Towards Efficient and Effective Alignment of Large Language Models

by

Yuxin Jiang

A Thesis Submitted to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in Individualized Interdisciplinary Program (Data Science and Analytics)

June 2025, Hong Kong

Towards Efficient and Effective Alignment of Large Language Models

by

Yuxin Jiang

Division of Emerging Interdisciplinary Areas

The Hong Kong University of Science and Technology

ABSTRACT

Large language models (LLMs) exhibit remarkable capabilities across diverse tasks, yet aligning them efficiently and effectively with human expectations remains a critical challenge. This thesis advances LLM alignment by introducing novel methodologies in data collection, training, and evaluation.

We first address alignment data collection. Existing approaches rely heavily on manually curated datasets or proprietary models. To overcome these limitations, we propose Lion, an adversarial distillation framework that iteratively refines training data by identifying and generating challenging instructions, enabling state-of-the-art zero-shot reasoning. Additionally, we introduce Web Reconstruction (WebR), a fully automated framework that synthesizes instruction-tuning data directly from raw web documents, significantly improving data diversity and scalability over existing synthetic data methods.

Next, we enhance alignment training through novel optimization techniques. We develop Learning to Edit (LTE), a framework that enables LLMs to efficiently integrate new knowledge while preserving existing information. LTE leverages meta-learning to improve both real-time and batch knowledge updates. Furthermore, we introduce Bridging

and Modeling Correlations (BMC), a refinement of Direct Preference Optimization (DPO) that explicitly captures token-level correlations in preference data, leading to superior alignment across QA and mathematical reasoning tasks.

Finally, we tackle the challenge of evaluating alignment. Existing benchmarks emphasize response quality but overlook adherence to specific constraints. To bridge this gap, we introduce `FollowBench`, a multi-level, fine-grained benchmark assessing LLMs' ability to follow complex constraints across diverse instruction types. Our results expose key weaknesses in current models' constraint adherence, offering insights for future improvements.

This thesis makes fundamental contributions to LLM alignment by pioneering novel strategies for data synthesis, training optimization, and evaluation. These advancements enhance efficiency, adaptability, and rigor, paving the way for safer and more controllable AI systems.

Authorization

I hereby declare that I am the sole author of the thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_____

Yuxin Jiang

3 June 2025

Towards Efficient and Effective Alignment of Large Language Models

by

Yuxin Jiang

This is to certify that I have examined the above Ph.D. thesis
and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by
the thesis examination committee have been made.

Prof. Wei Wang, Thesis Supervisor

Prof. Jiaqiang Huang, Thesis Co-Supervisor

Prof. Huamin Qu, Head of Division

Division of Emerging Interdisciplinary Areas

3 June 2025

ACKNOWLEDGMENTS

First and foremost, I extend my deepest gratitude to my supervisor, Prof. Wei Wang, for his invaluable guidance, generous support in experimental resources and funding, and unwavering understanding and encouragement throughout my four-year Ph.D. journey. His insatiable thirst for knowledge, pursuit of academic excellence, and rigorous scholarly attitude have profoundly influenced me and will continue to inspire me for a lifetime. I am also sincerely grateful to my co-supervisor, Prof. Jiaqiang Huang, for his support and insightful discussions. Additionally, I would like to express my heartfelt appreciation to my former supervisor, Prof. Fangzhen Lin, who first introduced me to the field of Natural Language Processing and set me on the path of scientific research.

I would like to thank all of my thesis defense committee, Prof. Yutao Yue, Prof. Cuiyun Gao, Prof. Xiaowen Chu, Prof. Zhijiang Guo, and Prof. Xuming Hu, for their valuable time and insightful feedback on my thesis.

Throughout my Ph.D. journey, I have experienced not only the joy of discovery but also the fulfillment of collaborating with brilliant and like-minded friends. I am deeply grateful to my collaborators, in order of our acquaintance: Dr. Ziyi Shou, Dr. Linhan Zhang, Mr. Chunkit Chan, Mr. Mingyang Chen, Mr. Bo Huang, Dr. Yufei Wang, Dr. Xingshan Zeng, Dr. Wai-Chung Kwan, and Mr. Qiyuan Zhang. Every publication is a testament to our collective wisdom, dedication, and relentless efforts. Over the past four years, frequent travels between Hong Kong, Guangzhou, and Shenzhen have been made more meaningful by the companionship of my dear friends—Mr. Hao Wu, Mr. Fengming Zhu, Mr. Biqing Fang, Mr. Zhihao Li, Mr. Mingyu Yang, Mr. Yinan Fan, Mr. Zhongkun Liao, Mr. Rongxin Liu, Mr. Zheng Wei, Mr. Zhengping Chen, Mr. Wentao Pan, and many others. Their support during the pandemic, our engaging discussions, and the moments of laughter we shared have been invaluable.

Lastly, I owe my deepest gratitude to my parents, Mr. Xubo Jiang and Ms. Qifang Zhao, for their unconditional love and unwavering support, and to my grandparents, Mr. Chunzhong Jiang and Ms. Honglian Yu, for their encouragement and belief in me. Above all, I am profoundly thankful to my girlfriend and future wife, Ms. Ying Lin, for her love,

companionship, and steadfast encouragement. Because of you all, my journey has been filled with warmth and happiness.

May we continue to chase our dreams without regret, cherish our youth, and embrace a future filled with success and joy.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Thesis Introduction

The recent advances in large language models (LLMs) have demonstrated extraordinary breakthroughs in various real-world applications. These models, typically based on transformer architectures and comprising tens to hundreds of billions of parameters, are trained on vast datasets sourced from the web using an autoregressive learning paradigm. Prominent examples include PaLM [33], LLaMA [170], and GPT-4 [135]. Compared to earlier, smaller models [32, 46], LLMs exhibit two defining characteristics: (1) the *scaling law* [86], which demonstrates systematic performance gains with increased model size, and (2) the *emergence capabilities* [185]—such as in-context learning [48], instruction following [137], and complex reasoning [186]—once a critical scale is surpassed. These advancements have led to transformative impacts across sectors such as finance [108], law [94], and healthcare [37], reshaping the way problems are approached and solved.

