iteratively constructs a series of intermediate steps $a_1, a_2, \ldots, a_T$ in response to an input query x to optimize the likelihood of arriving at a correct final answer. This conceptualization transforms reasoning into a structured framework amenable to reinforcement learning (RL), specifically through the lens of a Markov Decision Process (MDP), denoted as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$. The MDP encapsulates the dynamic interplay of states, actions, transitions, rewards, and temporal discounting, providing a robust mathematical foundation for training LLMs to navigate complex inference tasks. By framing reasoning as a sequence of deliberate choices, this approach enables the model to systematically explore and refine its logical pathways, drawing parallels to decision-making in domains like game playing or robotics, yet adapted to the unique challenges of linguistic and conceptual reasoning. The ultimate objective is to derive an optimal policy $\pi^*(a_t|s_t)$ that maximizes the expected cumulative reward, expressed as $J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=1}^{T} \gamma^t R(s_t, a_t) \right]$, leveraging RL techniques such as Proximal Policy Optimization (PPO) [46] or Advantage Actor-Critic (A2C) [200] to iteratively enhance reasoning capabilities based on environmental feedback.

State Space. The state space $\mathcal{S}$ forms the backbone of this MDP, with each state $s_t \in \mathcal{S}$ representing the current reasoning trajectory at timestep t, a rich composite of linguistic and structural elements critical to the inference process. Specifically, s_t encompasses the initial query x, the sequence of prior reasoning steps $\{a_1, \ldots, a_{t-1}\}$, and an internal memory representation that encodes logical dependencies and intermediate conclusions, such as partial solutions or inferred relationships. This state evolves dynamically as reasoning unfolds, mirroring the progression of thought by integrating both the explicit path articulated through generated steps and latent knowledge distilled from contextual understanding. For instance, in a mathematical proof, s_t might include the problem statement, previously derived equations, and a memory of applicable theorems, enabling the model to maintain coherence across steps. This multifaceted state representation ensures that the LLM can adaptively track its reasoning context, a prerequisite for tackling tasks requiring sustained logical continuity, such as multi-step problem-solving or narrative coherence in text generation.

Action Space. The action space $\mathcal{A}$ defines the range of possible decisions at each step, where an action $a_t \in \mathcal{A}$ corresponds to the selection of the next reasoning move, offering a versatile toolkit for advancing the inference process. These actions may include generating a token or phrase in natural language to articulate a reasoning segment, applying a predefined logical or mathematical transformation (e.g., algebraic simplification), selecting a relevant theorem or rule from a knowledge base to extend the reasoning chain, or halting the process upon reaching a conclusive answer. The action space's nature varies by task: it may be discrete, as in choosing from a finite set of logical rules in formal proofs, or continuous, as in producing free-form text in open-ended reasoning scenarios, reflecting the LLM's generative flexibility. This duality allows the model to navigate both structured domains, like symbolic logic, and unstructured ones, like commonsense reasoning, adapting its strategy to the task's demands while maintaining a coherent trajectory towards the solution.

Transition Function. Transition dynamics, encapsulated by the function $P(s_{t+1}|s_t, a_t)$, govern how the state evolves with each action, delineating the progression of the reasoning trajectory within the MDP framework. In contrast to traditional RL environments where stochasticity arises from external variables (e.g., environmental noise), reasoning transitions in LLMs are predominantly deterministic, driven by the model's autoregressive outputs or structured inference rules, such as applying a deductive step in a proof. However, uncertainties emerge from inherent model limitations—such as imperfect knowledge, ambiguous intermediate states, or probabilistic sampling in text generation—introducing variability that RL must address. For autoregressive LLMs, transitions follow a predictable sequence generation process, yet the potential for error accumulation or divergent interpretations necessitates a robust design to ensure reliability. This deterministic-

yet-uncertain dynamic underscores the need for adaptive policies that can stabilize reasoning across diverse contexts, from precise mathematical derivations to nuanced narrative constructions.

Reward Function. The reward function $R(s_t, a_t)$ serves as the evaluative core of the MDP, providing critical feedback on the quality of each reasoning step to guide the model's learning process. Unlike conventional RL tasks with explicit rewards (e.g., points in a game), reasoning rewards must be carefully engineered to balance sparsity and density, reflecting the task's complexity and goals. Sparse rewards, such as assigning a value only upon reaching a correct final answer, offer simplicity but may delay learning in multi-step scenarios, while dense rewards, which assess step-wise correctness, logical validity, or alignment with human preferences, provide granular guidance, as elaborated in §5.2.2. This flexibility allows the reward function to adapt to diverse reasoning demands—whether rewarding the application of a valid inference rule in a proof or the coherence of a narrative segment—ensuring that the model receives meaningful signals to refine its strategy across both immediate and extended inference horizons.

Discount Factor. γ: A scalar $\gamma \in [0, 1]$ that determines the trade-off between immediate and future rewards. A higher γ encourages multi-step reasoning optimization, promoting deep inference chains rather than short-term heuristics. Given this MDP formulation, the objective is to learn an optimal reasoning policy $\pi^*(a_t|s_t)$ that maximizes the expected cumulative reward:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=1}^{T} \gamma^t R(s_t, a_t) \right]. \tag{21}$$

This framework enables the application of reinforcement learning techniques such as such as Proximal Policy Optimization (PPO) [46] or Advantage Actor-Critic (A2C) [200] to refine LLM reasoning capabilities by iteratively adjusting the policy π_θ based on feedback from the reasoning environment.

5.2.2 Reward Design for Reasoning

Unlike traditional RL tasks with clear rewards like game scores, reasoning in LLMs demands structured reward designs reflecting correctness, efficiency, and informativeness. Common approaches include binary correctness rewards, assigning $r_T = 1$ for a correct final answer and $r_T = 0$ otherwise, which is simple but introduces high variance due to sparse feedback; step-wise accuracy rewards, offering incremental feedback based on metrics like inference rule validity or intermediate step consistency to guide multi-step reasoning; self-consistency rewards, measuring stability across multiple reasoning paths and assigning higher rewards for agreement to enhance robustness; and preference-based rewards, derived from RLHF or RLAIF, where a model $r_\phi(s_t, a_t)$ trained on human or AI feedback evaluates reasoning quality, providing nuanced guidance for complex tasks.

5.2.3 Large-Scale RL on Base Model

Large-scale Reinforcement Learning has emerged as a transformative post-training paradigm for enhancing the reasoning capabilities of LLMs, shifting the focus from traditional SFT to dynamic, self-evolving optimization strategies. This approach leverages extensive computational frameworks and iterative reward-based feedback to refine base models directly, bypassing the need for pre-annotated datasets and enabling autonomous development of complex inference skills. By integrating large-scale RL, LLMs can address intricate multi-step reasoning tasks (e.g., mathematical problem-solving, logical deduction, and strategic planning), where conventional SFT often falls short due to its reliance on static, human-curated data [45]. The DeepSeek-R1 model exemplifies this paradigm, employing advanced RL techniques to achieve state-of-the-art reasoning performance while optimizing resource efficiency, as illustrated in **Fig. 13**. This subsection delineates the key methodologies underpinning DeepSeek-R1's success, including novel optimization algorithms, adaptive exploration, and trajectory management, which collectively redefine the potential of RL-driven reasoning in LLMs.

Group Relative Policy Optimization. The DeepSeek-R1-Zero model leverages a sophisticated variant of Proximal Policy Optimization (PPO), termed Group Relative Policy Optimization (GRPO), to mitigate the substantial computational and resource demands inherent in traditional RL training for LLMs. Unlike standard PPO, which relies on extensive critic networks, GRPO employs a group-based baseline estimation to
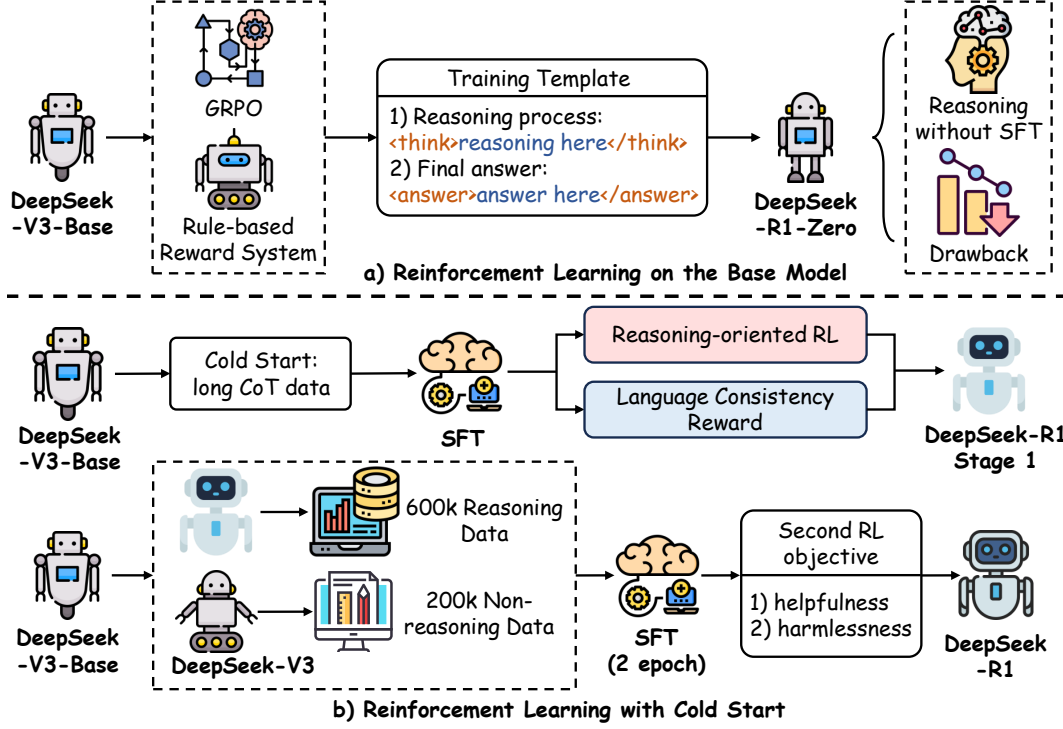
Figure 13: Workflow of Reinforcement Learning for Reasoning in DeepSeek-R1, illustrating the process for optimizing reasoning capabilities in Large Language Models.

streamline the optimization process, significantly reducing training overhead while preserving the robustness of policy updates. This efficiency enables large-scale RL deployment on resource-constrained systems, facilitating iterative refinement of reasoning strategies across extended trajectories. By optimizing the policy within manageable computational bounds, GRPO positions DeepSeek-R1-Zero as a scalable solution for enhancing reasoning capabilities, as depicted in **Fig. 13**, making it a cornerstone of contemporary RL-driven inference research.

DeepSeek-R1-Zero. DeepSeek-R1-Zero exemplifies the transformative potential of large-scale RL to elevate LLM reasoning without the conventional reliance on SFT as an initial step, instead adopting a pure RL-driven self-evolution paradigm. This approach enables the model to autonomously develop sophisticated reasoning skills by iteratively refining its internal CoT through reward feedback, bypassing the need for pre-annotated datasets typically required in SFT. The result is a marked improvement in performance across complex, multi-step reasoning tasks (e.g., mathematical problem-solving and logical derivations) demonstrating RL's capacity to unlock advanced inference capabilities from a base model. Positioned as one of the strongest open-source reasoning models, DeepSeek-R1-Zero's success underscores the viability of cold-start RL strategies, offering a resource-efficient alternative to traditional training pipelines while achieving parity with state-of-the-art benchmarks.

Stepwise Reward Modeling. To guide reasoning across a trajectory $\tau = (s_1, a_1, \ldots, s_T, a_T)$, DeepSeek-R1 employs a stepwise reward model f_θ that delivers granular feedback at each timestep, defined as $r_t = f_\theta(s_t, a_t \mid \mathcal{D}_{\text{reasoning}})$, where $\mathcal{D}_{\text{reasoning}}$ comprises human-annotated CoT sequences with step-level correctness labels. This dense reward structure contrasts with sparse end-of-sequence rewards by providing immediate, actionable insights into the quality of individual reasoning steps, enabling the model to fine-tune its strategies with precision. By leveraging expertly curated data, the reward model ensures that feedback aligns with human reasoning standards, fostering consistency and accuracy across extended inference chains, a critical feature for tackling tasks requiring protracted logical synthesis.

Adaptive Exploration. DeepSeek-R1 enhances policy optimization through an adaptive exploration mechanism integrated into its objective:

$$\mathcal{L}_{\text{PPO+}} = \mathbb{E}_\tau \left[\min \left(\frac{\pi_\phi(a|s)}{\pi_{\text{old}}(a|s)} A_t, \text{clip} \left(\frac{\pi_\phi(a|s)}{\pi_{\text{old}}(a|s)}, 1 - \epsilon, 1 + \epsilon \right) A_t \right) \right]$$
$$+ \lambda_t \mathcal{H}(\pi_\phi(\cdot|s)), \tag{22}$$

where the entropy term $\mathcal{H}$ is modulated by an adaptive coefficient $\lambda_t = \alpha \cdot \exp(-\beta \cdot \text{Var}(R(\tau_{1:t})))$, dynamically adjusting based on reward variance across the trajectory. This approach balances exploration and exploitation, encouraging the model to explore diverse reasoning paths early in training while converging to optimal strategies as variance decreases, thereby enhancing both robustness and efficiency in reasoning refinement.

Trajectory Pruning. To optimize computational efficiency during reasoning, DeepSeek-R1 implements a dual-attention critic $V_\psi(s_t) = \text{LocalAttn}(s_t) + \text{GlobalAttn}(s_{1:t})$, which evaluates the value of each state by combining local step assessments with global trajectory context. Pruning occurs when $V_\psi(s_t) < \gamma \cdot \max_{k \leq t} V_\psi(s_k)$, terminating low-value reasoning paths to focus resources on promising trajectories. This mechanism reduces wasteful exploration, accelerates convergence, and ensures that the model prioritizes high-quality reasoning sequences, contributing to its exceptional performance in complex inference tasks.

5.2.4 RL for Reasoning with Cold Start

DeepSeek-R1-Zero further advances RL's application by adopting a cold-start approach, eschewing SFT and relying entirely on large-scale RL from an untrained base model. This self-evolutionary strategy refines reasoning through iterative feedback, generating robust CoT sequences without pre-annotated data dependencies. By training directly on reasoning tasks, DeepSeek-R1-Zero demonstrates RL's versatility, achieving performance comparable to or exceeding models initialized with SFT, such as its DeepSeek-R1 counterpart. This approach not only reduces reliance on extensive labeled datasets but also showcases RL's potential to autonomously develop complex reasoning capabilities, offering a scalable paradigm for future LLM development. Collectively, RL provides a promising framework for enhancing reasoning, with effective reward design, policy optimization (e.g., GRPO), and exploration strategies remaining critical. Future research could explore hybrid methods integrating imitation learning or self-supervised objectives to further refine these capabilities, solidifying RL's role in advancing LLM inference.

6 PoLMs for Efficiency

Building on the post-training optimization techniques discussed in earlier chapters, post-training efficiency specifically targets the operational performance of LLMs after their initial pre-training. The principal goal is to optimize key deployment metrics (e.g., processing speed, memory usage, and resource consumption), thereby making LLMs more practical for real-world applications. Approaches to achieving post-training efficiency fall into three main categories: **Model Compression** (§6.1), which reduces the overall computational footprint through techniques such as pruning and quantization; **Parameter-Efficient Fine-Tuning** (§6.2), which updates only a fraction of a model's parameters or employs specialized modules, thus minimizing retraining costs and accelerating adaptation to new tasks; and **Knowledge Distillation** (§6.3), which transfers the knowledge from a larger, pre-trained model to a smaller model, enabling the smaller model to achieve comparable performance with reduced resource demands.

6.1 Model Compression

Model compression encompasses a set of techniques designed to reduce the size and computational demands of LLMs, which includes post-training quantization, parameter pruning, and low-rank approximation.

6.1.1 Post-training Quantization

A crucial compression method for LLMs is quantization, which converts high-precision data types X^H (30-bit floating point) into lower-precision formats X^L (8-bit integer)[201]. This conversion is formulated as:

$$X^L = \text{Round}(\frac{\text{absmax}(X^L)}{\text{absmax}(X^H)} X^H) = \text{Round}(\mathcal{K} \cdot X^H), \qquad (23)$$

where $\mathcal{K}$ represents the quantization constant, and absmax refers to the absolute maximum of the elements. The function Round transforms floating-point numbers into integers. LLM quantization encompasses both post-training quantization (PTQ) and quantization-aware training (QAT). PTQ enables adjustments to model weights and activations after pre-training, using a small calibration dataset to optimize for both computational efficiency and performance as illustrated in **Fig. 14**. Additionally, **Tab. 5** presents the performance metrics of several prominent quantization methods for LLMs.
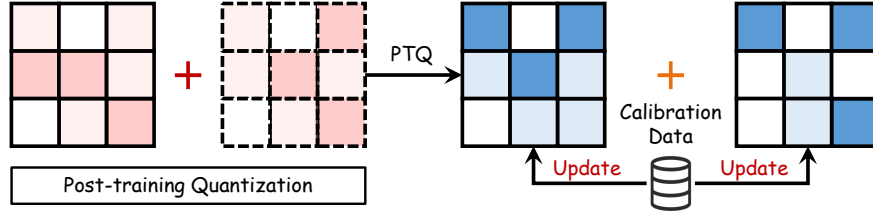

Figure 14: Illustrations of post-training quantization techniques for LLMs.

Weight-Only Quantization (WOQ). WOQ focuses on compressing model weights to improve efficiency. GPTQ [230] applies layer-wise quantization using Optimal Brain Quantization (OBQ), reducing weights to 3 or 4 bits to lower memory usage and processing time. To push efficiency further, QuIP [203] introduces incoherence processing for 2-bit quantization, offering an even more compact representation. Similarly, AWQ [204] and OWQ [205] address accuracy retention by maintaining high precision for particularly sensitive weights, thereby minimizing potential accuracy losses during inference. Finally, SpQR [201] combines sparse quantization with decoding, enabling efficient token-by-token inference while preserving model responsiveness.

Weight-Activation Co-Quantization (WAQ). WAQ integrates weights and activations to enhance efficiency. LLM.int8() [214] addresses activation outliers using precise storage and quantizes to 8 bits while maintaining performance. SmoothQuant [218] implements per-channel scaling, transferring the quantization difficulties from activations to weights for lossless results. Additionally, OS+ [219] mitigates the impact of outliers through channel-wise shifting and scaling, thereby boosting efficiency. OmniQuant [220] redirects the quantization hurdles from activations to weights and fine-tunes clipping thresholds for extreme values. To further enhance efficiency, RPTQ [231] groups similar channels to ensure uniformity in quantization parameters.

KV-Cache Quantization (KVQ). KV-Cache Quantization addresses memory optimization challenges in LLMs, especially as input token counts increase. KVQuant [224] introduces tailored methods for efficient inference with large context lengths, maintaining performance with minimal loss. KIVI [228] optimizes memory savings by applying distinct quantization strategies for key and value caches, achieving 2-bit quantization without fine-tuning. WKVQuant [225] further refines this with a two-dimensional quantization strategy and cross-block regularization, delivering memory efficiency comparable to weight-activation quantization with nearly the same performance.

6.1.2 Parameter Pruning

Parameters Pruning [232] is a crucial technique for improving the efficiency of LLMs by minimizing model size and complexity without sacrificing accuracy. As shown in **Fig. 15**, pruning can be divided into Unstructured Pruning, and Structured Pruning.

Unstructured Pruning. Unstructured pruning enhances the sparsity of LLMs by eliminating weights that are not critical. The approach known as SparseGPT [230] achieves as much as 60% sparsity through one-

Table 5: Overview of Quantization Methods for Large Language Models (2021–2025). This table summarizes representative quantization techniques, detailing their main LLMs, bit widths, perplexity differences, speedups, and release timelines across three metrics: **Bit Width** (bits for weights, activations, and KV cache), **Perplexity Difference** (performance variation on Wikitext-2 and C4 datasets), and **Speedup** (computational speed improvement relative to baseline models).

Type	Methods	Main LLMs	Bit Width			Perplexity Difference		Speedup	Release Time
			W	A	KV	Wikitext-2	C4		
WOQ	GPTQ [202]	OPT-175B	3	16	16	0.34	0.23	3.24×	Mar-2023
	SpQR [201]	LLaMA-30B	3.89	16	16	0.15	0.1	2.0×	Jun-2023
	QuIP [203]	LLaMA2-70B	2	16	16	3.007	3.228	-	2023
	AWQ [204]	LLaMA2-70B	3	16	16	0.42	-	3.2×	2024
	OWQ [205]	LLaMA-65B	3.01	16	16	0.72	-	-	Mar-2024
	EasyQuant [206]	LLaMA-65B	4	16	16	3.98	6.30	-	Mar-2024
	Agile-Quant [206]	BLOOM-7.1B	8	8	16	11.73	14.70	1.8x	Mar-2024
	LUT-GEMM [207]	LLaMA-65B	3	16	16	0.14	-	2.04×	Apr-2024
	SqueezeLLM [208]	LLaMA-13B	3	16	16	0.51	0.67	2.4×	Jun-2024
	DAQ [209]	LLaMA2-30B	3	16	16	4.23	-	-	Oct-2024
	MobileQuant [210]	TinyLlaMA-1.1B	4	16	16	15.6	-	-	Oct-2024
	GWQ [211]	LLaMA2-13B	4.63	16	16	4.88	6.47	1.2×	Dec-2024
WAQ	QT [212]	OPT-1.3B	8	8	16	17.74	-	-	Mar-2021
	ZeroQuant [213]	GPT-J-6B	8	8	16	0.16	-	3.67×	2022
	LLM.int8() [214]	OPT-13B	8	8	16	-	0.00	1.22×	2022
	RPTQ [215]	OPT-175B	4	4	16	2.26	2.15	-	May-2023
	Olive [216]	BLOOM-7B	4	4	16	2.11	2.24	4.5×	Jun-2023
	ZeroQuant-FP [217]	LLaMA-30B	4	8	16	0.18	0.13	-	Jul-2023
	SmoothQuant [218]	OPT-175B	8	8	16	0.18	-	1.56×	Oct-2023
	OS+ [219]	LLaMA-65B	4	4	16	5.77	-	-	Oct-2023
	OmniQuant [220]	LLaMA-7B	4	6	16	0.41	0.55	-	Mar-2024
	RoLoRA [221]	LLaMA3-8B	4	16	16	-	-	-	Sep-2024
	HotaQ [222]	LLaMA	4	4	8	-	-	2.5×	Oct-2024
	Q-DiT [223]	LLaMA3-8B	6	8	16	-	-	-	Nov-2024
KVQ	KVQuant [224]	LLaMA-65B	16	16	2	0.19	0.11	1.4×	Jan-2024
	WKVQuant [225]	LLaMA-13B	4	16	4	0.12	0.14	-	Feb-2024
	QAQ [226]	LLaMA3-8B	4	16	16	-	-	10×	Apr-2024
	ZipCache [227]	LLaMA3-8B	4	16	16	0.38	-	4.98×	May-2024
	KIVI [228]	LLaMA2-13B	4	16	16	-	-	2.6×	Jul-2024
	DL-QAT [229]	LLaMA2-7B	4	16	16	6.3	-	-	Nov-2024

shot pruning while maintaining minimal loss. The method Wanda [233] performs pruning based on weight magnitudes and activations without requiring retraining. Meanwhile, SAMSP [234] leverages the sensitivity of the Hessian matrix for dynamic adjustments to sparsity, aimed at minimizing errors. DSnoT [235] improves performance by employing iterative pruning cycles. Finally, Flash-LLM [236] retrieves sparse weights from global memory and reconstructs them densely in on-chip buffers to facilitate efficient computation.