Nonetheless, despite their capabilities, LLMs also come with significant limitations. Due to their training on large-scale, internet-derived datasets, they may absorb harmful or biased information, resulting in concerns such as misinformation [13], unfair social representations [149], and toxic or exclusionary outputs [187]. Furthermore, researchers have identified two troubling risk patterns: (1) *inverse scaling*, where specific issues may worsen as model size increases [118]; and (2) *emergent risks*, where new or intensified risks materialize in larger models [185], challenging existing mitigation strategies.

To address these growing concerns, a body of research has focused on developing **alignment** techniques to better steer LLM behavior in accordance with human instructions, goals, and ethical standards [137, 96, 148]. The concept of alignment can be traced back to Norbert Wiener's early warnings: "*We had better be quite sure that the purpose put into the machine is the purpose which we really desire*" [188]. In today's AI landscape, alignment generally refers to the principle that an artificial agent $\mathcal{A}$ should act in ways that

reflect the goals and intentions of a human agent $\mathcal{H}$—namely, "$\mathcal{A}$ **is trying to do what** $\mathcal{H}$ **wants it to do**" [202]. A formal treatment of this concept in the context of LLMs will be presented in §2.1.2.

The alignment process for LLMs generally unfolds across three foundational phases: (1) **Alignment Data Collection**, (2) **Alignment Training**, and (3) **Alignment Evaluation**, as illustrated in Figure 1.1. Despite recent advances, each of these stages continues to face significant challenges in terms of both **efficiency** (i.e., the cost, scalability, and speed of the process) and **efficacy** (i.e., the quality and impact of the resulting alignment). Specifically,

- **Alignment Data Collection** (§2.2)
  - *Efficiency challenges*: Collecting high-quality human feedback or preference data at scale remains expensive and time-consuming [207]. Filtering and curating alignment-specific data from large corpora also demands substantial computational resources.
  - *Efficacy challenges*: The collected data often lacks diversity, contains annotation noise, or fails to capture nuanced human preferences, limiting its ability to guide meaningful alignment [176].
- **Alignment Training** (§2.3)
  - *Efficiency challenges*: Fine-tuning large models with human feedback requires extensive computational resources, especially when conducted iteratively or with large-scale preference data [87].
  - *Efficacy challenges*: Alignment methods can suffer from over-optimization (e.g., reward hacking) [140], poor generalization to unseen instructions, and instability during training, all of which compromise the robustness of the aligned model.
- **Alignment Evaluation** (§2.4)
  - *Efficiency challenges*: Manual evaluation by human annotators is costly and slow, while automatic metrics often fail to generalize across tasks or correlate with human judgment [65].

– *Efficacy challenges*: Existing evaluation protocols are typically narrow in scope, focusing on surface-level correctness while neglecting deeper aspects such as fine-grained constraints, ethical alignment, and long-range coherence.

In response to these ongoing challenges, this thesis focuses on advancing both the efficiency and effectiveness of LLM alignment, with particular emphasis on the processes of data collection, model training, and evaluation. Through a comprehensive examination of each stage, we diagnose prevailing limitations and introduce novel approaches for improvement. Additionally, we assess the alignment capabilities of LLMs by analyzing their ability to follow nuanced and detailed instructions, a core component of successful alignment.

The first line of work in this thesis focuses on enhancing *data collection* for aligning LLMs. Previous studies often overlooked the possibility of incorporating any "feedback"–i.e., identifying challenging instructions where the model's performance falls short–to boost the model's proficiency iteratively. To address this, we propose a novel adversarial distillation framework aimed at more efficient alignment data generation. Leveraging the versatile role adaptability of LLMs, we prompt the teacher model to identify "hard" instructions and generate new "hard" instructions for the student model, creating a three-stage adversarial loop of imitation, discrimination, and generation. By applying this adversarial framework, we successfully transfer knowledge from ChatGPT to a student model (named Lion), using a mere 70k training data. Besides, our trained model surpasses conventional state-of-the-art (SOTA) instruction-tuned models like Vicuna-13B on challenging zero-shot reasoning benchmarks.

Furthermore, while existing automatic data synthesis methods alleviate the burden of manual curation, they often rely heavily on either the quality of seed data or strong assumptions about the structure and content of web documents. To tackle these challenges, we propose **Web Reconstruction** (WebR), a fully automated framework for synthesizing high-quality instruction-tuning (IT) data directly from raw web documents with minimal assumptions. Leveraging the inherent diversity of raw web content, we conceptualize *web reconstruction* as an instruction-tuning data synthesis task via a novel dual-perspective paradigm—*Web as Instruction* and *Web as Response*—where each web document is designated as either an instruction or a response to trigger the reconstruction process. Compre-

hensive experiments show that datasets generated by WebR outperform state-of-the-art baselines by up to 16.65% across four instruction-following benchmarks. Notably, WebR demonstrates superior compatibility, data efficiency, and scalability, enabling enhanced domain adaptation with minimal effort.

Beyond data synthesis, we explore *alignment training* in the context of knowledge editing—a task focused on updating specific factual knowledge in LLMs without degrading their overall performance. Most existing methods rely on memorization, which hinders models from integrating updated knowledge with existing information when answering questions. To overcome this limitation, we introduce a novel framework named **Learning to Edit** (LTE), designed to enable LLMs to effectively apply new knowledge into given queries. Drawing inspiration from the principle of "*teaching how to fish*," LTE is structured into two distinct stages. The first is the Alignment Stage, where LLMs are fine-tuned using a carefully constructed parallel corpus, ensuring that the model learns to make accurate, context-relevant modifications while maintaining unrelated information and overall linguistic quality. The second is the Inference Stage, which leverages a retrieval-augmented strategy to support efficient and large-scale application of knowledge edits. Extensive evaluations on four widely-used benchmarks and two LLM backbones, against seven competitive baselines, confirm that LTE achieves state-of-the-art results. It offers strong performance in editing accuracy, robustness in both batch and sequential settings, minimal impact on general capabilities, and fast inference times.