Structured Pruning. This approach focuses on pruning entire parameter groups in LLMs to enhance hardware efficiency and simplify structures. For instance, LLM-runer [237] assesses importance for LLaMA [65] and uses LoRA [92] to recover accuracy post-pruning. FLAP [238] optimizes compression without fine-tuning using a structured metric. Additionally, SliceGPT [239] employs PCA for pruning while maintaining efficiency. Sheared LLaMA [240] refines model shape with regularization-based pruning. LoRAPrune [241] enhances efficiency with iterative structural pruning based on LoRA importance. Furthermore, Deja Vu [242] reduces latency, maintaining accuracy by predicting key attention heads and MLP parameters, using contextual sparsity.
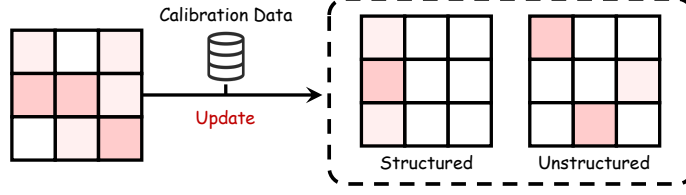
Figure 15: Illustrations of pruning parameters techniques for LLMs.

Low-Rank Approximation. Low-rank approximation serves to compress LLMs by approximating a weight matrix W with smaller matrices U and V, thereby achieving $W \approx UV^\top$. This methodology not only reduces the number of parameters but also enhances operational efficiency. For instance, TensorGPT [243] employs Tensor-Train Decomposition (TTD) to develop a more efficient embedding format. LoSparse [244] integrates low-rank approximation with pruning, specifically aimed at compressing coherent neuron components. FWSVD [245] implements a weighted SVD approach, whereas ASVD [246] provides a training-free SVD alternative, both targeting post-training efficiency. Lastly, SVD-LLM [247] further improves compression by establishing a direct relationship between singular values and compression loss.

6.2 Parameter-Efficient Fine-Tuning

The procedure for parameter-efficient fine-tuning (PEFT) consists of freezing the complete LLM backbone while only modifying a limited number of newly added parameters. As depicted in **Fig. 16**, PEFT methods are divided into four categories: additive PEFT, selective PEFT, reparameterized PEFT, and hybrid PEFT.

6.2.1 Additive PEFT

Additive PEFT incorporates new trainable modules to the LLM without changing the original parameters, allowing task-specific tuning while retaining the base model's knowledge, which is efficient for fine-tuning.
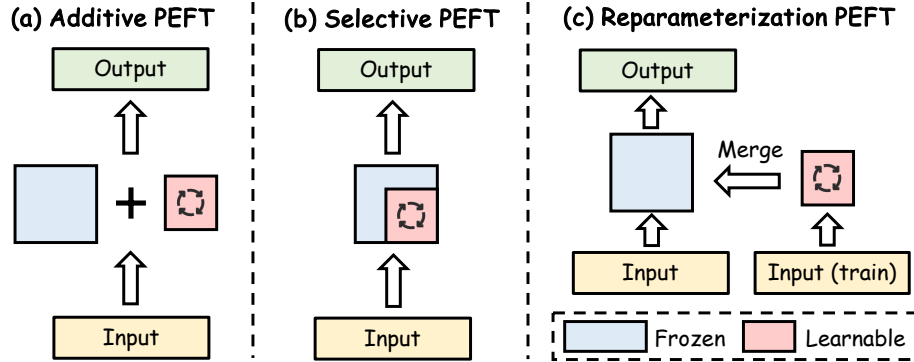

Figure 16: Illustrations of Parameter-Efficient Fine-Tuning (PEFT), illustrating approaches for resource-efficient adaptation in Large Language Models.

Adapters. Adapter integrates compact layers within transformer blocks, defined as:

$$\text{Adapter}(x) = W_{\text{up}}\sigma(W_{\text{down}}x) + x, \tag{24}$$

where an adapter layer includes a down-projection matrix $W_{\text{down}} \in \mathbb{R}^{r \times d}$, a non-linear activation σ, and an up-projection matrix $W_{\text{up}} \in \mathbb{R}^{d \times r}$. Here, d is the hidden layer dimension, and r is the bottleneck dimension, reducing complexity while maintaining performance. Building on this structure, Serial Adapter [248] introduced two modules per transformer block. AdapterFusion [249] improved efficiency by placing adapters post $Add\&Norm$. Parallel Adapter (PA) [250] ran adapters parallel to sublayers, while CoDA [251] optimized by running adapters in parallel with sublayers. Unlike AdapterFusion, MerA [252] unified adapters using optimal transport techniques for weights and activations.

Soft Prompt. Soft prompt enhances model performance by adding adjustable vectors to the input sequence instead of optimizing discrete tokens [253]. This approach is formalized as:

$$X^{(l)} = [s_1^{(l)}, \ldots, s_{N_S}^{(l)}, x_1^{(l)}, \ldots, x_{N_X}^{(l)}], \tag{25}$$

where $s_i^{(l)}$ denotes soft prompt tokens, and $x_i^{(l)}$ represents original input tokens. N_S and N_X are the counts of soft prompt and original input tokens, respectively. Prefix Tuning [254] introduces learnable vectors between transformer layers, stabilized by reparameterization and refined by P-Tuning v2 [99] and APT [255]. Meanwhile, Prompt Tuning [44] focuses on the initial embedding layer for large model optimization with low computational cost. Xprompt [256] and IDPG [257] streamline prompt generation and insertion. Methods like SPoT [258] and PTP [259] then address stability and convergence speed, while DePT [260] and SMoP [261] reduce computational demands through optimized prompt structures.

Other Additive Methods. In addition to earlier techniques, methods such as (IA)^3 [262] and SSF [263] focus on post-training efficiency by introducing minimal but powerful adjustments to model parameters. The self-attention and FFN operations are mathematically defined as:

$$SA(x) = \text{Softmax}\left(\frac{Q \cdot (l_k \odot K)^T}{\sqrt{d_{head}}}\right) \cdot (l_v \odot V), \tag{26}$$

$$FFN_{transformer}(x) = W_{up} \cdot (l_{ff} \odot \sigma(W_{down}x)), \tag{27}$$

where $\odot$ represents the Hadamard product, the scale vectors l_k and l_v can be smoothly incorporated into the weight matrices of A_Q and A_W. Additionally, IPA [264] aligns LLMs like GPT-4 with user-specific requirements. Moreover, it does not require changes to the underlying model and therefore maintains efficiency during the fine-tuning process.

6.2.2 Selective PEFT

Selective PEFT enhances efficiency by fine-tuning only a subset of parameters, as shown in **Fig. 16(b)**. This involves applying a binary mask $M = \{m_1, m_2, \ldots, m_n\}$ to the parameters $\theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$, where each m_i indicates if θ_i is selected for fine-tuning. The updated parameter set is expressed as:

$$\theta_i' = \theta_i - \eta \cdot m_i \cdot \frac{\partial \mathcal{L}}{\partial \theta_i}, \tag{28}$$

where η is the learning rate, and $\frac{\partial \mathcal{L}}{\partial \theta_i}$ is the gradient of the loss function. Only selected parameters (where $m_i = 1$) are updated, reducing computational costs while maintaining effectiveness. Early approaches include Diff pruning [265], which regularizes a learnable binary mask using a differentiable L_0-norm, and FishMask [266], which selects parameters based on fisher information for greater relevance. LT-SFT [267] applies the Lottery Ticket Hypothesis to identify impactful parameters. SAM [268] employs second-order approximations for selection, while Child-tuning [269] selects parameters dynamically in a child network. Additionally, FAR [270] and BitFit [271] further exemplify selective PEFT by focusing on optimizing specific parameter groups.

6.2.3 Reparameterized PEFT

Reparameterized PEFT mainly employs a low-rank parameterization to improve efficiency, as illustrated in **Fig. 16(c)**. LoRA (Low Rank Adaptation)[92] introduces two trainable matrices, $W_{up} \in \mathbb{R}^{d \times r}$ and $W_{down} \in \mathbb{R}^{r \times k}$, modifying the output as:

$$h_{out} = W_0 h_{in} + \alpha(W_{up} W_{down} h_{in}), \tag{29}$$

where α is a scaling factor. This approach allows efficient adaptation to new tasks while preserving core knowledge. Building on LoRA, Intrinsic SAID [272] minimizes the fine-tuning parameter space, further reducing computational demands. Dynamic variants, including DyLoRA [273] and AdaLoRA [274], adapt rank dynamically based on task-specific requirements, with AdaLoRA additionally incorporating SVD-based pruning for greater efficiency. SoRA [275] simplifies processes by removing orthogonality constraints, while

35

Laplace-LoRA [276] applies Bayesian calibration for fine-tuning. Compacter [277] and VeRA [278] further reduce parameter complexity. Additionally, DoRA [279] optimizes updates in the directional component and HiRA [280] employs the Hadamard product for high-rank updates, thereby enhancing both efficiency and performance. To tackle multiple tasks and evolving domains, Terra [281] integrates a time-varying matrix, and ToRA [282] utilizes a Tucker decomposition to further improve the LoRA structure. In addition to structural design, PiSSA [283] and LoRA-GA [284] optimize LoRA's initialization using SVD and gradient alignment. Meanwhile, LoRA+ [285], LoRA-Pro [286], and CopRA [287] further refine the gradient update strategies. Additionally, ComLoRA [288] employs competitive learning to select the best-performing LoRA component.

6.2.4 Hybrid PEFT

Hybrid PEFT methods enhance post-training efficiency by integrating or optimizing various fine-tuning strategies. A prominent technique, UniPELT [289], merges LoRA, prefix tuning, and adapters within transformer blocks. This method dynamically activates components through a gating mechanism managed by feedforward networks (FFNs) that produce scalars $G \in [0, 1]$, ultimately optimizing parameter utilization. Another innovative method, MAM Adapter [250], refines this technique by strategically positioning prefix tuning in self-attention layers and using scaled parallel adapters in feedforward layers. Furthermore, NAS-based approaches such as NOAH [290] and AUTOPEFT [291] improve post-training efficiency by identifying optimal PEFT configurations tailored to specific tasks. Additionally, HeadMap [292] identifies a series of attention heads (i.e., knowledge circuits) that play a key role in certain tasks using a greedy approach, and efficiently improves model performance by mapping the output of these attention heads back into the residual flow of LLM. Finally, LLM-Adapters [293] provide a framework for integrating various PEFT techniques within LLMs, ensuring the most effective module placements to maintain efficiency across different model scales.

6.3 Knowledge Distillation

Knowledge Distillation (KD) constitutes a cornerstone technique in the post-training optimization of LLMs, enabling the transfer of knowledge from a large, pre-trained teacher model to a compact student model to enhance efficiency without sacrificing performance. Initially introduced in the context of model compression, KD has garnered substantial attention for its capacity to distill complex knowledge into resource-efficient architectures, enabling deployment in constrained environments such as edge devices and embedded systems. By leveraging the nuanced output distributions of a teacher model—richer than traditional hard labels—KD empowers the student to replicate not only class predictions but also the inter-class relationships and subtle patterns ingrained in the teacher's representations. This process typically involves optimizing a composite loss function that balances supervised learning objectives with distillation-specific goals, substantially reducing computational and memory demands while preserving generalization capabilities.

The fundamental mechanism of KD hinges on minimizing a hybrid loss that integrates traditional classification loss with a distillation term. Formally, given a teacher model's soft output probabilities $\mathbf{p_t}$ and a student model's predictions $\mathbf{p_s}$, alongside true labels $\mathbf{y}$ and student outputs $\mathbf{y_s}$, the KD loss is expressed as:

$$\mathcal{L}_{KD} = \alpha \mathcal{L}_{CE}(\mathbf{y}, \mathbf{y_s}) + (1 - \alpha)\mathcal{L}_{KL}(\mathbf{p_t}, \mathbf{p_s}), \tag{30}$$

where $\mathcal{L}_{CE}$ denotes the cross-entropy loss capturing alignment with ground truth, $\mathcal{L}_{KL}$ represents the Kullback-Leibler divergence [294] measuring divergence between teacher and student distributions, and $\alpha \in [0, 1]$ is a hyperparameter modulating the trade-off between these objectives. The soft targets $\mathbf{p_t}$, often tempered by a temperature parameter T (i.e., $\mathbf{p_t} = \text{softmax}(\mathbf{z_t}/T)$, where $\mathbf{z_t}$ are teacher logits), encode richer probabilistic information, enabling the student to emulate the teacher's decision-making nuances beyond mere label accuracy.

KD is widely used in model compression for resource-limited settings and transfer learning, where a pre-trained teacher guides a task-specific student. Its effectiveness depends on factors such as teacher capacity, student architecture, and distillation loss design. Recent advances extend KD beyond output distillation, enabling more efficient and adaptable LLMs in post-training optimization. KD methods can be broadly categorized into black-box KD and white-box KD, depending on the level of access to the teacher model's

Table 6: Summary of Knowledge Distillation Methods for Large Language Models (2020–2025). This table outlines key distillation techniques, detailing their skills, teacher and student models, objectives, and release timelines, categorized as **Black-box KD** (access limited to teacher outputs, typically from closed-source LLMs) and **White-box KD** (access to teacher parameters or distributions, usually from open-source LLMs). Metrics include IF (Instruction Following), CoT (Chain-of-Thought), ICL (In-context Learning), SFT (Supervised Fine-Tuning), D&S (Divergence and Similarity), RL (Reinforcement Learning), TP (Think Pattern), NLU (Natural Language Understanding), and NLG (Natural Language Generation).

Type	Methods	Skill	Teacher Model	Student Model	Objective	Release Time
	SELF-INSTRUCT [295]	IF	T5-LM, GPT-3	GPT-3	SFT	Mar-2022
	MT-CoT [296]	CoT	GPT-3	T5	SFT	Oct-2022
	CoT Prompting [297]	CoT	GPT-3, PaLM	T5	SFT	Dec-2022
	Fine-tune-CoT [298]	CoT	GPT-3	T5, GPT-2, GPT-3	SFT	Dec-2022
	SOCRATIC CoT [299]	CoT	GPT-3	GPT-2	SFT	Dec-2022
	ICL Distillation [300]	ICL	GPT-2, BERT	GPT-2, BERT	SFT	Dec-2022
	SSLM [301]	CoT	GPT-3.5	T5	SFT	Jan-2023
	LaMini-LM [302]	IF	ChatGPT	Various Models	SFT	Apr-2023
	SCOTT [303]	CoT	GPT-neox	T5-3b	SFT	May-2023
Black-box KD	Distilling Step-by-Step [304]	CoT	PaLM	T5	SFT	May-2023
	Lion [305]	IF	ChatGPT	LLaMA	-	May-2023
	PaD [306]	CoT	GPT-3.5-turbo	CodeT5	SFT	May-2023
	AICD [307]	ICL	GPT-3.5-turbo	GPT-J	SFT	Jun-2023
	KPTD [308]	NLU	GPT-3.5	GPT-2, LLaMA	SFT	Jun-2023
	DRA [309]	CoT	ChatGPT	GPT-J	SFT	Oct-2023
	TDIG [231]	CoT	GPT-3.5-turbo, GPT-4	LLaMA	SFT	Dec-2023
	Selective Reflection-Tuning [310]	IF	ChatGPT	LLaMA	SFT	Feb-2024
	DEBATunE [311]	IF/TP	ChatGPT	LLaMA	SFT + RL	Feb-2024
	DeepSeek-R1 [28]	TP	DeepSeek-R1, DeepSeek-V3	LLaMA, Qwen	SFT	Jan-2025
	DynaBERT [312]	NLU	BERT, RoBERTa	DynaBERT, DynaRoBERTa	SFT	Apr-2020
	TED [313]	NLU	GPT-2	GPT-2	D&S + SFT	Oct-2022
	GKD [314]	NLG/NLU/IF	T5-XL	T5	D&S + RL	Jun-2023
	MINILLM [315]	IF	GPT-2, OPT, LLaMA	GPT-2, OPT, LLaMA	D&S	Jun-2023
	RICD [316]	IF	LLaMA	LLaMA	SFT + RL	Jul-2023
White-box KD	BabyLlama [317]	IF	GPT-2, LLaMA	LLaMA	D&S	Aug-2023
	MiniMoE [255]	NLU/TP	GPT-2, Pythia	GPT-GPT-2, Pythia	SFT	Nov-2023
	Genie [318]	NLG	Falcon, LLaMA	FLAN, LLaMA	SFT	Jan-2024
	Self-Rewarding [111]	IF	LLaMA	LLaMA	SFT + RL	Jan-2024
	DistiLLM [319]	IF/NLG	GPT-2, OPT, OpenLLaMA	GPT-2, OPT, OpenLLaMA	D&S + RL	Feb-2024
	AMMD [320]	IF	OpenLLaMA	OpenLLaMA	D&S + RL	Jun-2024

internal parameters and intermediate representations. As shown in **Tab. 6**, knowledge distillation methods can be broadly categorized into two types: Black-box KD and White-box KD. We provide a systematic summary of various knowledge distillation techniques in large language models (LLMs), along with their corresponding skills, teacher models, and student models.

Black-box KD. Black-box KD refers to a scenario in which the student model learns solely from the teacher's output logits, without access to its internal representations or architectural details. This approach, originally proposed by Hinton [321], aligns with the classical KD paradigm and is widely adopted due to its flexibility. A key advantage of black-box KD is that it treats the teacher model as an opaque function, enabling knowledge transfer even when the teacher is a proprietary or pre-trained model with restricted access. In practice, large teacher LLMs (e.g., ChatGPT and GPT-4 [9]) are commonly employed to generate high-quality outputs. Meanwhile, smaller language models (SLMs), including GPT-2 [14], T5 [322], Flan-T5 [323], and CodeT5 [324], serve as student models. These SLMs are optimized for efficiency while maintaining strong generalization capabilities, making them suitable for deployment in resource-constrained environments.

White-box KD. White-box KD extends the traditional distillation paradigm by leveraging additional insights from the teacher's internal representations. This approach is beneficial when the architecture of the teacher model is known and accessible, allowing for richer forms of supervision. Unlike black-box KD, which treats the teacher as an opaque function, white-box KD allows the student model to learn not only

from the teacher's output logits but also from its intermediate activations, hidden layers, and potentially even attention weights [325].

DeepSeek-R1: Direct Distillation of Reasoning Patterns. DeepSeek-R1 exemplifies the transformative potential of KD by distilling intricate reasoning patterns from large-scale models into compact architectures, significantly enhancing the reasoning capabilities of smaller LLMs without the computational burden of direct RL on such models. This approach, termed direct distillation, leverages a curated dataset of approximately 800k samples generated by a large teacher model, comprising 200k non-reasoning instances derived from DeepSeek-V3 and 600k reasoned instances produced by the DeepSeek-R1-Stage1 checkpoint. These samples form the foundation for SFT applied to open-source base models, such as Qwen and LLaMA mini-variants, enabling the student models to inherit sophisticated reasoning faculties typically reserved for their larger counterparts.
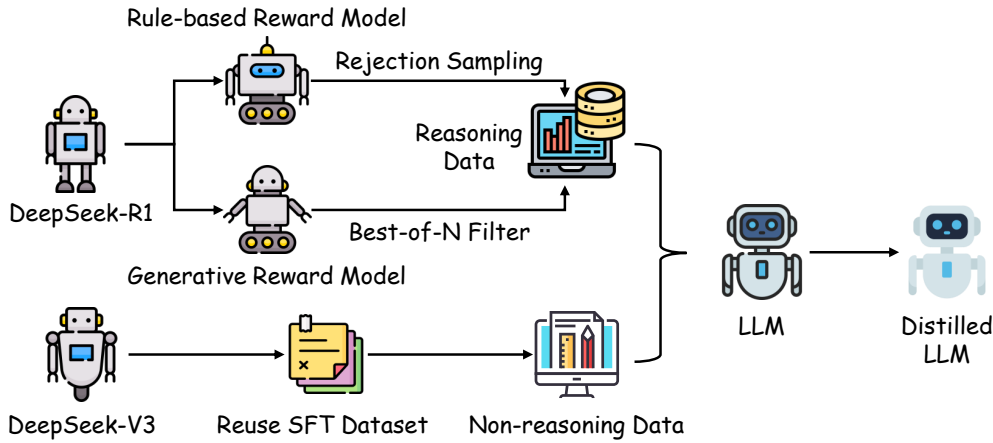

Figure 17: Workflow of knowledge distillation in DeepSeek-R1, illustrating the process of transferring reasoning patterns from large to compact models.

The direct distillation process in DeepSeek-R1 unfolds in a structured pipeline, as depicted in **Fig. 17**. Initially, the teacher model—pre-trained on extensive datasets—generates a diverse corpus encompassing both reasoning and non-reasoning outputs, capturing a spectrum of logical patterns and factual knowledge. The non-reasoning data (about 200k samples) provides a baseline of general knowledge, while the reasoning data (about 600k samples) encapsulates multi-step reasoning chains, refined through the teacher's advanced capabilities. This dataset is then employed in an SFT phase, where the student model is trained to align its output distribution with that of the teacher, using reasoning data for direct fine-tuning of the smaller model to distill a compact inference model. Unlike traditional RL applied directly to small models, which may yield suboptimal reasoning due to limited capacity, DeepSeek-R1's direct distillation circumvents such constraints by transferring pre-optimized reasoning behaviors, achieving superior performance with reduced resource demands.

A distinguishing feature of DeepSeek-R1's KD methodology is its emphasis on preserving reasoning integrity across model scales. By integrating reasoned trajectories from DeepSeek-R1-Stage1—a checkpoint refined through large-scale RL—the student models not only replicate factual accuracy but also emulate complex inference processes, such as those required for mathematical problem-solving or logical deduction. This targeted transfer contrasts with conventional KD, which often prioritizes classification tasks, and underscores DeepSeek-R1's innovation in reasoning-oriented distillation. Furthermore, the approach minimizes the need for extensive RL iterations on the student, leveraging the teacher's pre-computed reasoning outputs to streamline training, thus enhancing efficiency and scalability. This methodology positions DeepSeek-R1 as a paradigm for distilling advanced reasoning into compact LLMs, offering a blueprint for future post-training optimization efforts.

7 PoLMs for Integration and Adaptation

Integration and adaptation techniques are pivotal for enhancing the versatility and efficacy of LLMs across diverse real-world applications. These methodologies enable LLMs to seamlessly process heterogeneous data types, adapt to specialized domains, and leverage multiple architectural strengths, thereby addressing complex, multifaceted challenges. This chapter delineates three principal strategies: **Multi-modal Integration** (§7.1), which equips models to handle diverse data modalities such as text, images, and audio; **Domain Adaptation** (§7.2), which refines models for specific industries or use cases; and **Model Merging** (§7.3), which amalgamates capabilities from distinct models to optimize overall performance. Collectively, these approaches enhance LLMs' adaptability, efficiency, and robustness, broadening their applicability across varied tasks and contexts.
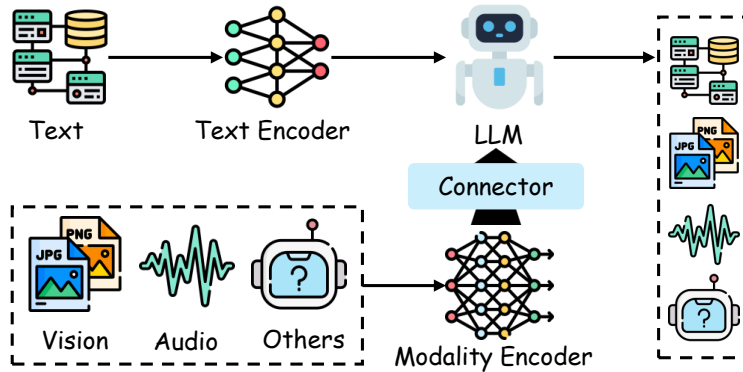

Figure 18: Taxonomy of modal connection methods in multi-modal integration, delineating projection-based, query-based, and fusion-based approaches.

7.1 Multi-Modal Integration

Building upon the post-training optimization strategies elucidated in preceding chapters, this section examines advanced methodologies designed to augment LLMs and Large Multi-modal Models (LMMs) for effective processing of multi-modal data. While supervised fine-tuning enhances LLMs' proficiency in task-specific contexts, its limitations in harnessing the full spectrum of multi-modal capabilities necessitate more sophisticated post-training approaches. These techniques enable LMMs to address complex, cross-modal tasks (e.g., generating web page code from visual inputs [326], interpreting nuanced cultural artifacts like memes [327], and performing mathematical reasoning without reliance on optical character recognition [50]), by integrating diverse data types into a unified framework. Typically, LMMs comprise a modal encoder, a pre-trained LLM backbone, and a modal connector [328], as depicted in **Fig. 18**. This architecture forms the foundation for post-training methods that refine each component, facilitating robust multi-modal integration and performance enhancement.

7.1.1 Modal Connection

Modal connection methods are pivotal in synthesizing multi-modal data into a coherent representational framework, categorized into three primary strategies: projection-based, query-based, and fusion-based approaches [328], as outlined in **Fig. 19**.

Projection-based Modal Connection. Projection-based methods transform diverse modal inputs into a unified text embedding space, aligning their features with the linguistic dimensions of LLMs for seamless integration. LLaMA-Adapter [329] exemplifies this approach by incorporating an image encoder to extend LLMs into multi-modal systems, enabling image-conditioned instruction tracking. Its successor, LLaMA-Adapter V2 [330], refines this process by embedding visual tags into early LLM layers, fostering improved assimilation of visual knowledge. FROMAGe [331] employs fine-tuning of input and output layers within a frozen LLM and visual encoder framework to enable cross-modal interactions, while LLaVA-1.5 [332] utilizes a
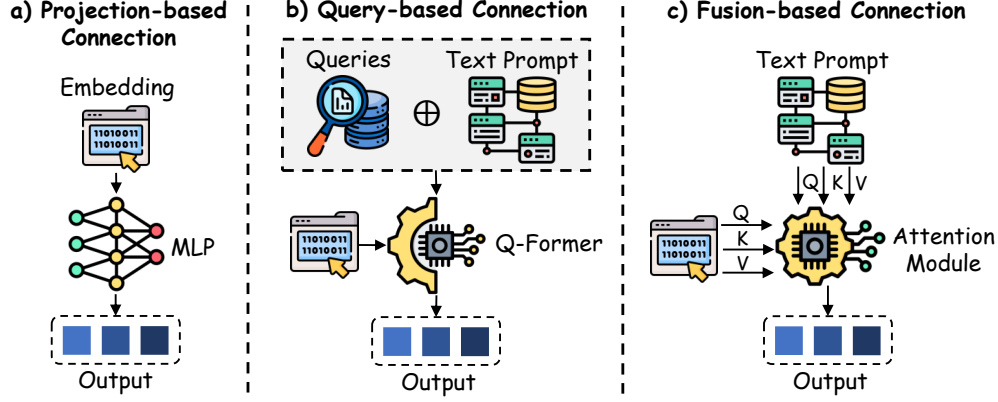
Figure 19: Taxonomy of modal connection methods in multi-modal integration, delineating projection-based, query-based, and fusion-based approaches.

bilinear multilayer perceptron (MLP) to bolster robustness in multi-modal processing. Recent developments, such as Shikra [333], integrate spatial coordinates to enhance natural language dialogues, and VILA [334] optimizes visual-language pre-training for superior zero-shot capabilities. DetGPT [335] further advances this paradigm by merging reasoning-driven object detection with natural language interaction, leveraging projection techniques to facilitate effective multi-modal communication. SOLO [336] employs a single Transformer architecture for unified and end-to-end vision-language modeling, by accepting both raw image patches (in pixels) and texts as inputs, without using a separate pre-trained vision encoder. Meanwhile, MiniGPT-4 [326] aligns a frozen visual encoder with Vicuna using a single projection layer, achieving GPT-4-like abilities with a two-stage training process. Idefics [337] excels with an autoregressive design and multi-stage pretraining for efficient inference. LaVIT [338] unifies vision and language with a discrete visual tokenizer for seamless generation. DeepSeek-VL2 [339] enhances high-resolution image understanding with dynamic tiling and multi-head latent attention. Finally, Qwen2.5-VL [340] advances multi-modal tasks with a redesigned Vision Transformer, excelling in perception and video comprehension.

Query-based Modal Connection. Query-based methods enhance multi-modal integration by employing learnable query tokens to extract structured information from diverse modalities, bridging the gap between textual and non-textual data. BLIP-2 [52] pioneered this approach with query transformers, integrating text and visual inputs efficiently. Video-LLaMA [341] extends this technique to video comprehension through combined visual encoders, while InstructBLIP [342] refines query mechanisms to ensure precise adherence to instructions. X-LLM [343] aligns multi-modal inputs via specialized interfaces, and subsequent innovations like mPLUG-Owl [344] and Qwen-VL [345] optimize the Q-Former architecture for computational efficiency. LION [346] further demonstrates the efficacy of query-based methods by advancing visual knowledge integration, underscoring their utility in enhancing LMM performance across varied tasks. Qwen-VL [345] is a series of large-scale vision-language models built upon Qwen-7B, incorporating a visual receptor, a position-aware adapter, and a three-stage training pipeline to enable multilingual, fine-grained vision-language understanding. Lyrics [347] is a fine-grained vision-language pre-training and instruction fine-tuning framework that enhances large vision-language models (LVLMs) by integrating semantic-aware visual objects through a visual refiner (image tagging, object detection, and semantic segmentation) and a Multi-scale Querying Transformer (MQ-Former).

Fusion-based Modal Connection. Fusion-based techniques deepen cross-modal interactions by embedding multi-modal features directly into the LLM architecture, fostering richer integration at the inference level. Flamingo [51] employs cross-attention layers to fuse visual features during token prediction, enabling dynamic multi-modal processing. OpenFlamingo [348] builds on this by allowing frozen LLMs to attend to vision encoder outputs, enhancing flexibility. Otter [349] introduces instruction tuning to improve multi-modal instruction-following, while CogVLM [350] integrates visual expert modules within Transformer layers for seamless feature synthesis. Obelics [351] leverages interleaved image-text training data, highlighting the robustness of fusion-based methods in achieving cohesive multi-modal performance. InternVL [352] is

a large-scale vision-language foundation model that scales up the vision encoder to 6 billion parameters and progressively aligns it with LLMs using a language middleware (QLLaMA). Llama 3 [25] is a new family of multilingual, tool-using foundation models developed by Meta, scaling up to 405B parameters with a 128K token context window, optimized through improved data quality, larger-scale training, and a structured post-training strategy.

Table 7: Overview of Encoders and Large Multi-modal Models Across Modalities (2022–2025). This table summarizes key multi-modal models, detailing their encoder categories, sizes, input projectors, LLM backbones, and release timelines across vision, audio, and other modalities. Metrics include **C-a** (Cross-attention), **Q-F** (Q-Former), **MQ-F** (Multi-Query Q-Former), and **LP** (Linear Projector), representing input projection mechanisms.

Modal	Model	Encoder		Input Projector	LLM Backbone	Release Time
		Category	Size			
Vision	Flamingo [51]	NFNet	0.3B	C-a	Chinchilla-1.4B/7B/70B	Apr-2022
	BLIP-2 [52]	CLIP/Eva	0.3B/1.5B	Q-F/LP	Flan-T5/OPT	Jan-2023
	MiniGPT-4 [326]	Eva	1.5B	Q-F/LP	Vicuna-13B	Apr-2023
	VideoChat [353]	Eva	1.5B	Q-F/LP	StableVicuna-13B	May-2023
	InstructBLIP [342]	ViT	1.5B	Q-F/LP	Flan-T5/Vicuna	May-2023
	Video-ChatGPT [354]	CLIP	0.3B	LP	Vicuna-v1.1	May-2023
	IDEFICS [337]	OpenCLIP	0.3B	C-a	LLaMA	May-2023
	Qwen-VL-(Chat) [345]	OpenCLIP	1.8B	C-a	Qwen-7B	Aug-2023
	LLaVA [53]	CLIP	0.3B	LP	Vicuna-13B	Oct-2023
	Lyrics [347]	CLIP	0.1B~0.3B	MQ-F/LP	Vicuna-13B	Oct-2023
	CogVLM [350]	Eva	1.5B	MLP	Vicuna-v1.5-7B	Nov-2023
	BT-Adapter [355]	CLIP	0.3B	LP	Vicuna-7B	Nov-2023
	SPHINX-Tiny [356]	DINOv2	1.0B	LP	TinyLlama-1.1B	Jan-2024
	VL-Mamba [357]	SigLIP	0.4B	MLP	Mamba-2.8B-Slimpj	Feb-2024
	Mipha [358]	SigLIP	0.4B	MLP	Phi-2-2.7B	Mar-2024
	IntrenVL [352]	InternImage	6.0B	C-a/MLP	Qwen-Llama-8B	Mar-2024
	Cobra [359]	SigLIP/DINOv2	0.4B	MLP	Mamba-2.8B-Zephyr	Apr-2024
	LaVIT [338]	ViT	0.1B~0.3B	C-a	LLaMA-7B	May-2024
	DeepSeekVL2 [339]	Dynamic Tiling	2.8B	MLP	DeepSeek-MoE-16B	May-2024
	VILA [334]	ViT	0.1B~0.3B	LP	LLaMA-2-7B/13B	Jun-2024
	Llama 3 [25]	SigLIP	1.5B	LP	LLaMA-3	Jul-2024
	QVQ [360]	ViT	0.6B	C-a	Qwen2-VL-72B	Dec-2024
	Qwen2.5-VL [340]	ViT	0.8B	MLP	Qwen-MoE-A2.7B	Feb-2025
	Claude3.7 [361]	-	-	-	Claude3.7	Feb-2025
	GPT-4.5 [362]	-	-	-	GPT-4.5	Feb-2025
Audio	AudioPaLM [363]	AudioMAE	0.25B	LP	PaLM-2-8B	Sep-2023
	Qwen-Audio [345]	Whisper-L-v2	1.5B	LP	Qwen-7B	Nov-2023
	SpeechGPT [364]	Transformer	0.1B~0.3B	LP	LLaMA-13B	Dec-2023
	VoiceCraft [365]	Neural-Codec	0.33B/0.83B	LP	Mamba-2.8B	Mar-2024
Others	X-LLM [343]	Transformer	0.1B~0.3B	Q-F/LP	ChatGLM-6B	May-2023
	Video-LLaMA [341]	Eva-CLIP	1.5B/0.17B	Q-F/LP	Vicuna/LLaMA	Jun-2023
	ImageBind-LLM [366]	ImageBind	0.17B	LP	LLaMA-7B	Jun-2023
	AnyMAL [367]	CLIP/CLAP	0.3B	C-a/LP	LLaMA-2	Sep-2023
	NEXT-GPT [368]	ImageBind	0.17B	LP	Vicuna-7B	Oct-2023
	CoDi-2 [369]	ImageBind	0.17B	MLP	LLaMA-2-Chat-7B	Jan-2024
	LL3DA [370]	Transformer	0.1B~0.3B	LP	OPT-1.3B	Feb-2024
	X-InstructBLIP [371]	Eva/BEATs/ULIP-2	1.5B	Q-F/LP	Vicuna-v1.1-7B/13B	Dec-2024

7.1.2 Modal Encoder

Modal encoders compress raw multi-modal inputs into compact, semantically rich representations, enabling efficient processing across diverse tasks and modalities. These components are essential for translating hetero-

geneous data into a format compatible with LLM backbones, supporting applications from visual reasoning to audio comprehension. **Tab. 7** provides a comprehensive summary of prevalent encoders utilized in vision, audio, and other modalities, detailing their characteristics and contributions to multi-modal integration.

Vision Encoder. Vision encoders are foundational to multi-modal learning, facilitating the interpretation and generation of visual data within LMMs. CLIP [372] establishes joint image-text representations through contrastive learning, enhancing cross-modal alignment. EVA [373] refines visual attention mechanisms to improve efficiency, while ImageBind [374] creates a unified embedding space across multiple modalities, elevating zero-shot recognition capabilities. SigLIP [375] introduces a paired sigmoid loss to optimize image-text pre-training, and DINOv2 [376] employs unsupervised learning to derive robust visual features from diverse sources. LLaVA [53] adapts self-instruct strategies to convert images into textual descriptions, generating novel datasets with advanced LLMs. Video-ChatGPT [354] supports conversational video understanding with large-scale instruction datasets, while BT-Adapter [355] optimizes video comprehension through efficient temporal modeling. VideoChat [353] focuses on spatiotemporal reasoning, leveraging specialized datasets, and models like CoDi-2 [369] and Mipha [358] achieve efficiency gains in multi-modal processing. VL-Mamba [357] and Cobra [359] introduce state-space models for optimized inference, and SPHINX-Tiny [356] emphasizes data diversity and training efficiency.

Audio Encoder. Audio encoders enhance LMMs' capacity to process and interpret auditory inputs, broadening their multi-modal scope. SpeechGPT [364] integrates large-scale speech datasets with convolutional and transformer architectures [377] to achieve robust instruction-following capabilities. AudioPaLM [363] combines text and speech processing using the Universal Speech Model (USM) encoder [378], excelling in tasks like zero-shot language translation. WavCaps [379] employs CNN14 [380] and HTSAT [381] to mitigate audio-language data scarcity, utilizing advanced LLMs to refine dataset quality and bolster learning outcomes, underscoring the critical role of audio modalities in multi-modal systems.

Other Encoder. Beyond vision and audio, encoders for additional modalities, such as 3D understanding and multi-modal fusion, are integral to comprehensive LMMs. NEXT-GPT [368] facilitates cross-modal content generation across text, images, video, and audio, advancing human-like AI capabilities with minimal parameter adjustments. ImageBind-LLM [366] aligns visual and linguistic embeddings to improve instruction-following across modalities. LL3DA [370] processes point cloud data for 3D reasoning and planning, introducing novel approaches to spatial understanding. X-LLM [343] employs Q-Former [52] for image and video inputs and C-Former [343] for speech, compressing audio features into token-level embeddings to enhance multi-modal learning efficiency.

7.2 Domain Adaptation

Domain Adaptation (DA) constitutes a pivotal post-training strategy for refining LLMs to excel within specialized domains, ensuring their efficacy in targeted applications. Rooted in the principles of transfer learning [382, 383], DA transforms an initial model, denoted M_{source}, through an adaptation function F_{adapt} to produce a domain-specific model M_{target}, as depicted:

$$\underbrace{M_{target}}_{\text{Target Model}} = \underbrace{F_{adapt}}_{\text{Adapt Method}}(\underbrace{M_{source}}_{\text{Source Model}}) \tag{31}$$

This process tailors M_{target} to address the unique demands and intricacies of a designated domain, thereby optimizing its performance and relevance. By enhancing LLMs' proficiency in fields such as programming [384, 385] and mathematical reasoning [386], DA not only elevates domain-specific capabilities but also improves computational efficiency, mitigating the limitations of general-purpose models that often struggle with domain-specific terminologies and reasoning paradigms. Furthermore, DA substantially reduces the reliance on extensive labeled datasets and computational resources typically required for training domain-specialized models from scratch [387], rendering it a cornerstone of post-training methodologies.

7.2.1 Knowledge Editing

Knowledge Editing represents a sophisticated post-training approach aimed at modifying LLMs to meet domain-specific requirements without compromising their foundational capabilities. This technique facilitates targeted parameter adjustments, preserving the model's pre-existing performance while integrating new or updated domain knowledge [388]. By enabling rapid adaptation to evolving knowledge landscapes, Knowledge Editing emerges as an indispensable component of post-training pipelines. An overview of principal methods (e.g., encompassing external knowledge utilization, integration, and intrinsic editing) is presented in **Tab. 8**.

Table 8: A comparative analysis of representative approaches for knowledge editing in LLMs. **Edit Area** specifies the components of the model that are targeted for modification; **Editor #Params** indicates the parameters that require updating during the editing process. L represents the number of layers subjected to modification, d_h denotes the dimensionality of the hidden layers within the transformer architecture, d_m refers to the intermediate dimension that exists between the up-projection and down-projection phases, and N symbolizes the total number of neurons that undergo updates within each individual layer.

Category	Method	Edit Area	Edit Function	Edited #Params
Identification	SERAC [389]	memory+classifier+auxiliary model	$\text{Output} \rightarrow \text{Model}_{cf}(\boldsymbol{x})$	-
	PokeMQA [390]	memory+retriever	$\text{Input} \rightarrow [\text{Mem} : \text{Input}]$	-
Association	CaliNET [391]	FFN+params	$\boldsymbol{h} \rightarrow \boldsymbol{h} + \text{FFN}_{\text{add}}(\boldsymbol{x})$	$N \times d_h$
	Transformer-Patcher[392]	FFN+params	$\boldsymbol{h} \rightarrow \boldsymbol{h} + \text{FFN}_{\text{add}}(\boldsymbol{x})$	$N \times d_h$
	REMEDI [393]	auxiliary model	$\boldsymbol{h} \rightarrow \text{REMEDI}(\boldsymbol{x})$	$d_h \times d_h$
	GRACE [394]	FFN+codebook	$\boldsymbol{h} \rightarrow \text{GRACE}(\boldsymbol{x})$	$N \times 2d_h$
	Eva-KELLM [395]	Attn or FFN	$\boldsymbol{h} \rightarrow \boldsymbol{h} + s \cdot \text{LoRA}(\boldsymbol{x})$	$2L \times 2d_{am}d_h$
	MELO [396]	Attn or FFN	$\boldsymbol{h} \rightarrow \boldsymbol{h} + s \cdot \text{LoRA}(\boldsymbol{x})$	$2L \times 2d_{am}d_h$
Editing	Constrained Fine-tuning [397]	Any	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$2 \times L \times d_m d_h$
	Editable Training [398]	Any	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$2 \times L \times d_m d_h$
	KnowledgeEditor[399]	Attn or FFN + auxiliary model	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$2 \times L \times d_m d_h$
	SLAG [400]	Attn or FFN + auxiliary model	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$2 \times L \times d_m d_h$
	MEND [401]	FFN + auxiliary model	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$2 \times L \times d_m d_h$
	MALMEN [402]	FFN	$\boldsymbol{W}_{\text{down}} \rightarrow \boldsymbol{W}'_{\text{down}}$	$L \times d_m d_h$
	LLM Surgery [403]	Any	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$2 \times L \times d_m d_h$
	KNE [404]	subset of parameters	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$L \times d_m d_h$
	OVERTONE [405]	Any	$\boldsymbol{W} \rightarrow \boldsymbol{W}'$	$2 \times L \times d_m d_h$

Formal Definition of Knowledge Editing. Consider an original LLM parameterized by θ, pre-trained on a dataset $\mathcal{D}_{\text{old}}$. Let $\mathcal{D}_{\text{new}}$ denote a dataset containing novel or updated information ΔK. The objective of Knowledge Editing is to derive a revised parameter set θ' by applying an adjustment $\Delta\theta$, effectively assimilating ΔK while minimizing degradation on $\mathcal{D}_{\text{old}}$. Formally, this is framed as a constrained optimization problem, where the updated parameters are defined as:

$$\theta' = \theta + \Delta\theta, \text{where } \mathcal{L}(\theta'; \mathcal{D}_{\text{new}}) \rightarrow \min, \tag{32}$$

with $\mathcal{L}$ representing a loss function (e.g., cross-entropy) assessing model quality on $\mathcal{D}_{\text{new}}$. To safeguard performance on the original dataset, a constraint is imposed:

$$\mathcal{L}(\theta'; \mathcal{D}_{\text{old}}) \leq \mathcal{L}(\theta; \mathcal{D}_{\text{old}}) + \epsilon, \tag{33}$$

where ϵ is a small positive constant limiting performance loss on $\mathcal{D}_{\text{old}}$. This formulation ensures that θ' incorporates ΔK while retaining the model's prior knowledge base. Practically, $\Delta\theta$ may be constrained to specific architectural components (e.g., attention layers (Attn) or feed-forward networks (FFN)), reducing computational overhead and preserving core functionality by avoiding comprehensive retraining.

43

Knowledge Identification. The initial phase of Knowledge Editing focuses on detecting and assimilating new information into the model. PokeMQA [390] employs a programmable scope detector and knowledge prompts to dissect queries, retrieving pertinent facts efficiently. Conversely, SERAC [389] integrates a counterfact model with a classifier to determine the applicability of new knowledge sources, offering a minimally invasive approach that preserves the base model's integrity without necessitating extensive structural modifications. [406] analyzes the reasons why LLM knowledge updating creates messy ripple effect. Real world edits usually originate from emerging events that encompass logical connections between new facts and past facts, based on this observation, EvEdit [407] proposes an event-based knowledge editing method to determine knowledge anchor and knowledge updating boundary.

Knowledge Association. Following identification, this phase associates newly acquired information with the model's existing knowledge framework. Transformer-Patcher [392] adapts transformer architectures to integrate updated facts, while CaliNET [391] recalibrates parameters to align with factual content. Methods such as Eva-KELLM [395], MELO [396], and REMEDI [393] refine specific behaviors for precise updates, and GRACE [394] enhances predictive accuracy post-knowledge insertion, ensuring seamless integration with prior representations.

Intrinsic Knowledge Editing. The final stage embeds associated facts into the model's internal structure, ensuring comprehensive assimilation. While traditional fine-tuning can be resource-intensive, advanced techniques mitigate this burden. Constrained Fine-tuning [397] and Meta-learning [399] minimize knowledge loss and overfitting risks. Editable Training [398] and KnowledgeEditor [399] enable swift parameter adjustments with minimal performance impact, whereas SLAG [400], MEND [401], and MALMEN [402] resolve edit conflicts and support large-scale updates, maintaining foundational competencies while incorporating new domain insights. LLM Surgery [403] unifies unlearning and editing by applying reverse gradients to remove outdated data, gradient descent to integrate new facts, and a KL divergence term to preserve existing knowledge, achieving significant computational efficiency. KNE [404] introduces a Knowledge Neuronal Ensemble method that pinpoints and updates only those neurons strongly associated with newly inserted facts, achieving more accurate edits while preserving unrelated knowledge. OVERTONE [405] tackles heterogeneous token overfitting in knowledge editing by introducing a token-level smoothing technique that adaptively refines the training objective, thereby preserving pre-trained knowledge and improving the model's ability to reason about newly inserted facts. These targeted techniques ensure that the model retains its foundational competencies while integrating newly acquired information.

7.2.2 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) integrates traditional information retrieval with contemporary LLMs to enhance the relevance and factual accuracy of generated outputs [48, 408, 409]. By dynamically retrieving pertinent information from external sources and embedding it into the generation process, RAG addresses deficiencies in LLMs' domain-specific knowledge and reduces the propensity for hallucinated content. This approach proves particularly effective in domains requiring precise, up-to-date information, such as question-answering systems [48], scientific research [410], and healthcare [411], where it adeptly handles complex queries and knowledge-intensive tasks. Moreover, RAG mitigates the prevalence of misleading responses in conversational systems, advancing the fidelity of knowledge-driven natural language generation [411, 412].

This subsection focuses on training-based RAG methodologies [413], recognizing that training-free RAG approaches [414, 415, 416] may compromise knowledge utilization efficiency due to the absence of task-specific optimization. Three predominant training strategies—Independent Training, Sequential Training, and Joint Training—enhance model adaptability and integration proficiency, as illustrated in **Fig. 20**.