We also advance the training of LLMs by improving Direct preference optimization (DPO), a widely adopted offline preference optimization algorithm. In conventional DPO, the generation of the winning response and the losing response within pairwise data are typically isolated, leading to weak correlations between them as well as suboptimal alignment performance. To address this issue, we propose an effective framework for Bridging and Modeling Correlations in pairwise data, named **BMC**. Firstly, we increase the consistency and informativeness of the pairwise preference signals through *targeted modifications*, synthesizing a pseudo-winning response by improving the losing response with the winning response as a reference. Secondly, we identify that DPO alone is insufficient to model these correlations and capture nuanced variations. Therefore, we propose learning token-level correlations by *dynamically* leveraging the policy model's confidence during training.

Comprehensive experiments on QA, math, and instruction-following tasks demonstrate the effectiveness of our approach, significantly surpassing competitive baselines, including DPO. Additionally, our in-depth quantitative analysis reveals the reasons behind our method's superior performance over DPO and showcases its versatility to other DPO variants.

Finally, we provide *evaluations* on the alignment of LLMs by analyzing their ability to follow nuanced and detailed instructions, a critical yet underexplored aspect of alignment. Existing benchmarks primarily focus on evaluating pure response quality, rather than assessing whether the response follows constraints stated in the instruction. To fill this research gap, we propose `FollowBench`, a **Multi-level Fine-grained Constraints Following Benchmark** for LLMs. `FollowBench` comprehensively includes five different types (i.e., Content, Situation, Style, Format, and Example) of fine-grained constraints. To enable a precise constraint following estimation on diverse difficulties, we introduce a *Multi-level* mechanism that incrementally adds a single constraint to the initial instruction at each increased level. To assess whether LLMs' outputs have satisfied every individual constraint, we propose to prompt strong LLMs with constraint-evolution paths to handle challenging open-ended instructions. By evaluating 13 closed-source and open-source popular LLMs on `FollowBench`, we highlight the weaknesses of LLMs in instruction following and point towards potential avenues for future work.

In summary, this thesis addresses the critical need for *efficient and effective alignment* of LLMs by tackling three foundational components: data collection, training, and evaluation. We have demonstrated the pervasive issues of low-quality or assumption-heavy synthetic data, ineffective training methods that ignore correlation structures or generalization needs, and insufficient benchmarks for measuring fine-grained instruction adherence. Through extensive research and rigorous experimentation, we introduce novel frameworks—Adversarial Distillation, WebR, LTE, BMC, and `FollowBench` —that significantly improve alignment across all stages. These contributions offer both theoretical insights and practical tools to advance the field of LLM alignment.

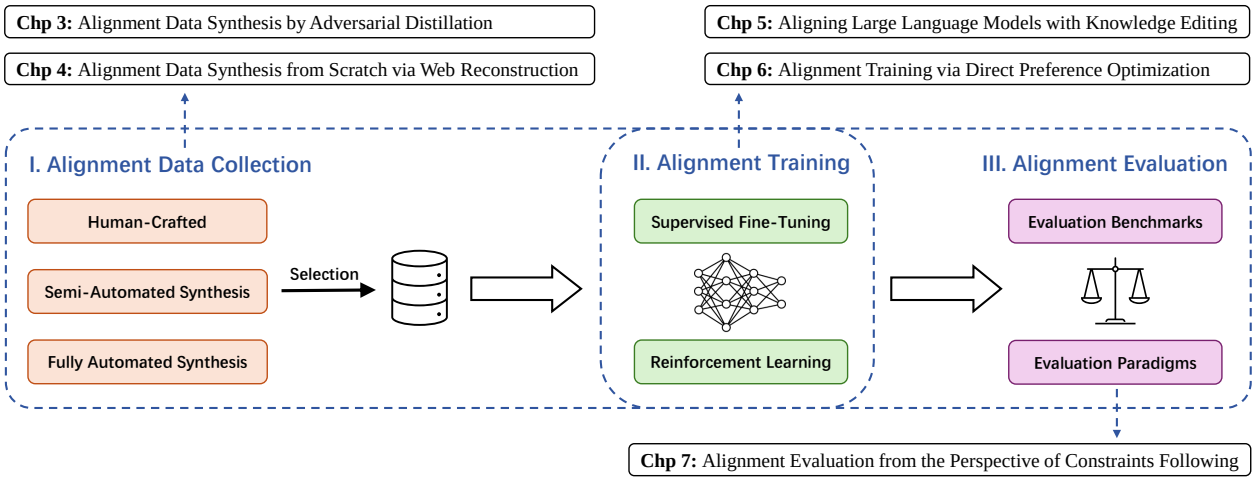


Figure 1.1: Roadmap of the thesis.

1.2 Thesis Organization

As illustrated in Figure 1.1, this thesis is structured as follows. Chapter 2 commences with the definition of LLM alignment, then provides an overview of the relevant literature on alignment data collection, training, and evaluation. In Chapter 3, we introduce Lion, an adversarial distillation framework that iteratively refines alignment training data by identifying and generating challenging instructions, enabling state-of-the-art zero-shot reasoning. Chapter 4 presents Web Reconstruction (WebR), a fully automated framework that synthesizes alignment training data directly from raw web documents, significantly enhancing data diversity and scalability compared to existing synthetic methods. In Chapter 5, we propose Learning to Edit (LTE), a framework designed for efficient and effective knowledge editing in LLMs. Chapter 6 introduces Bridging and Modeling Correlations (BMC), a refinement of Direct Preference Optimization (DPO) that explicitly captures token-level correlations in preference data, achieving superior alignment in QA and mathematical reasoning tasks. Chapter 7 presents `FollowBench`, a Multi-level Fine-grained Constraints Following Benchmark, designed to evaluate instruction following—the key attribution of alignment—in LLMs. Finally, Chapter 8 concludes the thesis by summarizing our contributions and suggesting directions for future research.

CHAPTER 2

BACKGROUND

2.1 Definition of LLM Alignment

In this section, we begin by briefly introducing the concept and evolution of LLMs in §2.1.1, and then proceed to formalize the alignment of LLMs in §2.1.2.

2.1.1 Large Language Models

The development of language models (LMs) has progressed through several key phases, starting from statistical approaches (e.g., Statistical Language Models, or SLMs) [144], followed by the emergence of neural language models (NLMs) [32], and culminating in the era of pre-trained language models (PLMs) such as BERT and Roberta [46, 115]. Building upon these foundations, **Large Language Models** (LLMs) have recently emerged as a dominant paradigm in natural language processing (NLP). These models are typically pre-trained on massive datasets using carefully designed objectives, and in some cases, incorporate multimodal data such as image-text pairs to enhance their representational power [49, 116].

LLMs differ significantly from their smaller predecessors, not only in scale but also in capability. A key property is the *scaling law*, which reveals that performance on a wide range of tasks tends to improve predictably as the number of parameters and training data increase [86]. Even more intriguingly, LLMs exhibit *emergent behaviors*—novel abilities that materialize only once the model surpasses a certain size threshold [185]. These include, but are not limited to, in-context learning [48], instruction following [137], and the ability to perform multi-step reasoning across diverse tasks and domains [186]. Such capabilities mark a significant shift in how AI systems are applied to real-world problems, expanding their utility well beyond traditional NLP settings and into broader fields such as education, healthcare, and scientific discovery.

2.1.2 Alignment of Large Language Models

The notion of alignment dates back to Norbert Wiener's cautionary insight, "*We had better be quite sure that the purpose put into the machine is the purpose which we really desire*" [188]. In modern AI discourse, alignment is often described as ensuring that the behavior of an artificial agent $\mathcal{A}$ aligns with the intentions of a human agent $\mathcal{H}$—formally, "$\mathcal{A}$ **is trying to do what** $\mathcal{H}$ **wants it to do**"[202]. Drawing from the principle of *value alignment* in reinforcement learning (RL) [69], we adopt a utility-based framework to define LLM alignment [179].

Formalization of LLM Alignment. Let $\mathcal{H}$ and $\mathcal{A}$ represent two intelligent agents with respective utility functions $U_{\mathcal{H}}(\boldsymbol{y})$ and $U_{\mathcal{A}}(\boldsymbol{y})$, where $\boldsymbol{y} \in \mathcal{Y}$ denotes an action and $U : \mathcal{Y} \to \mathbb{R}$. We consider $\mathcal{A}$ aligned with $\mathcal{H}$ over domain $\mathcal{Y}$ if the preference ordering of $\mathcal{H}$ is preserved by $\mathcal{A}$ —that is, for any $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{Y}$, whenever $U_{\mathcal{H}}(\boldsymbol{y}_1) > U_{\mathcal{H}}(\boldsymbol{y}_2)$, it follows that $U_{\mathcal{A}}(\boldsymbol{y}_1) > U_{\mathcal{A}}(\boldsymbol{y}_2)$. Misalignment can be quantitatively assessed using:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{y}_1,\boldsymbol{y}_2} \left| [U_{\mathcal{H}}(\boldsymbol{y}_1) - U_{\mathcal{H}}(\boldsymbol{y}_2)] - [U_{\mathcal{A}}(\boldsymbol{y}_1) - U_{\mathcal{A}}(\boldsymbol{y}_2)] \right|, \tag{2.1}$$

A more stringent criterion assumes $U_{\mathcal{H}}(\boldsymbol{y}) = U_{\mathcal{A}}(\boldsymbol{y})$ for all actions, yielding a simplified discrepancy measure:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{y}} \left| U_{\mathcal{H}}(\boldsymbol{y}) - U_{\mathcal{A}}(\boldsymbol{y}) \right|. \tag{2.2}$$

Approaches aimed at minimizing the alignment gap (Eq. 2.1) are generally grouped into two main categories: **Value Learning** and **Imitation Learning** [97].

Value Learning. This approach centers around constructing a *reward* function that encapsulates human preferences or goals. One can formalize the objective as follows:

$$\phi^* = \arg\min_{\phi} \mathbb{E}_{\boldsymbol{y},r^* \sim \mathcal{D}(\boldsymbol{y},r^*)} \left[(r^* - R_\phi(\boldsymbol{y}))^2 \right], \tag{2.3}$$

where $\mathcal{D}$ is a dataset of actions $\boldsymbol{y}$ and corresponding ground-truth rewards r^*, and R_ϕ is the reward model parameterized by ϕ. In scenarios where only the optimal action $\boldsymbol{y}^*$ is observed (instead of its reward), the model can be trained to rank $\boldsymbol{y}^*$ higher by minimizing:

$$\mathbb{E}_{\boldsymbol{y}^* \sim \mathcal{D}(\boldsymbol{y}^*), \boldsymbol{y} \sim p(\boldsymbol{y})} \left[\max(0, \alpha + R_\phi(\boldsymbol{y}) - R_\phi(\boldsymbol{y}^*)) \right], \tag{2.4}$$

with α as a margin hyperparameter and $p(\boldsymbol{y})$ the sampling distribution.

Classic RL algorithms like Deep Q-Networks [79], as well as frameworks such as Inverse Reinforcement Learning [132] and Preference Modeling [138], naturally fall under this formulation. Once the optimal reward function R_{ϕ^*} is identified, it can guide the agent's behavior through standard reinforcement learning paradigms, as will be elaborated in §2.3.2.

Imitation Learning. Rather than explicitly modeling a reward signal, imitation learning methods encourage the model to replicate ideal behaviors, implicitly capturing desired values [169]. Let $\pi(\boldsymbol{y})$ denote the target policy and $\pi_\theta(\boldsymbol{y})$ the agent's learned policy, parameterized by θ. The goal is to minimize a divergence metric $\mathbb{D}_f$ between the two distributions:

$$\theta^* = \arg\min_\theta \mathbb{D}_f[\pi(\boldsymbol{y}) \,\|\, \pi_\theta(\boldsymbol{y})], \tag{2.5}$$

where $\pi(\boldsymbol{y})$ is derived empirically from observed demonstrations. When $\mathbb{D}_f$ is the KL divergence, this reduces to the familiar *cross-entropy loss*, training the agent to mimic behavior aligned with human preferences. A well-known application of this principle is Supervised Fine-Tuning (SFT).