Independent Training. This strategy trains the retriever and generator as distinct modules, enabling flexibility in employing sparse or dense retrievers tailored to task demands. DPR [417], for instance, utilizes dual BERT networks to encode queries and paragraphs separately, applying contrastive learning to optimize retrieval without generator interaction. Likewise, [418] propose Reward-RAG, which leverages a reward model to fine-tune only the retriever according to GPT-based feedback, leaving the generator untouched.

Sequential Training. Sequential Training enhances efficiency by optimizing one module at a time, promoting synergy between the retriever and generator. It includes Retriever-First approaches [419, 420, 421, 422, 423],
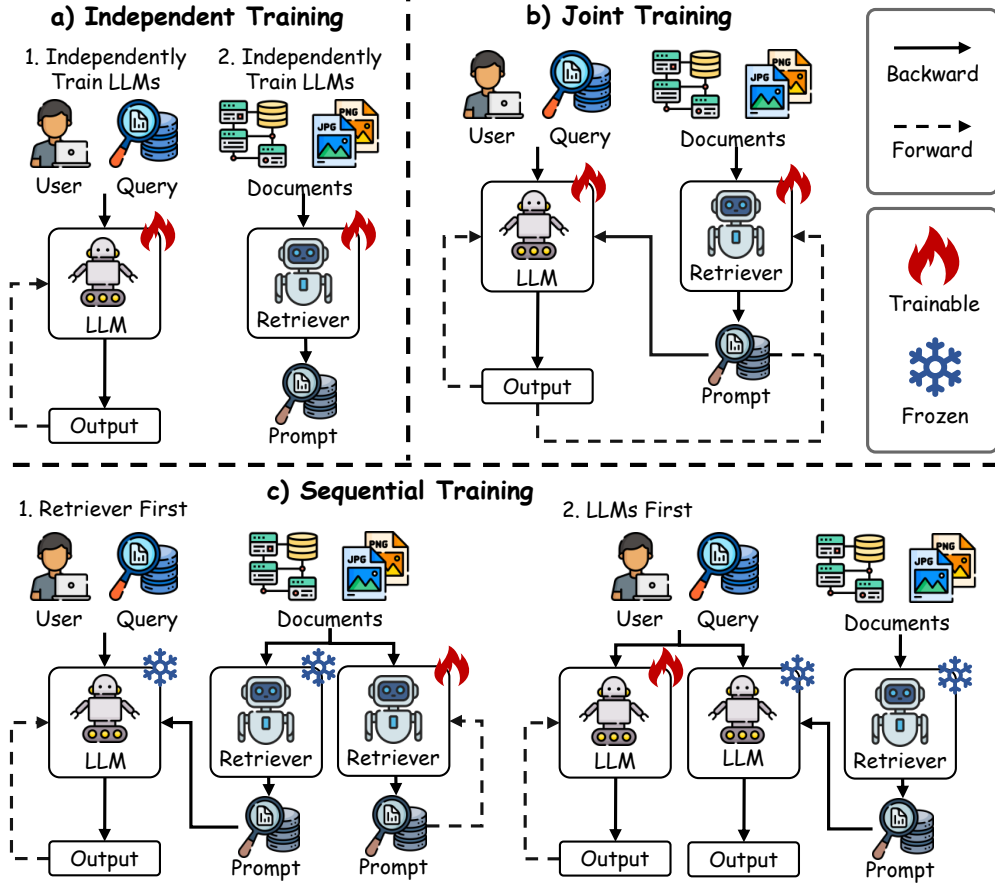
Figure 20: Taxonomy of Retrieval-Augmented Generation (RAG) training methods, categorizing Independent Training, Sequential Training, and Joint Training strategies.

like RETRO [424], which pre-trains a BERT-based retriever before training an encoder-decoder to seamlessly integrate retrieved content for improved performance. Alternatively, LLM-First methods [425, 426, 427], such as RA-DIT [428], first fine-tune the language model to effectively utilize retrieved knowledge, then refine the retriever for better alignment and coherence [419, 425].

Joint Training. Joint Training synchronizes retriever and generator optimization in an end-to-end framework. RAG [48] minimizes negative log-likelihood to co-train both components, while REALM [429] enhances retrieval precision with Maximum Inner Product Search (MIPS) [430]. These methods adapt to task-specific needs, maximizing external knowledge benefits and minimizing generation errors.

7.3 Model Merging

Model merging has emerged as a vital post-training strategy for enhancing the performance and efficiency of LLMs across both training and inference phases [431, 432]. This approach consolidates specialized models into a unified architecture, circumventing the need for extensive retraining and addressing challenges posed by large model sizes and computational demands. Unlike training on amalgamated datasets, model merging integrates single-task models into a cohesive entity capable of multi-task proficiency, offering a resource-efficient paradigm for multi-task learning. By streamlining the training pipeline and fostering the development of versatile models with robust generalization across applications, this technique optimizes LLM deployment in diverse contexts. Given a set of candidate models $M = \{M_1, M_2, \ldots, M_n\}$, the objective is to devise a merging function F_{merge} that yields a unified model M', potentially anchored by a base model M_1, as depicted:

$$M' = F_{merge} (M_1, M_2, \cdots, M_n) \tag{34}$$

where the annotations indicate: Merged model (M'), Merge Method (F_{merge}), Candidate Models and Base model ($M_1, M_2, \cdots, M_n$).

7.3.1 Model Merging at Hierarchical Levels

Model merging techniques are systematically categorized into three hierarchical levels—weight-level, output-level, and model-level merging—as illustrated in **Fig. 21**.
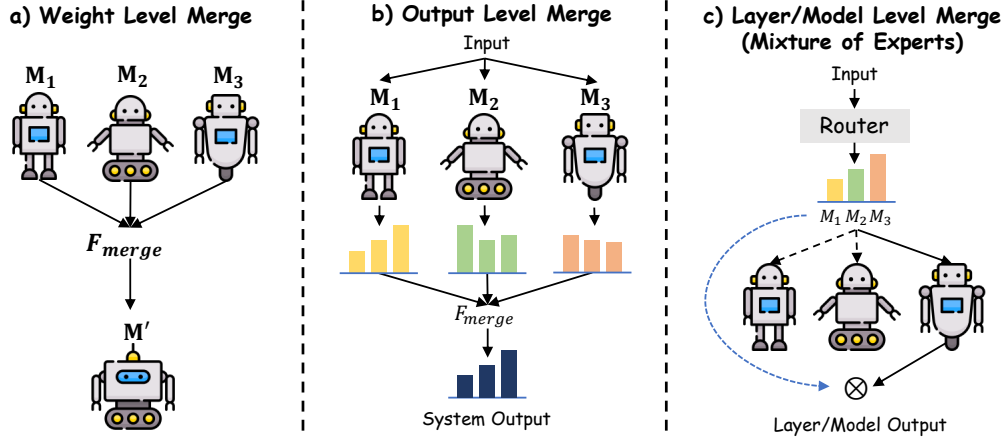

Figure 21: Taxonomy of model merging techniques, delineating hierarchical levels including weight-level, output-level, and model-level approaches for Large Language Models.

Weight-Level Model Merging. Weight-level merging directly manipulates the parameter space, making it particularly effective for models sharing architectural similarities or trained on related tasks. Formally, given parameter sets $\theta_1, \theta_2, \ldots, \theta_n \in \mathbb{R}^d$, a linear merging scheme aggregates these into a unified set θ' as:

$$\theta' = \alpha_1 \theta_1 + \alpha_2 \theta_2 + \ldots + \alpha_n \theta_n, \quad \text{subject to} \quad \alpha_k \geq 0, \sum_{k=1}^{n} \alpha_k = 1. \tag{35}$$

Model Soup [433, 434] exemplifies this by linearly combining weights from models fine-tuned on diverse tasks, yielding a single, efficient model. Task Arithmetic (TA) [435] extends this flexibility through arithmetic operations on parameters, enhancing performance adaptability. To mitigate alignment issues, TIES-merging [436] ensures parameter congruence, while DARE [437] minimizes interference by probabilistically adjusting parameter deltas, optimizing the merging process for coherence and efficiency.

Output-Level Model Merging. Output-level merging becomes advantageous when models diverge in architecture or initialization, rendering weight-level methods impractical. This approach aggregates output distributions rather than internal parameters, formulated as:

$$y' = \alpha y_1 + (1 - \alpha) y_2, \quad \alpha \in [0, 1], \tag{36}$$

where y_1 and y_2 represent probability distributions from models M_1 and M_2, respectively.akin to ensemble strategies, this method synthesizes model predictions into a unified output. LLMBlender [438] implements this by generating independent outputs and fusing them via ranking and generative processes, while FuseLLM [439] distills combined output probabilities into a single network for distributional fidelity. FuseChat [440] bridges weight- and output-level merging by transferring knowledge from multiple LLMs into a consolidated target, enhancing cross-model synergy.

Model-Level Model Merging. Model-level merging integrates submodels or layers through routing mechanisms, often within mixture-of-experts (MoE) frameworks, expressed as:

$$M' = \mathrm{Merge}(M_1, M_2), \tag{37}$$

where Merge denotes either hard or soft routing functions. The Switch Transformer [54] employs discrete gating to activate expert layers selectively, reducing computational load albeit with potential performance trade-offs due to rigid routing. SoftMoE [441] and SMEAR [442] utilize continuous gating to facilitate smoother transitions between experts, enhancing component integration and model cohesion.

7.3.2 Pre-Merging Methods

Pre-merging methods establish a compatibility foundation for model merging by optimizing the weight space, architectural coherence, and parameter alignment of independent models, thereby minimizing conflicts and interference in subsequent fusion stages. These techniques enhance the efficacy of merging processes, ensuring that the resulting unified model retains the strengths of its constituents while mitigating potential degradation.

Linearization Fine-tuning. This approach refines models within the tangent space of a pre-trained model, eschewing the original nonlinear parameter space to achieve weight disentanglement, which reduces interference during merging. Techniques such as partial linearization of adapters (e.g., TAFT [443]) or attention layers [444] align weight updates to disjoint input regions, preserving independent functionalities in the merged model [445]. By constraining updates to a linearized framework, this method facilitates seamless integration across diverse models.

Architecture Transformation. This strategy converts heterogeneous models with divergent architectures into a homogeneous form amenable to direct parameter merging. Approaches include knowledge distillation, as exemplified by FuseChat [440], and identity layer insertion, such as CLAFusion [446]. GAN Cocktail [447] initializes target models to assimilate outputs from varied architectures, enabling a unified merging process that bridges structural disparities effectively.

Weight Alignment. This method aligns models to a shared weight basin through permutation, capitalizing on the Linear Mode Connectivity (LMC) property to enhance compatibility. Techniques encompass optimal transport (OTFusion [448]), heuristic matching (Git re-basin [449]), and learning-based alignment (DeepAlign [450]). REPAIR [451] mitigates alignment failures in models lacking normalization layers, ensuring robust parameter convergence prior to fusion.

7.3.3 During-Merging Methods

During-merging methods focus on dynamically optimizing parameter fusion strategies to resolve task conflicts, mitigate interference, and elevate the performance and generalization capacity of the resultant merged model. These approaches address the challenges of integrating disparate models in real-time, enhancing the adaptability and robustness of the unified architecture.

Basic Merging. This method leverages straightforward parameter averaging or task vector arithmetic, defining the task vector τ_t as the deviation between fine-tuned parameters $\Theta^{(t)}$ for the t-th task and the initial pre-trained parameters $\Theta^{(0)}$:

$$\tau_t = \Theta^{(t)} - \Theta^{(0)}, \tag{38}$$

and facilitating multi-task learning through the formulation $\Theta^{(\mathrm{merge})} = \Theta^{(0)} + \lambda \sum_{t=1}^{T} \tau_t$ [435]. While computationally efficient and conceptually elegant, this approach often encounters task interference arising from unmitigated parameter interactions, constraining its utility in scenarios demanding intricate task reconciliation.

Weighted Merging. This strategy dynamically allocates merging coefficients based on the significance of individual models, tailoring contributions to optimize fusion outcomes. MetaGPT [452] computes optimal weights by normalizing the squared L2 norm of each task vector:

$$\lambda_t^* = \frac{\|\tau_t\|^2}{\sum_{k=1}^{T} \|\tau_k\|^2}, \tag{39}$$

thereby assigning greater influence to tasks with more substantial parameter shifts, as indicated by higher $\|\tau_t\|^2$. SLERP [432] employs spherical interpolation to ensure smooth parameter transitions, preserving model continuity, while Layer-wise AdaMerging [453] refines this process by optimizing coefficients at a per-layer granularity, enhancing task-specific precision within the merged architecture.

Subspace Merging. This approach projects model parameters into sparse subspaces to minimize interference while upholding computational efficiency, addressing overlap in parameter contributions. TIES-Merging [436] retains the top 20% of parameters by magnitude, resolving sign conflicts to maintain coherence, DARE [437] scales sparse weights to curtail redundancy, and Concrete [454] utilizes bi-level optimization to craft adaptive masks, ensuring meticulous integration of model components with reduced interference across tasks.

Routing-based Merging. This technique dynamically fuses models based on input-specific attributes, enabling a context-responsive integration process. SMEAR [442] calculates sample-dependent expert weights to prioritize pertinent features, Weight-Ensembling MoE [455] employs input-driven routing of linear layers for selective activation, and Twin-Merging [456] melds task-shared and task-private knowledge, fostering a flexible merging framework that adapts to diverse input demands and enhances multi-task robustness.

Post-calibration. This technique corrects representation bias post-merging by aligning hidden representations of the unified model with those of its independent constituents, mitigating performance degradation. Representation Surgery [319] exemplifies this by refining representational consistency, bolstering the merged model's robustness and accuracy.

8 Datasets

Post-training techniques are meticulously engineered to refine the adaptability of LLMs to specialized domains or tasks, leveraging datasets as the cornerstone of this optimization process. A thorough examination of prior research [457, 82] underscores that the quality, diversity, and relevance of data profoundly influence model efficacy, often determining the success of post-training endeavors. To elucidate the critical role of datasets in this context, we present a comprehensive review and in-depth analysis of those employed in post-training phases, categorizing them into three principal types based on their collection methodologies: human-labeled data, distilled data, and synthetic data. These categories reflect distinct strategies in data curation, with models adopting either a singular approach or a hybrid methodology integrating multiple types to balance scalability, cost, and performance. **Tab. 9** provides a detailed overview of these dataset types, encompassing their origins, sizes, languages, tasks, and post-training phases (e.g., SFT and RLHF), which we explore in subsequent sections to highlight their contributions and challenges in advancing LLM capabilities.

8.1 Human-Labeled Datasets

Human-labeled datasets are distinguished by their exceptional accuracy and contextual fidelity, attributes derived from annotators' nuanced understanding of task intricacies and their ability to make precise, context-sensitive adjustments. These datasets serve as a cornerstone for refining instruction fine-tuning, significantly enhancing LLM performance across a diverse array of tasks by providing high-quality, expertly curated training signals. Within this category, prominent exemplars such as Flan [17], P3 (Public Pool of Prompts) [459], Sup-Natinst (Super-Natural Instructions) [462], and Dolly-15K [468] stand out as widely adopted resources in LLM post-training, each contributing unique strengths to the optimization of model capabilities through human expertise.

Human-Labeled Data for SFT. In the SFT phase, human-labeled datasets play an indispensable role, as demonstrated by the contributions of Flan, Sup-Natinst, and Dolly-15K, which deliver meticulously crafted prompt-response pairs and task-specific instructions to elevate LLM efficacy across diverse NLP benchmarks.

• **Flan.** The Flan dataset [17] constitutes a foundational resource, originally encompassing 62 widely recognized NLP benchmarks—such as HellaSwag [482], MRPC [483], and ANLI [484]—to facilitate robust multi-task learning in English with its 1.8 million examples. Recently, FlanV2 [34] has emerged as an advanced iteration, expanding its predecessor by integrating Flan [17], P3 [459], Sup-Natinst [462], and a

Table 9: Summary of Datasets Utilized in Post-training of Large Language Models (2021–2025). This table outlines key datasets, detailing their sizes, origins, release timelines, and attributes across three metrics: **Lang** (Language: EN for English, CN for Chinese, ML for Multilingual), **Task** (Type: MT for Multi-task, TS for Single-task), and **Phase** (Usage: SFT for Supervised Fine-Tuning, RLHF for Reinforcement Learning from Human Feedback). Datasets span from OpenAI Summarization to Magpie Reasoning V2, categorized by Human-Labeled, Distilled, and Synthetic types.

Datasets	Nums	Origin	Phase	Lang	Task	Categories	Time
OpenAI Summarization [458]	93K	Openai	RLHF	EN	MT	Human-Labeled	Jun-2021
Flan [17]	1.8M	Google	SFT	EN	MT	Human-Labeled	Sep-2021
P3 [459]	23M	Bigscience	RLHF	ML	MT	Human-Labeled	Oct-2021
SHP [460]	349K	Stanfordnlp	RLHF	EN	MT	Human-Labeled	Oct-2021
WebGPT [461]	18K	OpenAI	RLHF	EN	TS	Human-Labeled	Dec-2021
Sup-Natinst [462]	15K	Allenai	SFT	ML	MT	Human-Labeled	Apr-2022
HH-RLHF [104]	284K	Anthropic	RLHF	ML	TS	Human-Labeled	Apr-2022
Self-Instruct-52K [86]	52K	UW	SFT	EN	MT	Synthetic Data	Dec-2022
Unnatural Instructions [97]	52K	Orhonovich	SFT	EN	MT	Synthetic Data	Dec-2022
FlanV2 [34]	-	Google	SFT	EN	MT	Human-Labeled	Oct-2022
xP3 [463]	78M	Bigscience	RLHF	ML	MT	Human-Labeled	Nov-2022
Alpaca [464]	52K	Tatsu-lab	SFT	EN	MT	Synthetic Data	Mar-2023
Vicuna [465]	53K	UC Berkeley	SFT	EN	MT	Synthetic Data	Mar-2023
OpenAssistant [466]	84.4K, 161K	Laion.ai	SFT	ML	MT	Human-Labeled	Jan-2023
HC3 [467]	161K	Hello-SimpleAI	RLHF	ML	TS	Distilled Data	Jan-2023
Dolly-15K [468]	15K	Databricks	SFT	EN	TS	Human-Labeled	May-2023
ShareGPT [469]	90K	RyokoAI	RLHF	ML	MT	Distilled Data	Apr-2023
Alpaca-GPT4 [18]	52K	Microsoft	SFT	ML	MT	Synthetic Data	Apr-2023
Evol-Instruct [470]	70K, 143K	WizardLM	SFT	EN	MT	Synthetic Data	Apr-2023
Belle [471]	0.5M, 1.1M	BelleGroup	SFT	CN	MT	Synthetic Data	Apr-2023
Openorca [472]	4.5M	Together	SFT	EN	MT	Human-Labeled	Jun-2023
StackExchange [473]	4.5M	HuggingFace	RLHF	EN	MT	Human-Labeled	2023
OpenHermes-1 [474]	243K	Teknium	SFT	EN	MT	Synthetic Data	2023
OpenHermes-2.5 [475]	1M	Teknium	SFT	EN	MT	Synthetic Data	2023
UltraChat [476]	200K, 28M	Thnlp	SFT	EN	MT	Synthetic Data	Oct-2023
Instinwild [477]	52K	NUS	SFT	ML	MT	Synthetic Data	Oct-2023
Baize [478]	653K	Project-Baize	SFT	EN	MT	Synthetic Data	Dec-2023
WildChat [479]	1M	Allenai	SFT	EN	MT	Synthetic Data	May-2024
GenQA [480]	513K	UMD	SFT	EN	MT	Synthetic Data	Jun-2024
Magpie Llama 3 [481]	300k, 1M	Allenai	SFT/DPO	EN	MT	Synthetic Data	Jun-2024
Magpie Phi-3 [481]	300k, 1M	Allenai	SFT	EN	TS	Synthetic Data	Jun-2024
Magpie Qwen-2 [481]	200k, 300k, 1M	Allenai	SFT	EN	TS	Synthetic Data	Jul-2024
Magpie Llama-3.1 [481]	300k, 500k, 1M	Allenai	SFT/DPO	EN	MT	Synthetic Data	Jul-2024
Magpie Gemma-2 [481]	200k, 534k	Allenai	SFT	EN	TS	Synthetic Data	Jul-2024
Magpie Qwen-2.5 [481]	300k, 1M	Allenai	SFT	EN	TS	Synthetic Data	Oct-2024
Magpie Llama-3.3 [481]	500k, 1M	Allenai	SFT	EN	TS	Synthetic Data	Jan-2025
Magpie Reasoning V2 [481]	150k, 250k	Allenai	SFT	EN	TS	Synthetic Data	Jan-2025

plethora of additional datasets into a cohesive, comprehensive corpus, thereby amplifying its utility for SFT across diverse linguistic and task domains.

• **Sup-Natinst.** Super-Natural Instructions (Sup-Natinst) [462] offers an expansive and diverse array of 76 task types across 55 languages, establishing itself as a versatile resource for multilingual LLM post-training. Each task is meticulously paired with an instruction comprising a clear task definition—outlining the mapping from input text to desired output—and a set of examples that illustrate both correct and incorrect responses, providing a robust framework for guiding models toward precise task execution and enhancing cross-linguistic adaptability.

• **Dolly-15k.** Developed by Databricks employees, Dolly-15K [468] represents a curated corpus of 15,000 high-quality, human-generated prompt-response pairs, explicitly designed for instruction fine-tuning of LLMs. Encompassing a broad spectrum of topics and scenarios—including brainstorming, content generation, information extraction, open-ended question answering, and summarization—this dataset reflects a rich diversity of task types, enabling models to adapt flexibly to varied instructional contexts with enhanced contextual relevance.

The potency of human-labeled datasets in SFT stems from their extensive coverage of tasks and scenarios, a feature exemplified by the aforementioned corpora. Complementing these, OpenAssistant [466] delivers a substantial multilingual dialogue corpus derived from global crowdsourcing efforts, freely available to advance research pursuits, while OpenOrca [472] extends FlanV2 [34] with millions of GPT-3.5 and GPT-4 completions, constituting a dynamic, expanding resource for fine-tuning and task alignment. However, despite their significant contributions to model generalization, the challenge of ensuring consistent annotation quality and diversity persists, necessitating rigorous quality control to maximize their impact.

Human-Labeled Data for RLHF. For RLHF, human-labeled datasets such as P3, its multilingual extension xP3 [463], and SHP [460] provide essential human-annotated evaluations that refine LLM alignment with user preferences, offering a nuanced feedback mechanism for reward modeling.

• **P3.** The P3 dataset [459] is a meticulously curated instruction-tuning resource, aggregating 23 million multi-task prompts from the Hugging Face Hub, each accompanied by manually crafted instructions to span a diverse suite of NLP tasks, thereby providing a rich foundation for RLHF to enhance LLM adaptability and precision across varied applications.

• **xP3.** xP3 (Crosslingual Public Pool of Prompts) [463] extends P3 into a multilingual framework, encompassing prompts and supervised data across 46 languages and 16 NLP tasks, designed to support multitask prompted fine-tuning for models like BLOOMZ and mT0. Its content integrates the English P3 dataset, four novel English tasks (e.g., translation, program synthesis), and 30 multilingual NLP datasets, offering a comprehensive resource for cross-linguistic RLHF optimization.

• **SHP.** SHP [460] comprises 349,000 human preference annotations for responses to questions and instructions across 18 subject areas, evaluating response helpfulness to train RLHF reward models and assess natural language generation (NLG) quality, distinguished by its exclusive reliance on human-authored data, setting it apart from hybrid datasets like HH-RLHF.