In § 2.3, we will explore how LLM alignment training maps onto the dichotomy between value-based and imitation-based strategies.

2.2 Alignment Data Collection

Before exploring *how to align LLMs*, we begin by examining *what they should be aligned with*. Effective alignment with human expectations requires high-quality training data that genuinely captures human values and preferences—such as those encapsulated by the *Helpful, Honest, and Harmless* (HHH) principle [4]. Each data point $I_k \in \mathcal{D}$, referred to as an **instruction** $I_k = (x_k, y_k)$, consists of an instructional prompt x_k and its corresponding model response y_k.

In this section, we first present methods for synthesizing high-quality alignment data, which broadly fall into three categories: (1) Human-Crafted Methods, (2) Semi-Automated

Synthetic Methods, and (3) Fully Automated Synthetic Methods. We then introduce advanced data selection techniques designed to retain only the most relevant and reliable instruction–response pairs in the final dataset.

2.2.1 Human-Crafted Method

This category involves enlisting human experts to design instructional data, as exemplified by datasets like SUPER-NI [181], OpenAssistant [91], and DOLLY [41]. For instance, the DOLLY dataset comprises 15,000 instruction-response pairs contributed by Databricks employees through a structured crowd-sourcing process. Contributors were guided to compose prompts and responses across eight instruction types, including seven from the taxonomy in [137] and one open-ended category. Notably, they were prohibited from referencing online materials or outputs from generative AI models. Although this method ensures high-quality instructional data, it is inherently limited in scale due to the labor-intensive and costly nature of manual data generation.

In contrast, alternative strategies like ShareGPT [31] and WildChat [209] gather human-authored instructions by extracting interaction logs from large language model users. This passive collection approach enables large-scale acquisition of diverse, natural prompts that often lead to informative and relevant responses. Additionally, public platforms such as Stack Overflow[1], Quora[2], and Zhihu[3], along with extensive user-generated repositories like Wikipedia[4], serve as valuable sources for human-generated instruction data. Nevertheless, mining such data carries the risk of introducing inappropriate or harmful content [209].

2.2.2 Semi-Automated Synthetic Method

The semi-automated approach for producing synthetic instruction-tuning datasets leverages LLMs to expand a limited amount of manually curated seed data through in-context

[1] https://stackoverflow.com/

[2] https://www.quora.com/

[3] https://www.zhihu.com/

[4] https://en.wikipedia.org/

learning mechanisms. A prominent example is the Self-Instruct framework [180], which utilizes ChatGPT's ability to generalize from provided examples to produce a broad range of instructions across multiple domains and task formats. This process includes iterative quality filtering to ensure the generated content meets specific standards, continuing until a sufficient volume of data is collected. Subsequent works such as Alpaca [164] and Evol-Instruct [194] build upon this strategy, aiming to improve the diversity, sophistication, and quality of the synthetic instructions. Despite their scalability, these methods often inherit limitations in data variety from the initial seed instructions [101].

2.2.3 Fully Automated Synthetic Method

The fully automated approach leverages LLMs to generate alignment data entirely from scratch. One representative strategy removes human supervision by creating data directly from web-sourced content. For example, WebInstruct [203] derives instruction-response datasets by mining question-answer (QA) pairs from web documents. However, this method is constrained by the necessity for QA pairs to be explicitly embedded in the source material, which is not always the case. Another line of work, such as Instruction Backtranslation [105, 133, 26], considers raw web text as candidate responses and relies on LLMs to infer the corresponding latent instructions. Still, these documents often include off-topic content or poorly suited phrasing, reducing their effectiveness as high-quality response material. In contrast, Magpie [196] bypasses the limitations of raw web text by directly prompting aligned LLMs using predefined templates to jointly produce both instructions and their associated responses, taking advantage of the LLMs' auto-regressive generation capabilities.

2.2.4 Approach to Data Selection

To construct a high-quality instruction–response dataset, a filtering mechanism is applied to remove low-quality entries. This is achieved using a scoring function $s(\cdot)$, which assigns a quality score to each instruction–response pair $I_k = (x_k, y_k)$. The final, cleaned dataset $\mathcal{D}'$ is formed by selecting only those pairs that surpass a predefined threshold τ:

$$\mathcal{D}' = \{(x_k, y_k) \in \mathcal{D} \mid s(x_k, y_k) \geqslant \tau\}. \tag{2.6}$$

One common choice for $s(\cdot)$ is the Instruction Following Difficulty (IFD) score [102], which reflects how well an instruction contributes to generating the corresponding response. This score is computed as:

$$s_\theta(x_k, y_k) = \frac{\sum_{t=1}^{T} \log p(y_k^t \mid x_k, y_k^{<t}; \theta)}{\sum_{t=1}^{T} \log p(y_k^t \mid y_k^{<t}; \theta)}. \tag{2.7}$$

This formulation provides a normalized comparison between the probability of the response being generated with versus without the instruction, offering a quantitative estimate of the instruction's utility. Pairs that fall below the IFD threshold are omitted, yielding a filtered dataset $\mathcal{D}'$. Alternative strategies for data selection make use of auxiliary models. For example, Instruction Mining [19] employs statistical regression techniques and multiple trained models to assess candidate data points. Similarly, ALPAGASUS [24] adopts a pre-trained LLM such as ChatGPT to evaluate the quality of samples.

In addition to quality, many selection frameworks also take into account the **diversity** and **importance** of samples. With respect to diversity, some studies [22, 17, 2] aim to ensure coverage across a broad range of tasks and language expressions, either by maximizing individual sample uniqueness (e.g., in terms of lexical or semantic features) or by ensuring the dataset spans a large representation space. Samples from underrepresented task types are often prioritized.

In terms of importance, research efforts such as [200, 191] attempt to identify which instruction–response pairs are most essential to include in the training set. Since large models already internalize extensive world knowledge during pretraining, they can often solve standard tasks without further tuning. Consequently, training efforts should focus on more challenging cases, where explicit alignment remains necessary. Selected examples thus serve as critical supplements to enhance the model's ability to follow complex instructions.

2.3 Alignment Training

Once alignment data has been gathered from diverse sources, the next step involves leveraging this data to adapt pre-trained LLMs so that their behavior becomes more consistent with human intentions.