These datasets enhance RLHF by providing diverse human-annotated evaluations that refine model alignment with user preferences. OpenAI Summarization [458] and Webgpt [461] offer structured, comparison-based feedback and Likert scale ratings, which help align model outputs more closely with human expectations. HH-RLHF [104] further strengthens this framework by including assessments of helpfulness and harmlessness, laying a solid foundation for models aimed at ensuring safety and ethical responses. Meanwhile, Stack-Exchange [473] contributes domain-specific, user-generated content that enriches training data, particularly benefiting models that require expertise in technical fields. However, these datasets encounter challenges such as scalability, potential biases from human annotations, and limited applicability beyond their specific domains. Thus, while they are valuable, these resources may need to be supplemented with broader datasets to achieve comprehensive model alignment across diverse real-world tasks.

8.2 Distilled Dataset

Distilled data arises from a sophisticated process of refining expansive raw datasets into compact, optimized subsets that preserve critical information for LLM training, balancing performance retention with enhanced training efficiency and reduced computational demands. This methodology yields datasets that often rival or surpass their unrefined counterparts in efficacy, accelerating model convergence and minimizing resource consumption, particularly within the RLHF phase. Key examples, ShareGPT [469] and HC3 (Human-ChatGPT Comparison Corpus) [467], exemplify this approach, serving as widely adopted resources for fine-tuning LLMs by distilling real-world interactions and comparative insights into actionable training signals.

• **ShareGPT.** ShareGPT [469] functions as a dynamic data collection platform, aggregating approximately 90,000 conversations uploaded via its API from authentic user interactions with ChatGPT or GPT-4. Com-

prising genuine human instructions and queries paired with corresponding AI responses, this dataset distills naturalistic dialogue patterns into a concentrated resource, enabling RLHF to refine LLMs' conversational fluency and contextual responsiveness with high relevance and quality.

• **HC3.** The HC3 dataset [467] is purposefully engineered to juxtapose AI-generated responses from Chat-GPT with human-authored answers, featuring 161,000 question-answer pairs across domains including open-ended topics, finance, medicine, law, and psychology. This distilled corpus facilitates comparative analysis of response characteristics and quality, empowering researchers to enhance LLMs' output authenticity and domain-specific accuracy during RLHF, while highlighting distinctions between human and AI-generated content.

8.3 Synthetic Datasets

Synthetic data constitutes a transformative asset in the SFT phase of LLM post-training, generated through AI models to deliver cost-effective, scalable, and privacy-preserving alternatives to human-labeled datasets. By automating the creation of instruction-response pairs and dialogues, synthetic data enables expansive training corpora that bolster model adaptability, with Self-Instruct-52K [86], Vicuna [465], and Baize [478] standing as principal examples widely utilized to enhance LLM instruction-following and dialogue generation capabilities.

Datasets Based on the Self-Instruct Method. Synthetic datasets employing the Self-Instruct method initiate with a modest set of manually crafted seed examples, leveraging LLMs to produce extensive instruction-following data that amplifies models' responsiveness to diverse directives, exemplified by Self-Instruct-52K, Alpaca, and the Magpie series, which collectively advance instruction tuning through scalable automation.

• **Self-Instruct-52K.** Self-Instruct-52K [86] establishes a foundational benchmark for instruction-following models, generating 52,000 examples from manually crafted seeds using a variety of prompt templates to guide LLMs, thereby enhancing their ability to interpret and execute task-specific instructions with precision and consistency.

• **Alpaca.** Alpaca [464] and Alpaca-GPT4 [18] expand an initial set of 175 seed pairs into 52,000 high-quality instruction-response pairs using GPT-3 and GPT-4, respectively, improving instruction-following proficiency, while InstInWild [477] adapts this approach for multilingual contexts, generating English and Chinese datasets to bolster cross-linguistic adaptability.

• **Magpie Datasets.** The Magpie datasets [481] harness aligned LLMs to generate instruction-response pairs from predefined templates, yielding specialized families like Magpie Reasoning V2 (emphasizing chain-of-thought reasoning), Magpie Llama-3 and Qwen-2 Series (tailored to popular models), Magpie Gemma-2 (for the Gemma architecture), and variants like Magpie-Air-DPO incorporating preference optimization signals, collectively enhancing SFT and instruction tuning across conversational and reasoning tasks.

Beyond these, datasets like Unnatural Instructions [97] (240K examples), Evol-Instruct [470] (70K-143K refined entries via iterative complexity enhancement), and Belle [471] (0.5M-1.1M Chinese dialogues from ChatGPT) significantly scale instruction generation, though challenges in quality assurance, complexity calibration, and bias mitigation persist, necessitating ongoing refinement to ensure reliability in intricate applications.

Datasets Based on Self-Chat Methods. Self-Chat datasets employ a technique where models simulate multi-turn conversations internally or with peers, enhancing dialogue generation capabilities and addressing deficiencies in existing corpora, with Baize, UltraChat, and OpenHermes exemplifying this approach through automated interaction strategies.

• **Baize.** Baize [478] utilizes ChatGPT's Self-Chat technique to produce 653,000 multi-turn dialogues, integrating seed data from Quora, Stack Overflow, and Alpaca to enrich instruction-following quality, thereby refining LLMs' conversational coherence and task adherence for SFT.

• **UltraChat.** UltraChat [476] employs multiple ChatGPT APIs to generate over 12 million high-quality conversation records across diverse topics, overcoming prevalent issues in multi-turn datasets like poor quality and inaccurate annotations, providing a robust SFT resource for dialogue enhancement.

• **Openhermes.** OpenHermes, developed by Teknium, includes OpenHermes-1 [474] (243K entries) and its expanded successor OpenHermes-2.5 [475] (1M entries), offering high-quality SFT datasets with increased volume and diversity, spanning a wide array of topics and task types to bolster dialogue and instruction-following proficiency.

These Self-Chat datasets enable models to craft multi-turn dialogues through self-interaction, as seen in Baize's use of ChatGPT with diverse seeds and UltraChat's extensive API-driven conversations, significantly improving dialogue quality and filling critical gaps in training data availability.

Datasets Based on Real User Interactions. Datasets derived from real user interactions harness authentic conversational exchanges with LLMs, capturing diverse and genuine inputs to enhance models' capacity to address real-world scenarios, with Vicuna, WildChat, and GenQA serving as key examples of this approach.

• **Vicuna.** Vicuna [465] is fine-tuned on approximately 70,000 user-shared conversations from ShareGPT's public API, processed by converting HTML to markdown, filtering low-quality samples, and segmenting lengthy dialogues to fit model context lengths, ensuring high-quality SFT data for realistic interaction modeling.

• **WildChat.** WildChat [479] comprises 1 million real-world user-ChatGPT interactions across multiple languages and prompt types, featuring unique exchanges like ambiguous requests and code-switching, serving dual purposes as an SFT resource and a tool for analyzing user behavior.

• **GenQA.** GenQA [480] offers a vast SFT dataset of over 10 million cleaned and filtered instruction samples, generated entirely by LLMs without human input or complex pipelines, complementing existing corpora by rapidly producing synthetic data to address coverage gaps.

Synthetic data's advantages in cost, scalability, and privacy are tempered by potential deficits in depth and authenticity compared to human-labeled counterparts, risking bias propagation and oversimplification. Reliance on AI-generated content may perpetuate model-inherent errors, underscoring the necessity of integrating synthetic and human-generated data to bolster LLM robustness and applicability across diverse contexts.

9 Applications

Despite the robust foundational capabilities imparted by pre-training, Large Language Models (LLMs) frequently encounter persistent limitations when deployed in specialized domains, including constrained context lengths, tendencies toward hallucination, suboptimal reasoning proficiency, and ingrained biases. These shortcomings assume critical significance in real-world applications, where precision, reliability, and ethical alignment are paramount. Such challenges prompt fundamental inquiries: (1) How can LLM performance be systematically enhanced to meet domain-specific demands? (2) What strategies can effectively mitigate the practical obstacles inherent in applied settings? Post-training emerges as a pivotal solution, augmenting LLMs' adaptability by refining their recognition of domain-specific terminology and reasoning patterns while preserving their broad-spectrum competencies. This chapter delineates the transformative applications of post-trained LLMs across professional, technical, and interactive domains, elucidating how tailored post-training methodologies address these challenges and elevate model utility in diverse contexts.

9.1 Professional Domains

Legal Assistant. The legal domain exemplifies a compelling arena for leveraging post-training to imbue LLMs with specialized expertise, enabling them to navigate the intricate landscape of legal knowledge and address multifaceted challenges inherent in jurisprudence. A burgeoning body of research [485, 486, 487] has investigated LLM applications in this field, spanning legal question answering [488, 489], judgment prediction [490, 491], document summarization [492, 493], and broader tasks like retrieval enhancement and judicial reasoning [494, 495, 496]. Post-trained legal assistants, such as those exemplified by LawGPT [497] and Lawyer-LLaMA [498], have demonstrated remarkable proficiency, not only offering dependable guidance across diverse legal matters but also achieving success in professional qualification exams, a testament to their advanced interpretive and analytical capabilities. Multilingual support, as seen in models like Lexi-iLaw [499] and SAUL [500], extends this utility to languages including English and Chinese, broadening

accessibility. Central to these advancements is post-training on curated legal corpora, such as ChatLaw [501], which integrates extensive legal texts into conversational datasets, enabling models to refine their reasoning and terminology recognition.

Healthcare and Medical. Post-training substantially elevates LLM performance across a spectrum of healthcare and medical applications, harnessing domain-specific data to address clinical and academic needs with precision. In clinical settings, LLMs facilitate tasks such as drug discovery [502], drug synergy prediction [503] and catalyst design [504], diagnostic support, medical record generation, and patient interaction, while in academia, they excel in medical report synthesis [505] and question answering [506], driven by performance gains from tailored post-training. For instance, ChatMed [507], honed on 500,000 medical consultation records, exemplifies enhanced diagnostic and consultative accuracy, while PULSE [508], fine-tuned with 4 million instructions spanning Chinese medical and general domains, showcases superior multi-task proficiency. These models outperform their general-purpose counterparts by leveraging post-trained adaptations that embed nuanced medical knowledge, highlighting the indispensability of customized datasets in achieving practical utility. Such advancements not only improve task-specific outcomes but also pave the way for integrating LLMs into healthcare workflows, where precision and contextual relevance are non-negotiable, underscoring post-training's transformative impact on real-world medical applications.

Finance and Economics. In the domains of finance and economics, LLMs exhibit considerable potential for tasks including sentiment analysis [509], information extraction [510], and question answering [511], with post-training amplifying their efficacy through domain-specific refinements. While general-purpose LLMs provide a solid foundation, specialized models like FinGPT [512] and DISC-FinLLM [513] demonstrate marked improvements when post-trained on financial corpora, excelling in tasks requiring nuanced understanding of market dynamics and terminology. Similarly, XuanYuan [514] employs extensive financial datasets and advanced post-training techniques to enhance accuracy in economic modeling and prediction, outperforming untuned benchmarks. These developments illustrate post-training's critical role in adapting LLMs to the intricate demands of financial applications, where precision in interpreting quantitative data and qualitative insights is paramount, ensuring models deliver reliable, domain-informed outputs that align with industry standards and expectations.

Mobile Agents. The evolution of large multi-modal models (LMMs) has catalyzed a burgeoning domain of agentic research focused on LMM-based graphical user interface (GUI) agents [515]. This field aims to develop AI assistants capable of executing tasks across diverse GUI environments, encompassing web interfaces [516, 517, 518, 519, 520], personal computing platforms [521, 522, 523, 524, 525], and mobile devices [526, 527, 528, 529, 530]. Within the mobile context, one research trajectory enhances the perceptual and reasoning capacities of individual agents through tool integration [526] and an additional exploration phase [527, 528]. Recent advancements exhibit considerable potential by employing multi-agent systems for decision-making and reflection [531, 529], thereby improving task efficacy. Notably, MobileAgent-E [532] introduces a hierarchical structure among agents, facilitating robust long-horizon planning and elevating the precision of low-level actions. These developments underscore the transformative role of multi-modal post-training strategies in fostering adaptive, efficient agents for complex mobile environments.

9.2 Technical and Logical Reasoning

Mathematical Reasoning. LLMs demonstrate significant promise in mathematical reasoning, spanning algebraic manipulations, calculus, and statistical analysis, with post-training pivotal in bridging the gap between computational and human-like proficiency. GPT-4 [9] achieves high scores on standardized math assessments, a feat attributed to its diverse pre-training corpus, yet post-training further refines this capability. DeepSeekMath [64], for instance, leverages specialized mathematical datasets and techniques like Supervised Fine-Tuning (SFT) and Group Relative Policy Optimization (GRPO) [64] to enhance its reasoning precision, tackling complex problems with structured chains of thought (CoT). OpenAI's o1 [41] advances this frontier through reinforcement learning (RL), iteratively optimizing reasoning strategies to achieve superior performance in multi-step derivations and proofs. This continuous refinement via post-training not only elevates accuracy but also aligns LLM outputs with rigorous mathematical logic, positioning them as valuable tools in educational and research contexts where advanced reasoning is essential.

Code Generation. Post-training has revolutionized code generation, empowering LLMs to excel in automated coding, debugging, and documentation, thereby transforming software development workflows. Codex [533], trained on a vast, diverse codebase, underpins GitHub Copilot *, delivering real-time coding assistance with remarkable accuracy. Specialized models like Code Llama [384] further refine this capability, leveraging post-training on programming-specific datasets to assist developers across languages and frameworks. OpenAI's o1 [41] extends its mathematical reasoning prowess to code generation, producing high-quality, context-aware code snippets that rival human outputs. Current research focuses on enhancing personalization, deepening contextual understanding, and embedding ethical safeguards to mitigate risks like code misuse, ensuring LLMs maximize productivity while adhering to responsible development principles in technical domains.

9.3 Understanding and Interaction

Recommendation System. LLMs have emerged as transformative agents in recommendation systems, analyzing user interactions, product descriptions, and reviews to deliver personalized suggestions with unprecedented granularity [534, 535, 536]. Post-training enhances their capacity to integrate sentiment analysis, enabling nuanced comprehension of content and emotional undertones, as evidenced in models like GPT-4 [9] and specialized systems like LLaRA [537] and AgentRec [538]. E-commerce giants such as Amazon and Taobao harness these capabilities to process review sentiments, search queries, and purchase histories, refining customer preference models and predicting interests with high fidelity [535]. Beyond ranking items, post-trained LLMs engage in conversational recommendation, planning, and content generation, elevating user experience by offering dynamic, context-sensitive interactions that adapt to evolving preferences, a testament to post-training's role in bridging data analysis with practical utility.

Speech Conversation. Post-trained LLMs have redefined speech processing, advancing recognition, synthesis, and translation to unprecedented levels of naturalness and accuracy [539]. These models tackle tasks like text-to-speech [540], text-to-audio generation [541], and speech recognition [542], powering ubiquitous tools such as Amazon's Alexa, Apple's Siri, and Alibaba's Tmall Genie. Whisper [543] exemplifies this progress with its high-fidelity transcription, while GPT-4o [78] introduces real-time voice interaction, merging multimodal inputs seamlessly. Future trajectories include multilingual translation and personalized voice synthesis, where post-training refines LLMs to break language barriers and tailor responses to individual user profiles, enhancing accessibility and engagement in human-computer interactions across global contexts.

Video Understanding. The extension of LLMs into video understanding marks a significant frontier, with post-training enabling models like Video-LLaMA [341] to perform captioning, summarization, and content analysis, streamlining multimedia creation and comprehension. Sora [544] further revolutionizes this domain by generating complex videos from textual prompts, democratizing content production by reducing technical barriers and fostering innovative storytelling. These advancements leverage post-training to adapt LLMs to visual-temporal data, enhancing their interpretative depth and utility in applications ranging from education to entertainment. However, they introduce challenges in computational scalability, privacy protection, and ethical governance, particularly concerning generated content misuse. As post-training methodologies evolve, addressing these issues will be imperative to ensure sustainable, responsible deployment in video-related applications, balancing innovation with societal considerations.

10 Open Problems and Future Directions

In this section, we critically evaluate the unresolved challenges and prospective trajectories in post-training methodologies for Large Language Models (LLMs), situating our analysis within the transformative advancements heralded by the releases of OpenAI's o1 [41] and DeepSeek-R1 [28]. These models, leveraging large-scale reinforcement learning (RL), have redefined reasoning benchmarks, yet their emergence amplifies the urgency of addressing persistent limitations in post-training techniques. The following subsections delineate six pivotal open problems, each underscored by its critical importance to the field's progression and the pressing need for resolution, alongside feasible strategies to propel future research and ensure the responsible evolution of LLMs across diverse applications.

---

*https://github.com/features/copilot

Reasoning Enhancement Beyond Large-Scale RL. The introduction of o1 and DeepSeek-R1 has marked a paradigm shift in LLM reasoning capabilities, harnessing extensive RL frameworks like RLHF and Group Relative Policy Optimization (GRPO) to achieve unprecedented accuracy in multi-step problem-solving, such as mathematical proofs and logical derivations. However, the reliance on binary reward signals and extensive human feedback exposes a critical limitation: their inability to generalize effectively across complex, open-ended tasks, such as scientific hypothesis generation or strategic decision-making in dynamic environments. This gap is urgent, as the demand for LLMs to emulate human-like reasoning in real-world contexts grows, and its importance lies in unlocking their potential as autonomous intellectual agents beyond current benchmarks. Current RL approaches struggle with reward sparsity and lack adaptability to task complexity, necessitating innovative frameworks. Feasible solutions include developing multi-objective RL systems that integrate self-supervised consistency checks (e.g., verifying logical coherence across reasoning steps) and domain-specific priors, such as mathematical axioms or scientific principles, to guide inference without exhaustive human annotations [545, 546]. Such advancements could reduce dependency on costly feedback loops, enhance scalability, and enable LLMs to tackle uncharted reasoning domains, a prospect made tangible by DeepSeek-R1's cold-start RL innovations.

Scalability of Post-Training for Next-Generation LLMs. As LLMs escalate in size and complexity, exemplified by the parameter-heavy architectures of next-generation models, the scalability of post-training emerges as a formidable and pressing challenge. The resource-intensive nature of RL-based methods, such as DeepSeek-R1's cold-start approach requiring extensive computational infrastructure, restricts accessibility to well-funded entities and raises significant sustainability concerns, particularly for multi-modal applications (e.g., video analysis) and real-time systems (e.g., conversational agents). This issue is critically important, as it threatens to widen the gap between resource-rich and resource-constrained research communities, impeding equitable progress in LLM development. While Parameter-Efficient Fine-Tuning (PEFT) [92] mitigates some overhead, its performance often degrades on large-scale datasets, underscoring the need for scalable alternatives. Viable future directions [547, 548, 549] include the design of lightweight RL algorithms—potentially adapting GRPO for reduced memory footprints—federated post-training frameworks that distribute computational loads across decentralized networks, and advanced distillation techniques that preserve reasoning and adaptability while minimizing resource demands. These solutions, if realized, could democratize post-training, aligning with the field's urgent need for sustainable and inclusive innovation.

Ethical Alignment and Bias Mitigation in RL-Driven Models. Post-training via RL, as demonstrated in o1's cautious alignment strategies, amplifies ethical risks by potentially reinforcing biases embedded in training datasets like HH-RLHF [104] or synthetic corpora, a challenge of paramount urgency given LLMs' deployment in sensitive domains such as healthcare diagnostics and judicial decision-making. The dynamic variability of ethical alignment—where fairness in one cultural context may constitute bias in another—poses a significant barrier to achieving universally trustworthy LLMs, making this issue critically important for ensuring equitable and safe AI systems. Current methods risk over-censorship, compromising utility (e.g., stifling creative outputs), or under-correction, perpetuating harmful biases (e.g., racial or gender disparities). Addressing this demands the development of fairness-aware RL objectives, incorporating multi-stakeholder preference models (e.g., aggregating diverse human judgments) and adversarial debiasing techniques to neutralize dataset biases during training. The feasibility of these approaches [550] is bolstered by recent advances in interpretability tools and multi-objective optimization, enabling a balanced trade-off between ethical robustness and practical functionality, a necessity underscored by o1's real-world deployment challenges.

Seamless Multi-Modal Integration for Holistic Reasoning. The trajectory toward multi-modal LLMs, foreshadowed by o1's reasoning enhancements and GPT-4o's synthesis capabilities [78], accentuates an urgent need for post-training methods that seamlessly integrate text, images, audio, and other data types to enable holistic reasoning—a capability critical for applications like real-time video analysis, augmented reality, and cross-modal scientific inquiry. Current approaches falter in achieving robust cross-modal alignment due to data heterogeneity and the scarcity of comprehensive multi-modal training corpora, limiting LLMs' ability to reason across diverse inputs cohesively. This challenge's importance lies in its potential to unlock transformative applications, yet its resolution remains elusive without scalable frameworks. DeepSeek-R1's cold-start RL offers a promising starting point, suggesting that unified modal encoders (e.g., capable of encoding heterogeneous data into a shared latent space) and dynamic RL policies that adaptively weight modal contributions

could bridge this gap. Future research should prioritize the creation of multi-modal benchmarks and synthetic datasets, building on efforts like Magpie [481], to drive progress, a feasible endeavor given recent strides in multi-modal pre-training and RL optimization.

Context-Adaptive Trustworthiness Frameworks. Trustworthiness in post-trained LLMs is increasingly recognized as a dynamic, context-dependent attribute rather than a static quality, as evidenced by o1's cautious outputs in sensitive domains like education versus its freer responses in creative tasks. This variability—where safety imperatives (e.g., avoiding misinformation in educational settings) may conflict with utility demands (e.g., fostering creativity in writing)—poses an urgent challenge, given its critical importance to user trust and LLM applicability across diverse real-world scenarios. Current post-training methods often over-prioritize safety, yielding utility trade-offs that diminish practical value, or fail to adapt to context-specific needs, undermining reliability. Resolving this requires context-sensitive RL models that dynamically adjust safety-utility trade-offs, leveraging real-time user feedback and interpretable safety metrics (e.g., transparency scores for generated outputs) to ensure adaptability. The feasibility of this approach [551] is supported by advances in adaptive learning systems and real-time monitoring, offering a pathway to balance trustworthiness with functionality, a pressing need as LLMs like o1 expand into high-stakes applications.

Accessibility and Democratization of Post-Training Innovations. The computational intensity of advanced post-training methods, epitomized by DeepSeek-R1's RL-driven approach, confines their application to resource-rich entities, presenting an urgent barrier to accessibility that stifles innovation within smaller research communities and industry sectors (i.e., an issue of paramount importance for fostering equitable progress in AI). This exclusivity not only limits the diversity of contributions but also hampers the field's ability to address global challenges collaboratively. Democratizing these innovations demands the development of efficient, open-source tools and frameworks that lower entry barriers without compromising quality, a goal rendered feasible through scalable PEFT adaptations for RL [92], collaborative platforms for sharing post-trained models (e.g., Hugging Face hubs), and streamlined synthetic data generation pipelines akin to Magpie [481]. Future efforts should focus on optimizing these solutions to enable widespread adoption, ensuring that the transformative potential of post-training—exemplified by o1 and DeepSeek-R1—extends beyond elite institutions to enrich the broader AI ecosystem.

Creative Intelligence & System 2 Thinking. The integration of creative intelligence into System 2 reasoning represents an emergent frontier in the evolution of LLMs, as highlighted in [552]. While reasoning LLMs like OpenAI's o1 and DeepSeek's R1 excel in deliberate, step-by-step logical analysis—mimicking System 2 thinking—their capacity for creative intelligence, which involves generating novel ideas, synthesizing disparate concepts, and adapting flexibly to unstructured problems, remains underexplored. This gap is critical, as creative intelligence underpins human-like problem-solving in domains such as artistic creation, scientific discovery, and strategic innovation, where rigid logical frameworks alone are insufficient. The urgency of this challenge lies in its potential to elevate LLMs from analytical tools to autonomous creative agents, a transformative leap toward Artificial General Intelligence (AGI). Below, we outline this open problem and propose future directions, drawing on insights from the survey.

11 Conclusion

This paper offers the first exhaustive survey of Post-training Language Models (PoLMs), systematically tracing their trajectory from ChatGPT's alignment origins in 2018 to DeepSeek-R1's reasoning milestone in 2025, and affirming their transformative influence on reasoning precision, domain adaptability, and ethical integrity. We have evaluated a broad spectrum of techniques (i.e., Fine-tuning, Alignment, Reasoning, Efficiency, and Integration and Adaptation), synthesizing their contributions across professional, technical, and interactive domains, from legal analysis to multi-modal comprehension. Our analysis underscores that PoLMs have markedly advanced LLM capabilities, evolving from initial alignment innovations to sophisticated reasoning frameworks; nonetheless, it reveals persistent challenges, including bias persistence, computational scalability, and context-variable ethical alignment. These findings, encapsulated within a novel taxonomy, emphasize the necessity of an integrative approach that aligns reasoning advancements with efficiency and ethical imperatives. We conclude that sustained interdisciplinary collaboration, rigorous methodological assessment, and the development of adaptive, scalable frameworks are critical to realizing LLMs' potential as reliable, responsible tools across diverse applications. As the pioneering survey of its kind, this work consolidates PoLMs progress over recent years and lays a robust intellectual foundation, inspiring future research to cultivate LLMs that adeptly integrate precision, ethical robustness, and versatility to meet the evolving demands of scientific and societal contexts.

References

[1] Alec Radford. Improving language understanding by generative pre-training. 2018.

[2] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *ArXiv*, abs/1103.0398, 2011.

[4] Swapan Ghosh. Developing artificial intelligence (ai) capabilities for data-driven business model innovation: Roles of organizational adaptability and leadership. *Journal of Engineering and Technology Management*, 75:101851, 2025.

[5] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

[6] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[7] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[8] Boyu Zhang, Hongyang Yang, and Xiao-Yang Liu. Instruct-fingpt: Financial sentiment analysis by instruction tuning of general-purpose large language models, 2023.

[9] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[10] Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, et al. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*, 2023.

[11] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

[12] Tomas Mikolov. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[13] Zhilin Yang. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

[14] 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.

[15] M Lewis. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[16] 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, 2020.

[17] 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.

[18] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

[19] Yixuan Su, Tian Lan, Huayang Li, Jialu Xu, Yan Wang, and Deng Cai. Pandagpt: One model to instruction-follow them all. *arXiv preprint arXiv:2305.16355*, 2023.

[20] Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. Evaluating large language models at evaluating instruction following. *arXiv preprint arXiv:2310.07641*, 2023.

[21] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

[22] Ohad Rubin, Jonathan Herzig, and Jonathan Berant. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*, 2021.

[23] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690, 2024.

[25] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[26] Google. Gemini-2.0-flash. 2025.

[27] Claude 3. The claude 3 model family: Opus, sonnet, haiku.

[28] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Jun-Mei Song, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. 2025.

[29] Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Ptqd: Accurate post-training quantization for diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

[30] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. *arXiv preprint arXiv:2311.17043*, 2023.

[31] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[32] Bo Adler, Niket Agarwal, Ashwath Aithal, Dong H Anh, Pallab Bhattacharya, Annika Brundyn, Jared Casper, Bryan Catanzaro, Sharon Clay, Jonathan Cohen, et al. Nemotron-4 340b technical report. *arXiv preprint arXiv:2406.11704*, 2024.

[33] Stephen H Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, et al. Promptsource: An integrated development environment and repository for natural language prompts. *arXiv preprint arXiv:2202.01279*, 2022.

[34] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*, 2023.

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

[36] Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. Pre-trained models: Past, present and future. *AI Open*, 2:225–250, 2021.

[37] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *Science China technological sciences*, 63(10):1872–1897, 2020.

[38] Zhichao Wang, Bin Bi, Shiva Kumar Pentyala, Kiran Ramnath, Sougata Chaudhuri, Shubham Mehrotra, Xiang-Bo Mao, Sitaram Asur, et al. A comprehensive survey of llm alignment techniques: Rlhf, rlaif, ppo, dpo and more. *arXiv preprint arXiv:2407.16216*, 2024.

[39] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *ArXiv*, abs/2403.14608, 2024.

[40] Tong Xiao and Jingbo Zhu. Foundations of large language models. *ArXiv*, abs/2501.09223, 2025.

[41] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[42] Colin Raffel, Noam M. 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. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2019.

[43] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, 2021.

[44] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[45] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022.

[46] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

[47] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, F. Xia, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903, 2022.

[48] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[49] 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, 2024.

[50] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[51] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[52] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[53] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *ArXiv*, abs/2304.08485, 2023.

[54] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

[55] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

[56] 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*, 2024.

[57] Maciej Świechowski, Konrad Godlewski, Bartosz Sawicki, and Jacek Ma'ndziuk. Monte carlo tree search: a review of recent modifications and applications. *Artificial Intelligence Review*, 56:2497–2562, 2021.

[58] Zhihong Shao, Damai Dai, Daya Guo, Bo Liu (Benjamin Liu), Zihan Wang, and Huajian Xin. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *ArXiv*, abs/2405.04434, 2024.

[59] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28, 2015.

[60] Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. Tl; dr: Mining reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, 2017.

[61] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019.

[62] Dongyoung Go, Tomasz Korbak, Germán Kruszewski, Jos Rozen, Nahyeon Ryu, and Marc Dymetman. Aligning language models with preferences through f-divergence minimization, 2023.

[63] Natasha Jaques, Judy Hanwen Shen, Asma Ghandeharioun, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Shane Gu, and Rosalind Picard. Human-centric dialog training via offline reinforcement learning, 2020.

[64] 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.

[65] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, 2023.

[66] Hugo Touvron, Louis Martin, Kevin R. Stone, et al. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288, 2023.

[67] DeepSeek-AI Xiao Bi, Deli Chen, Guanting Chen, et al. Deepseek llm: Scaling open-source language models with longtermism. *ArXiv*, abs/2401.02954, 2024.

[68] DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, et al. Deepseek-v3 technical report. *ArXiv*, abs/2412.19437, 2024.

[69] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, et al. Qwen technical report. *ArXiv*, abs/2309.16609, 2023.

[70] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, et al. Qwen2 technical report. *ArXiv*, abs/2407.10671, 2024.

[71] Qwen An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024.

[72] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[73] Mistral. Mistral-large2. 2024.

[74] Anthropic. claude2. 2023.

[75] Claude Ahtropic. Claude. [Online]. Available: `https://www.anthropic.com/claude`, 2024.

[76] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[77] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[78] OpenAI. GPT-4o System Card, 2024.

[79] OpenAI. Openai-o3. 2025.

[80] Team Glm Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Ming yue Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiaoyu Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yi An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhenyi Yang, Zhengxiao Du, Zhen-Ping Hou, and Zihan Wang. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *ArXiv*, abs/2406.12793, 2024.

[81] Databricks. Databricks/dbrx. 2024.

[82] Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.

[83] Jamba Team Barak Lenz, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, Daniel Gissin, Daniel Jannai, Dor Muhlgay, Dor Zimberg, Edden M. Gerber, Elad Dolev, Eran Krakovsky, Erez Safahi, Erez Schwartz, Gal Cohen, Gal Shachaf, Haim Rozenblum, Hofit Bata, Ido Blass, Inbal Magar, Itay Dalmedigos, Jhonathan Osin, Julie Fadlon, Maria Rozman, Matan Danos, Michael Gokhman, Mor Zusman, Naama Gidron, Nir Ratner, Noam Gat, Noam Rozen, Oded Fried, Ohad Leshno, Omer Antverg, Omri Abend, Opher Lieber, Or Dagan, Orit Cohavi, Raz Alon, Ro'i Belson, Roi Cohen, Rom Gilad, Roman Glozman, Shahar Lev, Shaked Haim Meirom, Tal Delbari, Tal Ness, Tomer Asida, Tom Ben Gal, Tom Braude, Uriya Pumerantz, Yehoshua Cohen, Yonatan Belinkov, Yuval Globerson, Yuval Peleg Levy, and Yoav Shoham. Jamba-1.5: Hybrid transformer-mamba models at scale. *ArXiv*, abs/2408.12570, 2024.

[84] Amazon Artificial General Intelligence. The amazon nova family of models: Technical report and model card. *Amazon Technical Reports*, 2024.

[85] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, et al. Kimi k1.5: Scaling reinforcement learning with llms. 2025.

[86] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

[87] Kavita A. Ganesan. Rouge 2.0: Updated and improved measures for evaluation of summarization tasks. *ArXiv*, abs/1803.01937, 2015.

[88] Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. *ArXiv*, abs/2308.12032, 2023.

[89] Qianlong Du, Chengqing Zong, and Jiajun Zhang. Mods: Model-oriented data selection for instruction tuning. *ArXiv*, abs/2311.15653, 2023.

[90] Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. Instruction mining: Instruction data selection for tuning large language models. 2023.

[91] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[92] J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685, 2021.

[93] Kai Lv, Yuqing Yang, Tengxiao Liu, Qi jie Gao, Qipeng Guo, and Xipeng Qiu. Full parameter fine-tuning for large language models with limited resources. *ArXiv*, abs/2306.09782, 2023.

[94] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Frederick Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. *ArXiv*, abs/1710.03740, 2017.

[95] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *ArXiv*, abs/1604.06174, 2016.

[96] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction tuning for large language models: A survey. *ArXiv*, abs/2308.10792, 2023.

[97] Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. *ArXiv*, abs/2212.09689, 2022.

[98] Zhen-Ru Zhang, Chuanqi Tan, Haiyang Xu, Chengyu Wang, Jun Huang, and Songfang Huang. Towards adaptive prefix tuning for parameter-efficient language model fine-tuning. *arXiv preprint arXiv:2305.15212*, 2023.

[99] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[100] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *AI Open*, 5:208–215, 2024.

[101] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

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

[103] Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. *ArXiv*, abs/2401.08967, 2024.

[104] 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*, 2022.

[105] 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*, 2022.

[106] Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

[107] Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18990–18998, 2024.

[108] Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Ren Lu, Thomas Mesnard, Johan Ferret, Colton Bishop, Ethan Hall, Victor Carbune, and Abhinav Rastogi. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. 2023.

[109] Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. Some things are more cringe than others: Iterative preference optimization with the pairwise cringe loss, 2024.

[110] Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*, 2024.

[111] Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

[112] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, 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.

[113] Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, et al. Lipo: Listwise preference optimization through learning-to-rank. *arXiv preprint arXiv:2402.01878*, 2024.

[114] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.

[115] Shitong Duan, Xiaoyuan Yi, Peng Zhang, Tun Lu, Xing Xie, and Ning Gu. Negating negatives: Alignment without human positive samples via distributional dispreference optimization. *arXiv preprint arXiv:2403.03419*, 2024.

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

[117] Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. Direct nash optimization: Teaching language models to self-improve with general preferences, 2024.

[118] 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*, 2024.

[119] Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*, 2024.

[120] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.

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

[122] Zhichao Wang, Bin Bi, Can Huang, Shiva Kumar Pentyala, Zixu James Zhu, Sitaram Asur, and Na Claire Cheng. Una: Unifying alignments of rlhf/ppo, dpo and kto by a generalized implicit reward function, 2024.

[123] Jiaming Ji, Boyuan Chen, Hantao Lou, Donghai Hong, Borong Zhang, Xuehai Pan, Juntao Dai, Tianyi Qiu, and Yaodong Yang. Aligner: Efficient alignment by learning to correct, 2024.

[124] Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. *arXiv preprint arXiv:2312.14925*, 2023.

[125] Baicen Xiao, Qifan Lu, Bhaskar Ramasubramanian, Andrew Clark, Linda Bushnell, and Radha Poovendran. Fresh: Interactive reward shaping in high-dimensional state spaces using human feedback. *arXiv preprint arXiv:2001.06781*, 2020.

[126] Riad Akrour, Marc Schoenauer, and Michele Sebag. Preference-based policy learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I 11*, pages 12–27. Springer, 2011.

[127] Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv preprint arXiv:1909.12200*, 2019.

[128] Jerry Zhi-Yang He and Anca D Dragan. Assisted robust reward design. *arXiv preprint arXiv:2111.09884*, 2021.

[129] Yuchen Cui, Qiping Zhang, Brad Knox, Alessandro Allievi, Peter Stone, and Scott Niekum. The empathic framework for task learning from implicit human feedback. In *Conference on Robot Learning*, pages 604–626. PMLR, 2021.

[130] Jianlan Luo, Perry Dong, Yuexiang Zhai, Yi Ma, and Sergey Levine. Rlif: Interactive imitation learning as reinforcement learning. *arXiv preprint arXiv:2311.12996*, 2023.

[131] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

[132] Gaurav R Ghosal, Matthew Zurek, Daniel S Brown, and Anca D Dragan. The effect of modeling human rationality level on learning rewards from multiple feedback types. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5983–5992, 2023.

[133] Lin Guan, Mudit Verma, Suna Sihang Guo, Ruohan Zhang, and Subbarao Kambhampati. Widening the pipeline in human-guided reinforcement learning with explanation and context-aware data augmentation. *Advances in Neural Information Processing Systems*, 34:21885–21897, 2021.

[134] Andreea Bobu, Marius Wiggert, Claire Tomlin, and Anca D Dragan. Inducing structure in reward learning by learning features. *The International Journal of Robotics Research*, 41(5):497–518, 2022.

[135] Andreea Bobu, Yi Liu, Rohin Shah, Daniel S Brown, and Anca D Dragan. Sirl: similarity-based implicit representation learning. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 565–574, 2023.

[136] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[137] Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pages 783–792. PMLR, 2019.

[138] Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR, 2019.

[139] Alexander Irpan, Kanishka Rao, Konstantinos Bousmalis, Chris Harris, Julian Ibarz, and Sergey Levine. Off-policy evaluation via off-policy classification. *Advances in Neural Information Processing Systems*, 32, 2019.

[140] Adam Gleave, Michael Dennis, Shane Legg, Stuart Russell, and Jan Leike. Quantifying differences in reward functions. *arXiv preprint arXiv:2006.13900*, 2020.

[141] Blake Wulfe, Ashwin Balakrishna, Logan Ellis, Jean Mercat, Rowan McAllister, and Adrien Gaidon. Dynamics-aware comparison of learned reward functions. *arXiv preprint arXiv:2201.10081*, 2022.

[142] Erik Jenner, Joar Max Viktor Skalse, and Adam Gleave. A general framework for reward function distances. In *NeurIPS ML Safety Workshop*, 2022.

[143] Joar Skalse, Lucy Farnik, Sumeet Ramesh Motwani, Erik Jenner, Adam Gleave, and Alessandro Abate. Starc: A general framework for quantifying differences between reward functions. *arXiv preprint arXiv:2309.15257*, 2023.

[144] Erik Jenner and Adam Gleave. Preprocessing reward functions for interpretability. *arXiv preprint arXiv:2203.13553*, 2022.

[145] Lingfeng Shen, Sihao Chen, Linfeng Song, Lifeng Jin, Baolin Peng, Haitao Mi, Daniel Khashabi, and Dong Yu. The trickle-down impact of reward (in-) consistency on rlhf. *arXiv preprint arXiv:2309.16155*, 2023.

[146] Ellen Novoseller, Yibing Wei, Yanan Sui, Yisong Yue, and Joel Burdick. Dueling posterior sampling for preference-based reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, pages 1029–1038. PMLR, 2020.

[147] Yichong Xu, Ruosong Wang, Lin Yang, Aarti Singh, and Artur Dubrawski. Preference-based reinforcement learning with finite-time guarantees. *Advances in Neural Information Processing Systems*, 33:18784–18794, 2020.

[148] Aadirupa Saha, Aldo Pacchiano, and Jonathan Lee. Dueling rl: Reinforcement learning with trajectory preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 6263–6289. PMLR, 2023.

[149] Xiaoyu Chen, Han Zhong, Zhuoran Yang, Zhaoran Wang, and Liwei Wang. Human-in-the-loop: Provably efficient preference-based reinforcement learning with general function approximation. In *International Conference on Machine Learning*, pages 3773–3793. PMLR, 2022.

[150] Souradip Chakraborty, Amrit Singh Bedi, Alec Koppel, Dinesh Manocha, Huazheng Wang, Mengdi Wang, and Furong Huang. Parl: A unified framework for policy alignment in reinforcement learning. *arXiv preprint arXiv:2308.02585*, page 3, 2023.

[151] 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*, pages 43037–43067. PMLR, 2023.

[152] Wenhao Zhan, Masatoshi Uehara, Nathan Kallus, Jason D Lee, and Wen Sun. Provable offline preference-based reinforcement learning. *arXiv preprint arXiv:2305.14816*, 2023.

[153] Zihao Li, Zhuoran Yang, and Mengdi Wang. Reinforcement learning with human feedback: Learning dynamic choices via pessimism. *arXiv preprint arXiv:2305.18438*, 2023.

[154] Banghua Zhu, Michael I Jordan, and Jiantao Jiao. Iterative data smoothing: Mitigating reward overfitting and overoptimization in rlhf. *arXiv preprint arXiv:2401.16335*, 2024.

[155] Dingwen Kong and Lin Yang. Provably feedback-efficient reinforcement learning via active reward learning. *Advances in Neural Information Processing Systems*, 35:11063–11078, 2022.

[156] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on learning theory*, pages 2137–2143. PMLR, 2020.

[157] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Ren Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. In *Forty-first International Conference on Machine Learning*.

[158] 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.

[159] Heejong Bong and Alessandro Rinaldo. Generalized results for the existence and consistency of the mle in the bradley-terry-luce model, 2022.

[160] Dahyun Kim, Yungi Kim, Wonho Song, Hyeonwoo Kim, Yunsu Kim, Sanghoon Kim, and Chanjun Park. sdpo: Don't use your data all at once, 2024.

[161] William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*, 2022.

[162] Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35:21314–21328, 2022.

[163] Emily First, Markus N Rabe, Talia Ringer, and Yuriy Brun. Baldur: Whole-proof generation and repair with large language models. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1229–1241, 2023.

[164] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

[165] Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al. Rarr: Researching and revising what language models say, using language models. *arXiv preprint arXiv:2210.08726*, 2022.

[166] Shuyang Jiang, Yuhao Wang, and Yu Wang. Selfevolve: A code evolution framework via large language models. *arXiv preprint arXiv:2306.02907*, 2023.

[167] Afra Feyza Akyürek, Ekin Akyürek, Aman Madaan, Ashwin Kalyan, Peter Clark, Derry Wijaya, and Niket Tandon. Rl4f: Generating natural language feedback with reinforcement learning for repairing model outputs. *arXiv preprint arXiv:2305.08844*, 2023.

[168] Kechi Zhang, Zhuo Li, Jia Li, Ge Li, and Zhi Jin. Self-edit: Fault-aware code editor for code generation. *arXiv preprint arXiv:2305.04087*, 2023.

[169] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495*, 2023.

[170] Shrisha Bharadwaj, Yufeng Zheng, Otmar Hilliges, Michael J. Black, and Victoria Fernandez-Abrevaya. Flare: Fast learning of animatable and relightable mesh avatars. *ACM Transactions on Graphics (TOG)*, 42:1 – 15, 2023.

[171] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295*, 2023.

[172] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[173] Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*, 2023.

[174] Seonghyeon Ye, Yongrae Jo, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, and Minjoon Seo. Selfee: Iterative self-revising llm empowered by self-feedback generation. *Blog post*, 2023.

[175] Qiang Yang, Gong-Wei Song, Wei neng Chen, Ya-Hui Jia, Xudong Gao, Zhen Lu, Sang-Woon Jeon, and Jun Zhang. Random contrastive interaction for particle swarm optimization in high-dimensional environment. *IEEE Transactions on Evolutionary Computation*, 28:933–949, 2024.

[176] Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. Refiner: Reasoning feedback on intermediate representations. *arXiv preprint arXiv:2304.01904*, 2023.

[177] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*, 2023.

[178] Keshav Ramji, Young-Suk Lee, Ram'on Fernandez Astudillo, Md Arafat Sultan, Tahira Naseem, Asim Munawar, Radu Florian, and Salim Roukos. Self-refinement of language models from external proxy metrics feedback. *arXiv preprint arXiv:2403.00827*, 2024.

[179] Chenrui Zhang, Lin Liu, Chuyuan Wang, Xiao Sun, Hongyu Wang, Jinpeng Wang, and Mingchen Cai. Prefer: Prompt ensemble learning via feedback-reflect-refine. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19525–19532, 2024.

[180] Seongyun Lee, Sue Hyun Park, Yongrae Jo, and Minjoon Seo. Volcano: mitigating multimodal hallucination through self-feedback guided revision. *arXiv preprint arXiv:2311.07362*, 2023.

[181] Yangruibo Ding, Marcus J Min, Gail Kaiser, and Baishakhi Ray. Cycle: Learning to self-refine the code generation. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA1):392–418, 2024.

[182] Leonardo Ranaldi, Freitas, et al. Self-refine instruction-tuning for aligning reasoning in language models. *arXiv preprint arXiv:2405.00402*, 2024.

[183] Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b. *arXiv preprint arXiv:2406.07394*, 2024.

[184] Wenqi Zhang, Yongliang Shen, Linjuan Wu, Qiuying Peng, Jun Wang, Yueting Zhuang, and Weiming Lu. Self-contrast: Better reflection through inconsistent solving perspectives. *arXiv preprint arXiv:2401.02009*, 2024.

[185] Zhaopeng Feng, Yan Zhang, Hao Li, Wenqiang Liu, Jun Lang, Yang Feng, Jian Wu, and Zuozhu Liu. Improving llm-based machine translation with systematic self-correction. *arXiv preprint arXiv:2402.16379*, 2024.

[186] Wenda Xu, Daniel Deutsch, Mara Finkelstein, Juraj Juraska, Biao Zhang, Zhongtao Liu, William Yang Wang, Lei Li, and Markus Freitag. Llmrefine: Pinpointing and refining large language models via fine-grained actionable feedback. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1429–1445, 2024.

[187] Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Wang. Pride and prejudice: Llm amplifies self-bias in self-refinement. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15474–15492, 2024.

[188] Xin Quan, Marco Valentino, Louise A Dennis, and André Freitas. Verification and refinement of natural language explanations through llm-symbolic theorem proving. *arXiv preprint arXiv:2405.01379*, 2024.

[189] Mohamed Abdelaal, Samuel Lokadjaja, and Gilbert Engert. Gllm: Self-corrective g-code generation using large language models with user feedback. 2025.

[190] Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. *Advances in Neural Information Processing Systems*, 36, 2024.

[191] Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.

[192] Jitao Sang, Yuhang Wang, Jing Zhang, Yanxu Zhu, Chao Kong, Junhong Ye, Shuyu Wei, and Jinlin Xiao. Improving weak-to-strong generalization with scalable oversight and ensemble learning. *arXiv preprint arXiv:2402.00667*, 2024.

[193] Chujie Zheng, Ziqi Wang, Heng Ji, Minlie Huang, and Nanyun Peng. Weak-to-strong extrapolation expedites alignment. *arXiv preprint arXiv:2404.16792*, 2024.

[194] 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. In *Proc. the 41st International Conference on Machine Learning (ICML2024)*, 2024.

[195] Ziqi Wang, Le Hou, Tianjian Lu, Yuexin Wu, Yunxuan Li, Hongkun Yu, and Heng Ji. Enable lanuguage models to implicitly learn self-improvement from data. In *Proc. The Twelfth International Conference on Learning Representations (ICLR2024)*, 2024.

[196] Peter Hase, Mohit Bansal, Peter Clark, and Sarah Wiegreffe. The unreasonable effectiveness of easy training data for hard tasks. *arXiv preprint arXiv:2401.06751*, 2024.

[197] Zhiqing Sun, Longhui Yu, Yikang Shen, Weiyang Liu, Yiming Yang, Sean Welleck, and Chuang Gan. Easy-to-hard generalization: Scalable alignment beyond human supervision. *arXiv preprint arXiv:2403.09472*, 2024.