2.3.1 Supervised Fine-Tuning

A widely adopted approach for alignment is Supervised Fine-Tuning (SFT), which falls under the umbrella of *Imitation Learning*, as outlined in Section 2.1.2. In this framework, given an input prompt x, the model is trained to generate the reference output y by minimizing the cross-entropy loss over the target tokens:

$$\mathcal{L}^{\text{SFT}}(\theta) = -\sum_{t=1}^{T} \log p(y^t \mid x, y^{<t}; \theta). \tag{2.8}$$

Through this process, LLMs are encouraged to produce coherent and contextually appropriate completions that align with the semantics of the input prompt. Despite its simplicity and effectiveness, SFT has an inherent limitation—it only exposes the model to optimal responses and lacks explicit feedback on less desirable outputs. Nevertheless, SFT-trained models or their objective functions are frequently incorporated into preference-based training pipelines to enhance stability and serve as a form of regularization (see Section 2.3.2).

In the conventional SFT paradigm, all parameters of the language model are fine-tuned, which can become prohibitively expensive in terms of computational resources and memory usage, especially as model scales exceed tens of billions of parameters. To mitigate these costs, a class of techniques known as parameter-efficient fine-tuning (PEFT) has emerged. These methods, including LoRA [77], Prefix Tuning [106], and Adapter modules [76], introduce a small number of trainable components—such as additional prompts or lightweight modules—while keeping the majority of the original model weights unchanged. This results in a substantial reduction in memory usage without compromising performance.

2.3.2 Reinforcement Learning

From a methodological standpoint, the application of RL in aligning LLMs generally unfolds in three main stages:

- **SFT:** The process typically starts by adapting a pre-trained language model using

supervised learning on curated, high-quality datasets. This step establishes a foundational level of adherence to expected formats and stylistic conventions.

- **Reward Model Training:** Once the model is fine-tuned, it is used to generate responses that are then annotated with human preferences. Several techniques exist for modeling these preferences, among which the Bradley-Terry (BT) model [14] is frequently employed. Alternatively, the Plackett-Luce model [145] can be used, particularly when multiple responses are ranked simultaneously. The reward model is trained to approximate these preference annotations, effectively learning a scalar-valued reward function that evaluates the quality of responses.

- **RL optimization:** As discussed earlier in §2.1.2, after identifying the optimal reward function R_{ϕ^*}, it is utilized to guide the learning of the language model through feedback-based optimization.

The objective during this optimization phase is formulated as follows:

$$\pi_\theta^*(y \mid x) = \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)} \Big[R_{\phi^*}(x, y) - \beta \mathbb{D}_{KL} \left[\pi_\theta(y \mid x) \,\|\, \pi_{ref}(y \mid x) \right] \Big]. \tag{2.9}$$

This formulation captures two key objectives: (1) encouraging the generation of high-reward responses, and (2) constraining the updated policy $\pi_\theta(y \mid x)$ to remain close to the behavior of the supervised baseline $\pi_{ref}(y \mid x)$.

In the subsequent section, we introduce several prominent RL methods designed to optimize the objective in Eq. 2.9, as illustrated in Figure 2.1.

Proximal Policy Optimization (PPO) [154] is a widely adopted reinforcement learning technique for aligning LLMs with human feedback [34]. In this framework, a policy π_θ with parameters θ is refined using a reward signal R_{ϕ^*}. PPO improves the policy by optimizing a surrogate objective that includes a clipping mechanism, ensuring a balance between effective learning and policy stability. Letting $r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{old}(a|s)}$ denote the ratio between the updated and previous policy probabilities for action a in state s, the PPO

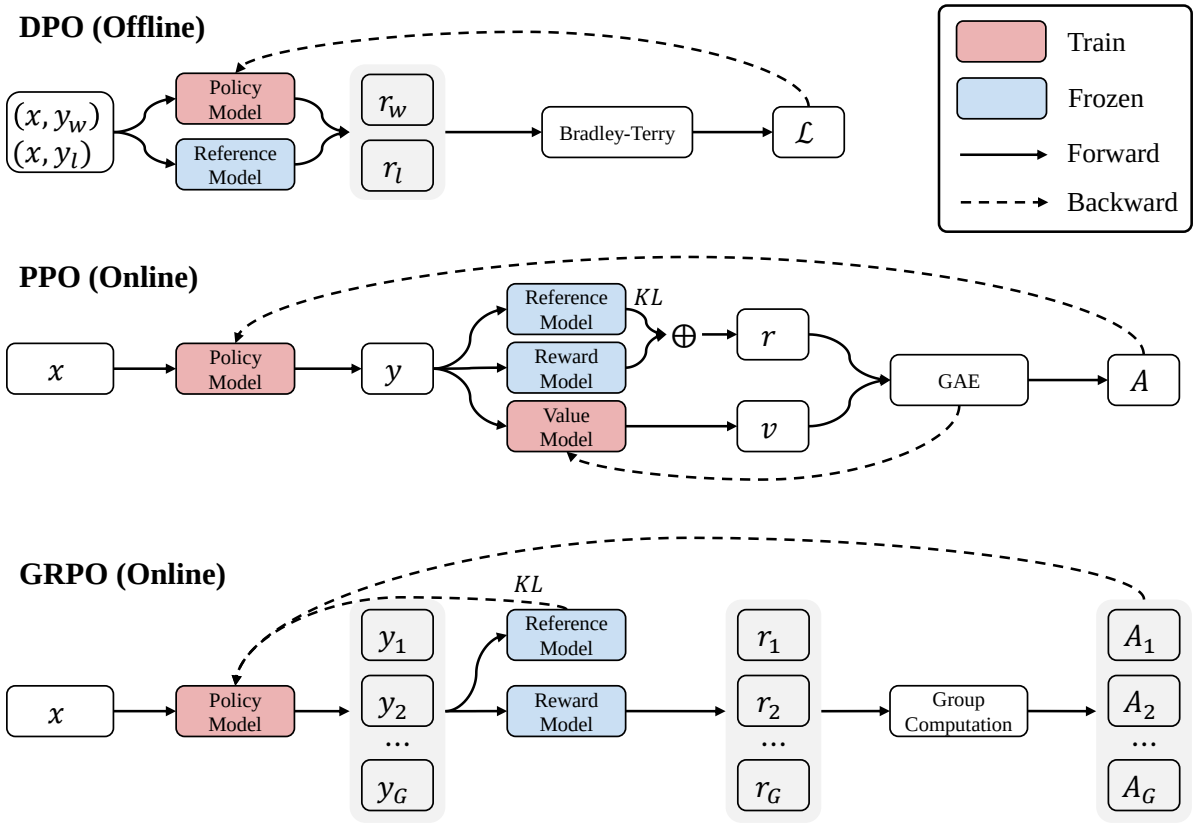


Figure 2.1: Demonstration of RL optimization algorithms: DPO, PPO, and GRPO.

objective becomes:

$$\mathcal{L}^{\text{PPO}}(\theta) = -\mathbb{E}_{s\sim P(S),a\sim\pi_{\text{old}}(A|s)}\Big[\min\Big(r(\theta)A(s,a),\ \text{clip}\left(r(\theta),1-\epsilon,1+\epsilon\right)A(s,a)\Big) - \beta\mathbb{D}_{\text{KL}}\left[\pi_\theta(y\mid x)\,\|\,\pi_{\text{ref}}(y\mid x)\right]\Big], \tag{2.10}$$

where $A(s,a)$ represents an estimate of the advantage function, while ϵ is a tunable parameter that limits how much the policy is permitted to shift. Advantage estimates are typically derived using Generalized Advantage Estimation (GAE) [153], which leverages both reward feedback and a learned value function. The clipping operation mitigates excessive changes in the policy, thereby preventing destabilizing updates in text generation tasks and supporting more robust training dynamics.

PPO is a foundational component in Reinforcement Learning from Human Feedback (RLHF) [137], where LLMs are guided by explicit human preference data to better match user expectations. More recent approaches, such as Reinforcement Learning from AI Feedback (RLAIF) [96], substitute human evaluations with model-generated signals. Em-

pirical findings [96, 146] suggest that RLAIF can offer a scalable and resource-efficient pathway for fine-tuning LLMs, making it a compelling alternative to traditional human-in-the-loop methods.

Direct Preference Optimization (DPO) [148] is a popular method for offline preference tuning, frequently employed in RLHF. It simplifies training and improves stability by reformulating the reward function. Based on the reinforcement learning objective given in Eq.2.9, DPO represents the optimal reward R_{ϕ^*} using the closed-form relationship:

$$R_{\phi^*}(x, y) = \beta \log \frac{\pi^*_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x), \tag{2.11}$$

where $Z(x)$ denotes the partition function. This formulation enables the use of the Bradley-Terry (BT) model [14], which defines the probability of a preferred output y_w over y_l as $p(y_w \succ y_l) = \sigma(R_{\phi^*}(x, y_w) - R_{\phi^*}(x, y_l))$. DPO further shifts from modeling rewards directly to modeling policy behavior, leading to the objective:

$$\mathcal{L}^{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} \right) \right]. \tag{2.12}$$

Here, the dataset $\mathcal{D}$ contains triples (x, y_w, y_l), with x representing the instruction input, and y_w, y_l denoting the preferred and non-preferred responses, respectively.

Since the inception of DPO, numerous studies have sought to advance this method by refining its training objective [183]. For instance, IPO [6] introduces an alternative pairwise preference loss to mitigate overfitting to the preference dataset, while R-DPO [142] incorporates a regularization term to prevent the exploitation of latent length bias in the training data.

Group Relative Policy Optimization (GRPO) [156] streamlines the conventional PPO approach by discarding the separate value (critic) network. Instead of relying on a value function, it calculates a baseline using the average reward from a set of responses generated for the same input. This strategy simplifies the training pipeline, lowers memory consumption, and enhances the stability of policy updates.

Given an instruction x, GRPO generates a batch of responses $\{y_1, y_2, \ldots, y_G\}$ using the prior policy π_θ^{old}. A reward model assigns a score to each sample, producing corresponding rewards $\{r_1, r_2, \ldots, r_G\}$. These reward values are standardized within the group by subtracting the mean and dividing by the standard deviation:

$$\hat{A}_i = \tilde{r}_i = \frac{r_i - \text{mean}(r)}{\text{std}(r)}, \tag{2.13}$$

where $\hat{A}_i$ represents the advantage of the i-th response. If we define the importance sampling ratio as $r_i(\theta) = \frac{\pi\theta(a_i|s)}{\pi\text{old}(a_i|s)}$, the GRPO objective is formulated as follows:

$$\mathcal{L}^{\text{GRPO}}(\theta) = -\mathbb{E}_{s\sim P(S),\, a_i\sim\pi_{\text{old}}(A|s)} \frac{1}{G} \sum_{i=1}^{G} \Big[\min\Big(r_i(\theta)\hat{A}_i,\ \text{clip}(r_i(\theta), 1-\epsilon, 1+\epsilon)\, \hat{A}_i \Big) - \beta\, \mathbb{D}_{\text{KL}}\big[\pi_\theta(y_i \mid x) \,\|\, \pi_{\text{ref}}(y_i \mid x)\big] \Big]. \tag{2.14}$$

By exploiting the statistical properties of grouped outputs, GRPO eliminates the need for learning a separate critic model, as done in traditional actor-critic algorithms. This makes it a resource-efficient and scalable solution, particularly as adopted in the DeepSeek R1 model [67], while still preserving sensitivity to subtle quality differences between generated responses.

2.4 Alignment Evaluation

Following the collection of alignment datasets and the subsequent training of LLMs, the next critical step is to assess how well these models align with the intended objectives. This section introduces the evaluation benchmarks in §2.4.1 and outlines different evaluation paradigms in §2.4.2.

2.4.1 Evaluation Benchmarks

A wide range of benchmark suites has been developed to measure the effectiveness of alignment in LLMs. Broadly, these can be grouped into two categories: closed-ended and open-ended benchmarks. Closed-ended benchmarks primarily test a model's capabilities in predefined tasks with known answers, while open-ended benchmarks assess

performance in more flexible, real-world scenarios where responses are subjective or unconstrained.