[198] Jianyuan Guo, Hanting Chen, Chengcheng Wang, Kai Han, Chang Xu, and Yunhe Wang. Vision superalignment: Weak-to-strong generalization for vision foundation models. *arXiv preprint arXiv:2402.03749*, 2024.

[199] QWen. Qwq: Reflect deeply on the boundaries of the unknown. 2023.

[200] Volodymyr Mnih. Asynchronous methods for deep reinforcement learning. *arXiv preprint arXiv:1602.01783*, 2016.

[201] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[202] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[203] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36, 2024.

[204] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.

[205] Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. Owq: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13355–13364, 2024.

[206] Hanlin Tang, Yifu Sun, Decheng Wu, Kai Liu, Jianchen Zhu, and Zhanhui Kang. Easyquant: An efficient data-free quantization algorithm for llms. *arXiv preprint arXiv:2403.02775*, 2024.

[207] Gunho Park, Baeseong Park, Minsub Kim, Sungjae Lee, Jeonghoon Kim, Beomseok Kwon, Se Jung Kwon, Byeongwook Kim, Youngjoo Lee, and Dongsoo Lee. Lut-gemm: Quantized matrix multiplication based on luts for efficient inference in large-scale generative language models. *arXiv preprint arXiv:2206.09557*, 2022.

[208] Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization. *arXiv preprint arXiv:2306.07629*, 2023.

[209] Yingsong Luo and Ling Chen. Daq: Density-aware post-training weight-only quantization for llms. *arXiv preprint arXiv:2410.12187*, 2024.

[210] Fuwen Tan, Royson Lee, Łukasz Dudziak, Shell Xu Hu, Sourav Bhattacharya, Timothy Hospedales, Georgios Tzimiropoulos, and Brais Martinez. Mobilequant: Mobile-friendly quantization for on-device language models. *arXiv preprint arXiv:2408.13933*, 2024.

[211] Yihua Shao, Siyu Liang, Zijian Ling, Minxi Yan, Haiyang Liu, Siyu Chen, Ziyang Yan, Chenyu Zhang, Haotong Qin, Michele Magno, et al. Gwq: Gradient-aware weight quantization for large language models. *arXiv preprint arXiv:2411.00850*, 2024.

[212] Stefanos Angelidis, Reinald Kim Amplayo, Yoshihiko Suhara, Xiaolan Wang, and Mirella Lapata. Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293, 2021.

[213] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183, 2022.

[214] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332, 2022.

[215] Zhihang Yuan, Lin Niu, Jiawei Liu, Wenyu Liu, Xinggang Wang, Yuzhang Shang, Guangyu Sun, Qiang Wu, Jiaxiang Wu, and Bingzhe Wu. Rptq: Reorder-based post-training quantization for large language models. *arXiv preprint arXiv:2304.01089*, 2023.

[216] Cong Guo, Jiaming Tang, Weiming Hu, Jingwen Leng, Chen Zhang, Fan Yang, Yunxin Liu, Minyi Guo, and Yuhao Zhu. Olive: Accelerating large language models via hardware-friendly outlier-victim pair quantization. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*, pages 1–15, 2023.

[217] Xiaoxia Wu, Zhewei Yao, and Yuxiong He. Zeroquant-fp: A leap forward in llms post-training w4a8 quantization using floating-point formats. *arXiv preprint arXiv:2307.09782*, 2023.

[218] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023.

[219] Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling. *arXiv preprint arXiv:2304.09145*, 2023.

[220] Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*, 2023.

[221] Xijie Huang, Zechun Liu, Shih-Yang Liu, and Kwang-Ting Cheng. Rolora: Fine-tuning rotated outlier-free llms for effective weight-activation quantization. *arXiv preprint arXiv:2407.08044*, 2024.

[222] Xuan Shen, Zhaoyang Han, Lei Lu, Zhenglun Kong, Peiyan Dong, Zhengang Li, Yanyue Xie, Chao Wu, Miriam Leeser, Pu Zhao, et al. Hotaq: Hardware oriented token adaptive quantization for large language models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2024.

[223] Lei Chen, Yuan Meng, Chen Tang, Xinzhu Ma, Jingyan Jiang, Xin Wang, Zhi Wang, and Wenwu Zhu. Q-dit: Accurate post-training quantization for diffusion transformers. *arXiv preprint arXiv:2406.17343*, 2024.

[224] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *arXiv preprint arXiv:2401.18079*, 2024.

[225] Yuxuan Yue, Zhihang Yuan, Haojie Duanmu, Sifan Zhou, Jianlong Wu, and Liqiang Nie. Wkvquant: Quantizing weight and key/value cache for large language models gains more. *arXiv preprint arXiv:2402.12065*, 2024.

[226] Shichen Dong, Wen Cheng, Jiayu Qin, and Wei Wang. Qaq: Quality adaptive quantization for llm kv cache. *arXiv preprint arXiv:2403.04643*, 2024.

[227] Yefei He, Luoming Zhang, Weijia Wu, Jing Liu, Hong Zhou, and Bohan Zhuang. Zipcache: Accurate and efficient kv cache quantization with salient token identification. *arXiv preprint arXiv:2405.14256*, 2024.

[228] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024.

[229] Wenjing Ke, Zhe Li, Dong Li, Lu Tian, and Emad Barsoum. Dl-qat: Weight-decomposed low-rank quantization-aware training for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 113–119, 2024.

[230] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR, 2023.

[231] Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Bin Sun, Xinglin Wang, Heda Wang, and Kan Li. Turning dust into gold: Distilling complex reasoning capabilities from llms by leveraging negative data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18591–18599, 2024.

[232] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[233] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.

[234] Hang Shao, Bei Liu, and Yanmin Qian. One-shot sensitivity-aware mixed sparsity pruning for large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11296–11300. IEEE, 2024.

[235] Dayou Du, Yijia Zhang, Shijie Cao, Jiaqi Guo, Ting Cao, Xiaowen Chu, and Ningyi Xu. Bitdistiller: Unleashing the potential of sub-4-bit llms via self-distillation. *arXiv preprint arXiv:2402.10631*, 2024.

[236] Haojun Xia, Zhen Zheng, Yuchao Li, Donglin Zhuang, Zhongzhu Zhou, Xiafei Qiu, Yong Li, Wei Lin, and Shuaiwen Leon Song. Flash-llm: Enabling cost-effective and highly-efficient large generative model inference with unstructured sparsity. *arXiv preprint arXiv:2309.10285*, 2023.

[237] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *arXiv preprint arXiv:2305.11627*, 2023.

[238] Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Fluctuation-based adaptive structured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10865–10873, 2024.

[239] Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*, 2024.

[240] Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.

[241] Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. Loraprune: Structured pruning meets low-rank parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 3013–3026, 2024.

[242] Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR, 2023.

[243] Mingxue Xu, Yao Lei Xu, and Danilo P Mandic. Tensorgpt: Efficient compression of the embedding layer in llms based on the tensor-train decomposition. *arXiv preprint arXiv:2307.00526*, 2023.

[244] Yixiao Li, Yifan Yu, Qingru Zhang, Chen Liang, Pengcheng He, Weizhu Chen, and Tuo Zhao. Losparse: Structured compression of large language models based on low-rank and sparse approximation. In *International Conference on Machine Learning*, pages 20336–20350. PMLR, 2023.

[245] Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. Language model compression with weighted low-rank factorization. *arXiv preprint arXiv:2207.00112*, 2022.

[246] Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. Asvd: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*, 2023.

[247] Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*, 2024.

[248] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[249] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*, 2020.

[250] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[251] Tao Lei, Junwen Bai, Siddhartha Brahma, Joshua Ainslie, Kenton Lee, Yanqi Zhou, Nan Du, Vincent Zhao, Yuexin Wu, Bo Li, et al. Conditional adapters: Parameter-efficient transfer learning with fast inference. *Advances in Neural Information Processing Systems*, 36:8152–8172, 2023.

[252] Shwai He, Run-Ze Fan, Liang Ding, Li Shen, Tianyi Zhou, and Dacheng Tao. Mera: Merging pre-trained adapters for few-shot learning. *arXiv preprint arXiv:2308.15982*, 2023.

[253] Aleksandar Petrov, Philip HS Torr, and Adel Bibi. When do prompting and prefix-tuning work? a theory of capabilities and limitations. *arXiv preprint arXiv:2310.19698*, 2023.

[254] Yangguang Li, Feng Liang, Lichen Zhao, Yufeng Cui, Wanli Ouyang, Jing Shao, Fengwei Yu, and Junjie Yan. Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. *arXiv preprint arXiv:2110.05208*, 2021.

[255] Chen Zhang, Dawei Song, Zheyu Ye, and Yan Gao. Towards the law of capacity gap in distilling language models. *arXiv preprint arXiv:2311.07052*, 2023.

[256] Fang Ma, Chen Zhang, Lei Ren, Jingang Wang, Qifan Wang, Wei Wu, Xiaojun Quan, and Dawei Song. Xprompt: Exploring the extreme of prompt tuning. *arXiv preprint arXiv:2210.04457*, 2022.

[257] Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, VG Vydiswaran, and Hao Ma. Idpg: An instance-dependent prompt generation method. *arXiv preprint arXiv:2204.04497*, 2022.

[258] Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. Spot: Better frozen model adaptation through soft prompt transfer. *arXiv preprint arXiv:2110.07904*, 2021.

[259] Lichang Chen, Heng Huang, and Minhao Cheng. Ptp: Boosting stability and performance of prompt tuning with perturbation-based regularizer. *arXiv preprint arXiv:2305.02423*, 2023.

[260] Zhengxiang Shi and Aldo Lipani. Dept: Decomposed prompt tuning for parameter-efficient fine-tuning. *arXiv preprint arXiv:2309.05173*, 2023.

[261] Joon-Young Choi, Junho Kim, Jun-Hyung Park, Wing-Lam Mok, and SangKeun Lee. Smop: Towards efficient and effective prompt tuning with sparse mixture-of-prompts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14306–14316, 2023.

[262] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

[263] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. *Advances in Neural Information Processing Systems*, 35:109–123, 2022.

[264] Ximing Lu, Faeze Brahman, Peter West, Jaehun Jung, Khyathi Chandu, Abhilasha Ravichander, Prithviraj Ammanabrolu, Liwei Jiang, Sahana Ramnath, Nouha Dziri, et al. Inference-time policy adapters (ipa): Tailoring extreme-scale lms without fine-tuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6863–6883, 2023.

[265] Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*, 2020.

[266] Yi-Lin Sung, Varun Nair, and Colin A Raffel. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34:24193–24205, 2021.

[267] Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. Composable sparse fine-tuning for cross-lingual transfer. *arXiv preprint arXiv:2110.07560*, 2021.

[268] Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 12799–12807, 2023.

[269] Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*, 2021.

[270] Danilo Vucetic, Mohammadreza Tayaranian, Maryam Ziaeefard, James J Clark, Brett H Meyer, and Warren J Gross. Efficient fine-tuning of bert models on the edge. In *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1838–1842. IEEE, 2022.

[271] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

[272] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*, 2020.

[273] Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*, 2022.

[274] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

[275] Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. *arXiv preprint arXiv:2311.11696*, 2023.

[276] Adam X Yang, Maxime Robeyns, Xi Wang, and Laurence Aitchison. Bayesian low-rank adaptation for large language models. *arXiv preprint arXiv:2308.13111*, 2023.

[277] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035, 2021.

[278] Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki Markus Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

[279] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

[280] Qiushi Huang, Tom Ko, Zhan Zhuang, Lilian Tang, and Yu Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[281] Zhan Zhuang, Yulong Zhang, Xuehao Wang, Jiangang Lu, Ying Wei, and Yu Zhang. Time-varying lora: Towards effective cross-domain fine-tuning of diffusion models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[282] Xuehao Wang, Zhan Zhuang, Feiyang Ye, and Yu Zhang. MTSAM: Multi-task fine-tuning for segment anything model. In *The Thirteenth International Conference on Learning Representations*, 2025.

[283] Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072, 2025.

[284] Shaowen Wang, Linxi Yu, and Jian Li. Lora-ga: Low-rank adaptation with gradient approximation. *Advances in Neural Information Processing Systems*, 37:54905–54931, 2025.

[285] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024.

[286] Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. Lora-pro: Are low-rank adapters properly optimized? *arXiv preprint arXiv:2407.18242*, 2024.

[287] Zhan Zhuang, Xiequn Wang, Yulong Zhang, Wei Li, Yu Zhang, and Ying Wei. Copra: A progressive lora training strategy. *arXiv preprint arXiv:2410.22911*, 2024.

[288] Qiushi Huang, Tom Ko, Lilian Tang, and Yu Zhang. ColoRA: A competitive learning approach for enhancing loRA. In *The Thirteenth International Conference on Learning Representations*, 2025.

[289] Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Wen-tau Yih, and Madian Khabsa. Unipelt: A unified framework for parameter-efficient language model tuning. *arXiv preprint arXiv:2110.07577*, 2021.

[290] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. *arXiv preprint arXiv:2206.04673*, 2022.

[291] Han Zhou, Xingchen Wan, Ivan Vulić, and Anna Korhonen. Autopeft: Automatic configuration search for parameter-efficient fine-tuning. *Transactions of the Association for Computational Linguistics*, 12:525–542, 2024.

[292] Xuehao Wang, Liyuan Wang, Binghuai Lin, and Yu Zhang. Headmap: Locating and enhancing knowledge circuits in LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025.

[293] Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933*, 2023.

[294] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[295] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[296] Shiyang Li, Jianshu Chen, Yelong Shen, Zhiyu Chen, Xinlu Zhang, Zekun Li, Hong Wang, Jing Qian, Baolin Peng, Yi Mao, et al. Explanations from large language models make small reasoners better. *arXiv preprint arXiv:2210.06726*, 2022.

[297] Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*, 2022.

[298] Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*, 2022.

[299] Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. Distilling reasoning capabilities into smaller language models. *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073, 2023.

[300] Yukun Huang, Yanda Chen, Zhou Yu, and Kathleen McKeown. In-context learning distillation: Transferring few-shot learning ability of pre-trained language models. *arXiv preprint arXiv:2212.10670*, 2022.

[301] Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pages 10421–10430. PMLR, 2023.

[302] Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *arXiv preprint arXiv:2304.14402*, 2023.

[303] Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. Scott: Self-consistent chain-of-thought distillation. *arXiv preprint arXiv:2305.01879*, 2023.

[304] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

[305] Yuxin Jiang, Chunkit Chan, Mingyang Chen, and Wei Wang. Lion: Adversarial distillation of proprietary large language models. *arXiv preprint arXiv:2305.12870*, 2023.

[306] Xuekai Zhu, Biqing Qi, Kaiyan Zhang, Xinwei Long, Zhouhan Lin, and Bowen Zhou. Pad: Program-aided distillation can teach small models reasoning better than chain-of-thought fine-tuning. *arXiv preprint arXiv:2305.13888*, 2023.

[307] Yuxuan Liu. Learning to reason with autoregressive in-context distillation. In *The Second Tiny Papers Track at ICLR 2024*, 2024.

[308] Shankar Padmanabhan, Yasumasa Onoe, Michael Zhang, Greg Durrett, and Eunsol Choi. Propagating knowledge updates to lms through distillation. *Advances in Neural Information Processing Systems*, 36:47124–47142, 2023.

[309] Zhaoyang Wang, Shaohan Huang, Yuxuan Liu, Jiahai Wang, Minghui Song, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, et al. Democratizing reasoning ability: Tailored learning from large language model. *arXiv preprint arXiv:2310.13332*, 2023.

[310] Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. Selective reflection-tuning: Student-selected data recycling for llm instruction-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 16189–16211, 2024.

[311] Ming Li, Jiuhai Chen, Lichang Chen, and Tianyi Zhou. Can llms speak for diverse people? tuning llms via debate to generate controllable controversial statements, 2024.

[312] Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. Dynabert: Dynamic bert with adaptive width and depth. *Advances in Neural Information Processing Systems*, 33:9782–9793, 2020.

[313] Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. Less is more: Task-aware layer-wise distillation for language model compression. In *International Conference on Machine Learning*, pages 20852–20867. PMLR, 2023.

[314] Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024.

[315] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*, 2023.

[316] Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. Rlcd: Reinforcement learning from contrastive distillation for language model alignment, 2024.

[317] Inar Timiryasov and Jean-Loup Tastet. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. *arXiv preprint arXiv:2308.02019*, 2023.

[318] Asaf Yehudai, Boaz Carmeli, Yosi Mass, Ofir Arviv, Nathaniel Mills, Assaf Toledo, Eyal Shnarch, and Leshem Choshen. Genie: Achieving human parity in content-grounded datasets generation. *arXiv preprint arXiv:2401.14367*, 2024.

[319] Enneng Yang, Li Shen, Zhenyi Wang, Guibing Guo, Xiaojun Chen, Xingwei Wang, and Dacheng Tao. Representation surgery for multi-task model merging. *ArXiv*, abs/2402.02705, 2024.

[320] Chen Jia. Adversarial moment-matching distillation of large language models, 2024.

[321] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[322] 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, 2023.

[323] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

[324] Yue Wang, Weishi Wang, Shafiq Joty, and Steven C. H. Hoi. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation, 2021.

[325] Tommaso Furlanello, Zachary C. Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks, 2018.

[326] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[327] Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023.

[328] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*, 2023.

[329] Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.

[330] Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023.

[331] Jing Yu Koh, Ruslan Salakhutdinov, and Daniel Fried. Grounding language models to images for multimodal inputs and outputs. In *International Conference on Machine Learning*, pages 17283–17300. PMLR, 2023.

[332] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024.

[333] K Chen, Z Zhang, W Zeng, R Zhang, F Zhu, and R Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. arxiv 2023. *arXiv preprint arXiv:2306.15195*.

[334] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26689–26699, 2024.

[335] Renjie Pi, Jiahui Gao, Shizhe Diao, Rui Pan, Hanze Dong, Jipeng Zhang, Lewei Yao, Jianhua Han, Hang Xu, Lingpeng Kong, et al. Detgpt: Detect what you need via reasoning. *arXiv preprint arXiv:2305.14167*, 2023.

[336] Yangyi Chen, Xingyao Wang, Hao Peng, and Heng Ji. Solo: A single transformer for scalable vision-language modeling. In *Transactions on Machine Learning Research*, 2024.

[337] Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? *Advances in Neural Information Processing Systems*, 37:87874–87907, 2025.

[338] Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Chao Liao, Jianchao Tan, Yadong Mu, et al. Unified language-vision pretraining in llm with dynamic discrete visual tokenization. In *International Conference on Learning Representations*, 2024.

[339] Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, et al. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*, 2024.

[340] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025.

[341] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023.

[342] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023.

[343] Feilong Chen, Minglun Han, Haozhi Zhao, Qingyang Zhang, Jing Shi, Shuang Xu, and Bo Xu. X-llm: Bootstrapping advanced large language models by treating multi-modalities as foreign languages. *arXiv preprint arXiv:2305.04160*, 2023.

[344] Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*, 2023.

[345] 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*, 1(2):3, 2023.

[346] Gongwei Chen, Leyang Shen, Rui Shao, Xiang Deng, and Liqiang Nie. Lion: Empowering multimodal large language model with dual-level visual knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26540–26550, 2024.

[347] Junyu Lu, Dixiang Zhang, Songxin Zhang, Zejian Xie, Zhuoyang Song, Cong Lin, Jiaxing Zhang, Bingyi Jing, and Pingjian Zhang. Lyrics: Boosting fine-grained language-vision alignment and comprehension via semantic-aware visual objects. *arXiv preprint arXiv:2312.05278*, 2023.

[348] Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Gadre, Shiori Sagawa, et al. Openflamingo: An open-source framework for training large autoregressive vision-language models. *arXiv preprint arXiv:2308.01390*, 2023.

[349] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Jingkang Yang, and Ziwei Liu. Otter: a multi-modal model with in-context instruction tuning. corr abs/2305.03726 (2023), 2023.

[350] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, et al. Cogvlm: Visual expert for pretrained language models. *arXiv preprint arXiv:2311.03079*, 2023.

[351] Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander Rush, Douwe Kiela, et al. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Advances in Neural Information Processing Systems*, 36, 2024.

[352] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. InternVL: Scaling up Vision Foundation Models and Aligning for Generic Visual-Linguistic Tasks. *arXiv prepring arXiv:2312.14238*, 2023.

[353] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*, 2023.

[354] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*, 2023.

[355] Ruyang Liu, Chen Li, Yixiao Ge, Ying Shan, Thomas H Li, and Ge Li. One for all: Video conversation is feasible without video instruction tuning. *arXiv preprint arXiv:2309.15785*, 2023.

[356] Peng Gao, Renrui Zhang, Chris Liu, Longtian Qiu, Siyuan Huang, Weifeng Lin, Shitian Zhao, Shijie Geng, Ziyi Lin, Peng Jin, et al. Sphinx-x: Scaling data and parameters for a family of multi-modal large language models. *arXiv preprint arXiv:2402.05935*, 2024.

[357] Yanyuan Qiao, Zheng Yu, Longteng Guo, Sihan Chen, Zijia Zhao, Mingzhen Sun, Qi Wu, and Jing Liu. Vl-mamba: Exploring state space models for multimodal learning. *arXiv preprint arXiv:2403.13600*, 2024.

[358] Minjie Zhu, Yichen Zhu, Xin Liu, Ning Liu, Zhiyuan Xu, Chaomin Shen, Yaxin Peng, Zhicai Ou, Feifei Feng, and Jian Tang. A comprehensive overhaul of multimodal assistant with small language models. *arXiv preprint arXiv:2403.06199*, 2024.

[359] Han Zhao, Min Zhang, Wei Zhao, Pengxiang Ding, Siteng Huang, and Donglin Wang. Cobra: Extending mamba to multi-modal large language model for efficient inference. *arXiv preprint arXiv:2403.14520*, 2024.

[360] QWen. Qvq: To see the world with wisdom. 2024.

[361] Anthropic. Claude 3.7 Sonnet, 2025.

[362] OpenAI. OpenAI GPT-4.5 System Card, 2025.

[363] Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. Audiopalm: A large language model that can speak and listen. *arXiv preprint arXiv:2306.12925*, 2023.

[364] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. *arXiv preprint arXiv:2305.11000*, 2023.

[365] Puyuan Peng, Po-Yao Huang, Shang-Wen Li, Abdelrahman Mohamed, and David Harwath. Voicecraft: Zero-shot speech editing and text-to-speech in the wild. *arXiv preprint arXiv:2403.16973*, 2024.

[366] Jiaming Han, Renrui Zhang, Wenqi Shao, Peng Gao, Peng Xu, Han Xiao, Kaipeng Zhang, Chris Liu, Song Wen, Ziyu Guo, et al. Imagebind-llm: Multi-modality instruction tuning. *arXiv preprint arXiv:2309.03905*, 2023.

[367] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Tushar Nagarajan, Matt Smith, Shashank Jain, Chun-Fu Yeh, Prakash Murugesan, Peyman Heidari, Yue Liu, et al. Anymal: An efficient and scalable any-modality augmented language model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1314–1332, 2024.

[368] Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. Next-gpt: Any-to-any multimodal llm. *arXiv preprint arXiv:2309.05519*, 2023.

[369] Zineng Tang, Ziyi Yang, Mahmoud Khademi, Yang Liu, Chenguang Zhu, and Mohit Bansal. Codi-2: In-context interleaved and interactive any-to-any generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27425–27434, 2024.

[370] Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. Ll3da: Visual interactive instruction tuning for omni-3d understanding reasoning and planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26428–26438, 2024.

[371] Artemis Panagopoulou, Le Xue, Ning Yu, Junnan Li, Dongxu Li, Shafiq Joty, Ran Xu, Silvio Savarese, Caiming Xiong, and Juan Carlos Niebles. X-instructblip: A framework for aligning x-modal instruction-aware representations to llms and emergent cross-modal reasoning. *arXiv preprint arXiv:2311.18799*, 2023.

[372] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[373] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19358–19369, 2023.

[374] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15180–15190, June 2023.

[375] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.

[376] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[377] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[378] Yu Zhang, Wei Han, James Qin, Yongqiang Wang, Ankur Bapna, Zhehuai Chen, Nanxin Chen, Bo Li, Vera Axelrod, Gary Wang, et al. Google usm: Scaling automatic speech recognition beyond 100 languages. *arXiv preprint arXiv:2303.01037*, 2023.

[379] Xinhao Mei, Chutong Meng, Haohe Liu, Qiuqiang Kong, Tom Ko, Chengqi Zhao, Mark D. Plumbley, Yuexian Zou, and Wenwu Wang. Wavcaps: A chatgpt-assisted weakly-labelled audio captioning dataset for audio-language multimodal research. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:3339–3354, 2024.

[380] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[381] Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Hts-at: A hierarchical token-semantic audio transformer for sound classification and detection. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 646–650. IEEE, 2022.

[382] Wouter M Kouw and Marco Loog. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*, 2018.

[383] Jingjing Li, Zhiqi Yu, Zhekai Du, Lei Zhu, and Heng Tao Shen. A comprehensive survey on source-free domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(8):5743–5762, 2024.

[384] 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.

[385] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*, 2022.

[386] Shima Imani, Liang Du, and Harsh Shrivastava. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*, 2023.

[387] Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. Continual learning for large language models: A survey. *arXiv preprint arXiv:2402.01364*, 2024.

[388] Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024.

[389] Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR, 2022.

[390] Hengrui Gu, Kaixiong Zhou, Xiaotian Han, Ninghao Liu, Ruobing Wang, and Xin Wang. Pokemqa: Programmable knowledge editing for multi-hop question answering. *arXiv preprint arXiv:2312.15194*, 2023.

[391] Qingxiu Dong, Damai Dai, Yifan Song, Jingjing Xu, Zhifang Sui, and Lei Li. Calibrating factual knowledge in pretrained language models. *arXiv preprint arXiv:2210.03329*, 2022.

[392] Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*, 2023.

[393] Evan Hernandez, Belinda Z Li, and Jacob Andreas. Inspecting and editing knowledge representations in language models. *arXiv preprint arXiv:2304.00740*, 2023.

[394] Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36, 2024.

[395] Suhang Wu, Minlong Peng, Yue Chen, Jinsong Su, and Mingming Sun. Eva-kellm: A new benchmark for evaluating knowledge editing of llms. *arXiv preprint arXiv:2308.09954*, 2023.

[396] Lang Yu, Qin Chen, Jie Zhou, and Liang He. Melo: Enhancing model editing with neuron-indexed dynamic lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19449–19457, 2024.

[397] Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.

[398] Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitriy Pyrkin, Sergei Popov, and Artem Babenko. Editable neural networks. *arXiv preprint arXiv:2004.00345*, 2020.

[399] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

[400] Peter Hase, Mona Diab, Asli Celikyilmaz, Xian Li, Zornitsa Kozareva, Veselin Stoyanov, Mohit Bansal, and Srinivasan Iyer. Methods for measuring, updating, and visualizing factual beliefs in language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2714–2731, 2023.

[401] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.

[402] Chenmien Tan, Ge Zhang, and Jie Fu. Massive editing for large language models via meta learning. *arXiv preprint arXiv:2311.04661*, 2023.

[403] Akshaj Kumar Veldanda, Shi-Xiong Zhang, Anirban Das, Supriyo Chakraborty, Stephen Rawls, Sambit Sahu, and Milind Naphade. Llm surgery: Efficient knowledge unlearning and editing in large language models. *arXiv preprint arXiv:2409.13054*, 2024.

[404] Yongchang Li, Yujin Zhu, Tao Yan, Shijian Fan, Gang Wu, and Liang Xu. Knowledge editing for large language model with knowledge neuronal ensemble. *arXiv preprint arXiv:2412.20637*, 2024.

[405] Tianci Liu, Zihan Dong, Linjun Zhang, Haoyu Wang, and Jing Gao. Mitigating heterogeneous token overfitting in llm knowledge editing. *arXiv preprint arXiv:2502.00602*, 2025.

[406] Jiaxin Qin, Zixuan Zhang, Chi Han, Manling Li, Pengfei Yu, and Heng Ji. Why does new knowledge create messy ripple effects in llms? In *Proc. The 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP2024)*, 2024.

[407] Jiateng Liu, Pengfei Yu, Yuji Zhang, Sha Li, Zixuan Zhang, and Heng Ji. Evedit: Event-based knowledge editing with deductive editing boundaries. In *Proc. The 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP2024)*, 2024.

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

[409] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, and Bin Cui. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473*, 2024.

[410] Ye Zhang, Mengran Zhu, Yulu Gong, and Rui Ding. Optimizing science question ranking through model and retrieval-augmented generation. *International Journal of Computer Science and Information Technology*, 1(1):124–130, 2023.

[411] Guangzhi Xiong, Qiao Jin, Zhiyong Lu, and Aidong Zhang. Benchmarking retrieval-augmented generation for medicine. *arXiv preprint arXiv:2402.13178*, 2024.

[412] Boyu Zhang, Hongyang Yang, Tianyu Zhou, Muhammad Ali Babar, and Xiao-Yang Liu. Enhancing financial sentiment analysis via retrieval augmented large language models. In *Proceedings of the fourth ACM international conference on AI in finance*, pages 349–356, 2023.

[413] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501, 2024.

[414] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*, 2020.

[415] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*, 2023.

[416] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[417] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

[418] Thang Nguyen, Peter Chin, and Yu-Wing Tai. Reward-rag: Enhancing rag with reward driven supervision. *arXiv preprint arXiv:2410.03780*, 2024.

[419] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2023.

[420] Sara Sarto, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Retrieval-augmented transformer for image captioning. In *Proceedings of the 19th international conference on content-based multimedia indexing*, pages 1–7, 2022.

[421] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2024.

[422] Boxin Wang, Wei Ping, Peng Xu, Lawrence McAfee, Zihan Liu, Mohammad Shoeybi, Yi Dong, Oleksii Kuchaiev, Bo Li, Chaowei Xiao, et al. Shall we pretrain autoregressive language models with retrieval? a comprehensive study. *arXiv preprint arXiv:2304.06762*, 2023.

[423] Yinghao Zhu, Changyu Ren, Shiyun Xie, Shukai Liu, Hangyuan Ji, Zixiang Wang, Tao Sun, Long He, Zhoujun Li, Xi Zhu, et al. Realm: Rag-driven enhancement of multimodal electronic health records analysis via large language models. *arXiv preprint arXiv:2402.07016*, 2024.

[424] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

[425] Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*, 2023.

[426] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*, 2023.

[427] Liang Wang, Nan Yang, and Furu Wei. Learning to retrieve in-context examples for large language models. *arXiv preprint arXiv:2307.07164*, 2023.

[428] Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, et al. Ra-dit: Retrieval-augmented dual instruction tuning. *arXiv preprint arXiv:2310.01352*, 2023.

[429] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020.

[430] Fumin Shen, Wei Liu, Shaoting Zhang, Yang Yang, and Heng Tao Shen. Learning binary codes for maximum inner product search. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4148–4156, 2015.

[431] Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*, 2024.

[432] Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's mergekit: A toolkit for merging large language models. *arXiv preprint arXiv:2403.13257*, 2024.

[433] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR, 2022.

[434] Negin Entezari, Saba A Al-Sayouri, Amirali Darvishzadeh, and Evangelos E Papalexakis. All you need is low (rank) defending against adversarial attacks on graphs. In *WSDM*, 2020.

[435] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

[436] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36, 2024.

[437] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024.

[438] Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. *arXiv preprint arXiv:2306.02561*, 2023.

[439] Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge fusion of large language models. *arXiv preprint arXiv:2401.10491*, 2024.

[440] Fanqi Wan, Longguang Zhong, Ziyi Yang, Ruijun Chen, and Xiaojun Quan. Fusechat: Knowledge fusion of chat models. *arXiv preprint arXiv:2408.07990*, 2024.

[441] Joan Puigcerver, Carlos Riquelme, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts. *arXiv preprint arXiv:2308.00951*, 2023.

[442] Mohammed Muqeeth, Haokun Liu, and Colin Raffel. Soft merging of experts with adaptive routing. *arXiv preprint arXiv:2306.03745*, 2023.

[443] Tian-Yu Liu, Aditya Golatkar, and Stefan 0 Soatto. Tangent transformers for composition, privacy and removal. *ArXiv*, abs/2307.08122, 2023.

[444] Ruochen Jin, Bojian Hou, Jiancong Xiao, Weijie J. Su, and Li Shen. Fine-tuning linear layers only is a simple yet effective way for task arithmetic. *CoRR*, abs/2407.07089, 2024.

[445] Guillermo Ortiz-Jiménez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. *ArXiv*, abs/2305.12827, 2023.

[446] Dang Nguyen, Trang Thi Thu Nguyen, Khai Nguyen, D.Q. Phung, Hung Bui, and Nhat Ho. On cross-layer alignment for model fusion of heterogeneous neural networks. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2021.

[447] Omri Avrahami, Dani Lischinski, and Ohad Fried. Gan cocktail: mixing gans without dataset access. *ArXiv*, abs/2106.03847, 2021.

[448] Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. *ArXiv*, abs/1910.05653, 2019.

[449] Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha S. Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *ArXiv*, abs/2209.04836, 2022.

[450] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. In *International Conference on Machine Learning*, 2021.

[451] Keller Jordan, Hanie Sedghi, Olga Saukh, Rahim Entezari, and Behnam Neyshabur. Repair: Renormalizing permuted activations for interpolation repair. *ArXiv*, abs/2211.08403, 2022.

[452] Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. Metagpt: Merging large language models using model exclusive task arithmetic. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

[453] Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[454] An Quang Tang, Li Shen, Yong Luo, Liang Ding, Han Hu, Bo Du, and Dacheng Tao. Concrete subspace learning based interference elimination for multi-task model fusion. *ArXiv*, abs/2312.06173, 2023.

[455] Anke Tang, Li Shen, Yong Luo, Nan Yin, Lefei Zhang, and Dacheng Tao. Merging multi-task models via weight-ensembling mixture of experts. *CoRR*, abs/2402.00433, 2024.

[456] Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. Twin-merging: Dynamic integration of modular expertise in model merging. *ArXiv*, abs/2406.15479, 2024.

[457] Tom Gunter, Zirui Wang, Chong Wang, Ruoming Pang, Andy Narayanan, Aonan Zhang, Bowen Zhang, Chen Chen, Chung-Cheng Chiu, David Qiu, et al. Apple intelligence foundation language models. *arXiv preprint arXiv:2407.21075*, 2024.

[458] Shu Hu, Yiming Ying, Xin Wang, and Siwei Lyu. Sum of ranked range loss for supervised learning. *Journal of Machine Learning Research*, 23(112):1–44, 2022.

[459] 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.

[460] Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. Understanding dataset difficulty with v-usable information. In *International Conference on Machine Learning*, 2021.

[461] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[462] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*, 2022.

[463] Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*, 2022.

[464] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

[465] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023.

[466] Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

[467] Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arxiv:2301.07597*, 2023.

[468] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm. *Company Blog of Databricks*, 2023.

[469] RyokoAI. Sharegpt52k. `https://huggingface.co/datasets/RyokoAI/ShareGPT52K`, 2023. Accessed: 2024-04-27.

[470] Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*, 2023.

[471] Yunjie Ji, Yong Deng, Yan Gong, Yiping Peng, Qiang Niu, Baochang Ma, and Xiangang Li. Belle: Be everyone's large language model engine, 2023.

[472] W Lian, B Goodson, E Pentland, et al. Openorca: An open dataset of gpt augmented flan reasoning traces, 2023.

[473] Nathan Lambert, Lewis Tunstall, Nazneen Rajani, and Tristan Thrush. Huggingface h4 stack exchange preference dataset, 2023.

[474] Teknium. Openhermes dataset, 2023a.

[475] Teknium. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants, 2023b.

[476] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

[477] Jinjie Ni, Fuzhao Xue, Kabir Jain, Mahir Hitesh Shah, Zangwei Zheng, and Yang You. Instruction in the wild: A user-based instruction dataset. `https://github.com/XueFuzhao/InstructionWild`, 2023.

[478] Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *arXiv preprint arXiv:2304.01196*, 2023.

[479] Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatGPT interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*, 2024.

[480] Jiuhai Chen, Rifaa Qadri, Yuxin Wen, Neel Jain, John Kirchenbauer, Tianyi Zhou, and Tom Goldstein. Genqa: Generating millions of instructions from a handful of prompts. *arXiv preprint arXiv:2406.10323*, 2024.

[481] Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *arXiv preprint arXiv:2406.08464*, 2024.

[482] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

[483] Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*, 2005.

[484] Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*, 2019.

[485] Jaromir Savelka, Kevin D. Ashley, Morgan A. Gray, Hannes Westermann, and Huihui Xu. Explaining legal concepts with augmented large language models (gpt-4), 2023.

[486] Hannes Westermann, Jaromir Savelka, and Karim Benyekhlef. Llmediator: Gpt-4 assisted online dispute resolution, 2023.

[487] Chengyuan Liu, Shihang Wang, Yangyang Kang, Lizhi Qing, Fubang Zhao, Changlong Sun, Kun Kuang, and Fei Wu. More than catastrophic forgetting: Integrating general capabilities for domain-specific llms, 2024.

[488] Behrooz Mansouri and Ricardo Campos. Falqu: Finding answers to legal questions, 2023.

[489] Antoine Louis, Gijs van Dijck, and Gerasimos Spanakis. Interpretable long-form legal question answering with retrieval-augmented large language models, 2023.

[490] Ilias Chalkidis, Manos Fergadiotis, Dimitrios Tsarapatsanis, Nikolaos Aletras, Ion Androutsopoulos, and Prodromos Malakasiotis. Paragraph-level rationale extraction through regularization: A case study on European court of human rights cases. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[491] Dietrich Trautmann, Alina Petrova, and Frank Schilder. Legal prompt engineering for multilingual legal judgement prediction, 2022.

[492] Xue Zongyue, Liu Huanghai, Hu Yiran, Kong Kangle, Wang Chenlu, Liu Yun, and Shen Weixing. Leec: A legal element extraction dataset with an extensive domain-specific label system, 2023.

[493] Hang Jiang, Xiajie Zhang, Robert Mahari, Daniel Kessler, Eric Ma, Tal August, Irene Li, Alex Pentland, Yoon Kim, Deb Roy, and Jad Kabbara. Leveraging large language models for learning complex legal concepts through storytelling. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024.

[494] Hai-Long Nguyen, Duc-Minh Nguyen, Tan-Minh Nguyen, Ha-Thanh Nguyen, Thi-Hai-Yen Vuong, and Ken Satoh. Enhancing legal document retrieval: A multi-phase approach with large language models, 2024.

[495] Atin Sakkeer Hussain and Anu Thomas. Large language models for judicial entity extraction: A comparative study, 2024.

[496] Aniket Deroy, Kripabandhu Ghosh, and Saptarshi Ghosh. Applicability of large language models and generative models for legal case judgement summarization, 2024.

[497] Zhi Zhou, Jiang-Xin Shi, Peng-Xiao Song, Xiao-Wen Yang, Yi-Xuan Jin, Lan-Zhe Guo, and Yu-Feng Li. Lawgpt: A chinese legal knowledge-enhanced large language model, 2024.

[498] Quzhe Huang, Mingxu Tao, Chen Zhang, Zhenwei An, Cong Jiang, Zhibin Chen, Zirui Wu, and Yansong Feng. Lawyer llama. https://github.com/AndrewZhe/lawyer-llama, 2023.

[499] Haitao Li. Lexilaw. https://github.com/CSHaitao/LexiLaw, 2023.

[500] Pierre Colombo, Telmo Pires, Malik Boudiaf, Rui Melo, Dominic Culver, Sofia Morgado, Etienne Malaboeuf, Gabriel Hautreux, Johanne Charpentier, and Michael Desa. Saullm-54b & saullm-141b: Scaling up domain adaptation for the legal domain, 2024.

[501] Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. Chatlaw. https://github.com/PKU-YuanGroup/ChatLaw, 2023.

[502] Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. Translation between molecules and natural language. In *Proc. The 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP2022)*, 2022.

[503] Carl Edwards, Aakanksha Naik, Tushar Khot, Martin Burke, Heng Ji, and Tom Hope. Synergpt: In-context learning for personalized drug synergy prediction and drug design. In *Proc. 1st Conference on Language Modeling (COLM2024)*, 2024.

[504] Henry William Sprueill, Carl Edwards, Mariefel V Olarte, Udishnu Sanyal, Heng Ji, and Sutanay Choudhury. Monte carlo thought search: Large language model querying for complex scientific reasoning in catalyst design. In *Proc. The 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP2023) Findings*, 2023.

[505] Wenjun Hou, Kaishuai Xu, Yi Cheng, Wenjie Li, and Jiang Liu. Organ: Observation-guided radiology report generation via tree reasoning, 2023.

[506] Yunsoo Kim, Jinge Wu, Yusuf Abdulle, and Honghan Wu. Medexqa: Medical question answering benchmark with multiple explanations, 2024.

[507] Wei Zhu and Xiaoling Wang. Chatmed: A chinese medical large language model. https://github.com/michael-wzhu/ChatMed, 2023.

[508] Shaoting Zhang Xiaofan Zhang, Kui Xue. Pulse: Pretrained and unified language service engine. 2023.

[509] Wei Luo and Dihong Gong. Pre-trained large language models for financial sentiment analysis, 2024.

[510] Simerjot Kaur, Charese Smiley, Akshat Gupta, Joy Sain, Dongsheng Wang, Suchetha Siddagangappa, Toyin Aguda, and Sameena Shah. Refind: Relation extraction financial dataset. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23. ACM, 2023.

[511] Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. Financebench: A new benchmark for financial question answering, 2023.

[512] Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. Fingpt: Open-source financial large language models. *arXiv preprint arXiv:2306.06031*, 2023.

[513] Wei Chen, Qiushi Wang, Zefei Long, Xianyin Zhang, Zhongtian Lu, Bingxuan Li, Siyuan Wang, Jiarong Xu, Xiang Bai, Xuanjing Huang, and Zhongyu Wei. Disc-finllm: A chinese financial large language model based on multiple experts fine-tuning. *arXiv preprint arXiv:2310.15205*, 2023.

[514] Xuanyu Zhang and Qing Yang. Xuanyuan 2.0: A large chinese financial chat model with hundreds of billions parameters. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4435–4439, 2023.

[515] Shuai Wang, Weiwen Liu, Jingxuan Chen, Weinan Gan, Xingshan Zeng, Shuai Yu, Xinlong Hao, Kun Shao, Yasheng Wang, and Ruiming Tang. Gui agents with foundation models: A comprehensive survey. *arXiv preprint arXiv:2411.04890*, 2024.

[516] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[517] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v(ision) is a generalist web agent, if grounded. In *Forty-first International Conference on Machine Learning*, 2024.

[518] Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models. *arXiv preprint arXiv:2401.13919*, 2024.

[519] Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. Assistantbench: Can web agents solve realistic and time-consuming tasks?, 2024.

[520] Revanth Gangi Reddy, Sagnik Mukherjee, Jeonghwan Kim, Zhenhailong Wang, Dilek Hakkani-Tur, and Heng Ji. Infogent: An agent-based framework for web information aggregation. *arXiv preprint arXiv:2410.19054*, 2024.

[521] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents, 2023.

[522] Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. UFO: A UI-Focused Agent for Windows OS Interaction. *arXiv preprint arXiv:2402.07939*, 2024.

[523] Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, et al. Visualagentbench: Towards large multimodal models as visual foundation agents. *arXiv preprint arXiv:2408.06327*, 2024.

[524] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *arXiv preprint arXiv:2404.07972*, 2024.

[525] Weihao Tan, Ziluo Ding, Wentao Zhang, Boyu Li, Bohan Zhou, Junpeng Yue, Haochong Xia, Jiechuan Jiang, Longtao Zheng, Xinrun Xu, et al. Towards general computer control: A multimodal agent for red dead redemption ii as a case study. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.

[526] Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *arXiv preprint arXiv:2401.16158*, 2024.

[527] Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users, 2023.

[528] Yanda Li, Chi Zhang, Wanqi Yang, Bin Fu, Pei Cheng, Xin Chen, Ling Chen, and Yunchao Wei. Appagent v2: Advanced agent for flexible mobile interactions. *arXiv preprint arXiv:2408.11824*, 2024.

[529] Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *arXiv preprint arXiv:2406.01014*, 2024.

[530] Xiao Liu, Bo Qin, Dongzhu Liang, Guang Dong, Hanyu Lai, Hanchen Zhang, Hanlin Zhao, Iat Long Iong, Jiadai Sun, Jiaqi Wang, et al. Autoglm: Autonomous foundation agents for guis. *arXiv preprint arXiv:2411.00820*, 2024.

[531] Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, et al. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*, 2024.

[532] Zhenhailong Wang, Haiyang Xu, Junyang Wang, Xi Zhang, Ming Yang, Ji Zhang, Fei Huang, and Heng Ji. Mobile-agent-e: Self-evolving mobile assistant for complex real-world tasks. In *arxiv*, 2025.

[533] OpenAI. OpenAI Codex, 2021.

[534] Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, volume 2012 of *RecSys '23*. ACM, 2023.

[535] Luke Friedman, Sameer Ahuja, David Allen, Zhenning Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, Brian Chu, Zexi Chen, and Manoj Tiwari. Leveraging large language models in conversational recommender systems, 2023.

[536] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. Llmrec: Large language models with graph augmentation for recommendation, 2024.

[537] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. Llara: Large language-recommendation assistant, 2024.

[538] An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation, 2024.

[539] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous. Tacotron: Towards end-to-end speech synthesis, 2017.

[540] Yinghao Aaron Li, Cong Han, Vinay S. Raghavan, Gavin Mischler, and Nima Mesgarani. Styletts 2: Towards human-level text-to-speech through style diffusion and adversarial training with large speech language models, 2023.

[541] Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models, 2023.

[542] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition, 2018.

[543] OpenAI. Whisper. https://github.com/openai/whisper, 2023.

[544] OpenAI. Sora, 2024.

[545] Wesley H Holliday, Matthew Mandelkern, and Cedegao E Zhang. Conditional and modal reasoning in large language models. *arXiv preprint arXiv:2401.17169*, 2024.

[546] Pengyu Cheng, Tianhao Hu, Han Xu, Zhisong Zhang, Yong Dai, Lei Han, Xiaolong Li, et al. Self-playing adversarial language game enhances llm reasoning. *Advances in Neural Information Processing Systems*, 37:126515–126543, 2025.

[547] Zifei Xu, Alexander Lan, Tristan Webb, Sayeh Sharify, Xin Wang, et al. Scaling laws for post-training quantized large language models. *arXiv preprint arXiv:2410.12119*, 2024.

[548] Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats L Richter, Quentin Anthony, Timothée Lesort, Eugene Belilovsky, and Irina Rish. Simple and scalable strategies to continually pre-train large language models. *arXiv preprint arXiv:2403.08763*, 2024.

[549] Ziheng Jiang, Haibin Lin, Yinmin Zhong, Qi Huang, Yangrui Chen, Zhi Zhang, Yanghua Peng, Xiang Li, Cong Xie, Shibiao Nong, et al. {MegaScale}: Scaling large language model training to more than 10,000 {GPUs}. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, pages 745–760, 2024.

[550] Adam Dahlgren Lindström, Leila Methnani, Lea Krause, Petter Ericson, Íñigo Martínez de Rituerto de Troya, Dimitri Coelho Mollo, and Roel Dobbe. Ai alignment through reinforcement learning from human feedback? contradictions and limitations. *arXiv preprint arXiv:2406.18346*, 2024.

[551] Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, et al. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*, 3, 2024.

[552] Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025.