Closed-ended Benchmarks. Closed-ended evaluation benchmarks typically feature test cases where the set of possible answers is predetermined and finite—such as in multiple-choice formats. Below, we review several widely adopted benchmarks in this category:

- **General Knowledge:** The MMLU dataset [72] is a prominent English-language benchmark for evaluating the factual and academic knowledge of LLMs under zero-shot and few-shot conditions. It includes a wide-ranging set of questions across 57 disciplines, from elementary subjects to specialized professional fields in areas like science, humanities, and social sciences. Its subject diversity and granularity make it a powerful tool for identifying the limits of LLMs' knowledge. Analogous Chinese benchmarks include C-MMLU [100], C-Eval [80], M3KE [111], and AGIEval [212]. These benchmarks evaluate Chinese-language LLMs using varied subject matter and difficulty levels, drawing on questions from sources like national college entrance exams, advanced mathematics contests, and legal assessments.

- **Reasoning:** Reasoning represents a core aspect of intelligence, essential for handling complex problems. Remarkably, large-scale LLMs often exhibit emergent reasoning capabilities as model size increases. To test these abilities, several benchmarks have been developed. For numerical reasoning, GSM8K [38] and MATH [73] serve as standard benchmarks. Commonsense reasoning tasks are evaluated using datasets like CSQA [162] and StrategyQA [60], which require inference based on everyday knowledge. The BBH benchmark [161] evaluates a wide range of logical tasks including temporal understanding, categorization, and causality.

- **Programming:** Several benchmarks focus on testing LLMs' capabilities in code generation. HumanEval [25], HumanEval+ [112], and MBPP [5] consist of Python programming challenges paired with test cases that assess the functional correctness of generated solutions. The DS1000 dataset [95] includes 1,000 tasks across seven popular data science libraries, offering two evaluation modes—code completion and code insertion—based on automated test case validation.

Open-ended Benchmarks. Unlike closed-ended benchmarks, open-ended evaluations are characterized by their flexible, unconstrained response formats. These typically involve conversational or instruction-following tasks without predefined correct answers. Early open-ended datasets like Vicuna-80 [31], Open-Assistant-953 [91], and User-Instructions-252 [180] use relatively small collections of synthetic prompts to gauge LLM performance. However, these benchmarks tend to be limited in scope, often allowing comparison across only a few models at a time. To address this, newer benchmarks such as AlpacaEval [107] and Arena-Hard [104] introduce competitive evaluation strategies where model outputs are directly compared to a reference model. A higher Win Rate indicates superior performance, enabling a more scalable and interpretable comparison across numerous LLM candidates.

2.4.2 Evaluation Paradigms

In scenarios where open-ended tasks lack definitive reference answers, external evaluators—either human or language models—are often necessary for assessment. This section outlines the main paradigms used for evaluation, encompassing both human annotators and LLMs.

Human-based Evaluation. Traditional automatic metrics such as BLEU [141] and ROUGE [109] depend on the existence of reference outputs and tend to show weak alignment with human judgments, making them unsuitable for open-ended response evaluation. To overcome this limitation, human raters are frequently employed to judge the quality of model-generated outputs in such settings. For instance, [180, 190] adopt an ordinal annotation approach, instructing annotators to classify responses into one of four quality categories: acceptable, minor issues, major issues, or unacceptable. However, this method is susceptible to subjective biases, often leading to low agreement among annotators [85]. As an alternative, [164] suggests a comparative judgment method, where annotators are shown outputs from two different models alongside the same prompt and asked to identify the superior response. Building on this idea, [211] introduces the Elo rating system—commonly used in competitive games like chess—to compute relative rankings among multiple LLMs. In this framework, the scores of each model are dynamically

adjusted after every comparison, based on prior ratings and evaluation outcomes.

LLMs-based Evaluation. Although human evaluation delivers high-quality insights, it is often resource-intensive and slow. Moreover, as LLM-generated content increasingly mirrors human writing, it becomes harder for annotators to reliably differentiate between the two [35]. To reduce reliance on manual labor and references, recent work has turned to using LLMs themselves as evaluators across various natural language generation (NLG) tasks. These approaches often bypass the need for gold-standard references and leverage LLMs' own reasoning abilities. Some studies mimic human pairwise comparison setups by prompting LLMs—such as GPT-4—to choose the better output given two candidate responses for a single input [107, 211], thereby automating evaluation in a cost-effective and scalable manner.

CHAPTER 3

ALIGNMENT DATA SYNTHESIS BY ADVERSARIAL DISTILLATION

3.1 Introduction

LLMs capable of following natural language instructions have exhibited tremendous success in generalizing zero-shot to new tasks and aligning with human values [125, 184]. Due to various concerns, the most advanced LLMs, such as ChatGPT [136] and GPT-4 [135] that boasting billions of parameters, are typically proprietary, comprising both the model parameter and the training data. To foster increased transparency regarding their intricate operational mechanics, a surge in research efforts focusing on knowledge distillation from a proprietary "teacher" LLM to an open-source "student" LLM. This is typically accomplished by aligning the responses of the student model with those of the teacher model to a set of instructions, which can be manually or automatically generated [180, 164, 211, 194].

However, previous works employ a unidirectional approach to knowledge transfer (solid arrow in Figure 3.1), where the teacher imparts knowledge to the student without considering any "feedback". To better illustrate this using a tangible classroom scenario, the "feedback" refers to identifying the "hard" examples or problems where the student's performance falls short. This feedback guarantees that the teacher can provide bespoke training that centers on "hard" examples, thereby paving the way for more effective and tailored learning experiences for the student.

Inspired by adversarial knowledge distillation (AKD), which aims to iteratively improve the student model's performance by learning from generated hard samples [55, 122, 74], we propose an adversarial framework for distilling a proprietary LLM into a compact student model. Nevertheless, these AKD methodologies necessitate accessibility to the weights or gradients of the teacher model, which cannot be directly adapted to our setting. To circumvent this problem, we leverage the unparalleled role adaptability of

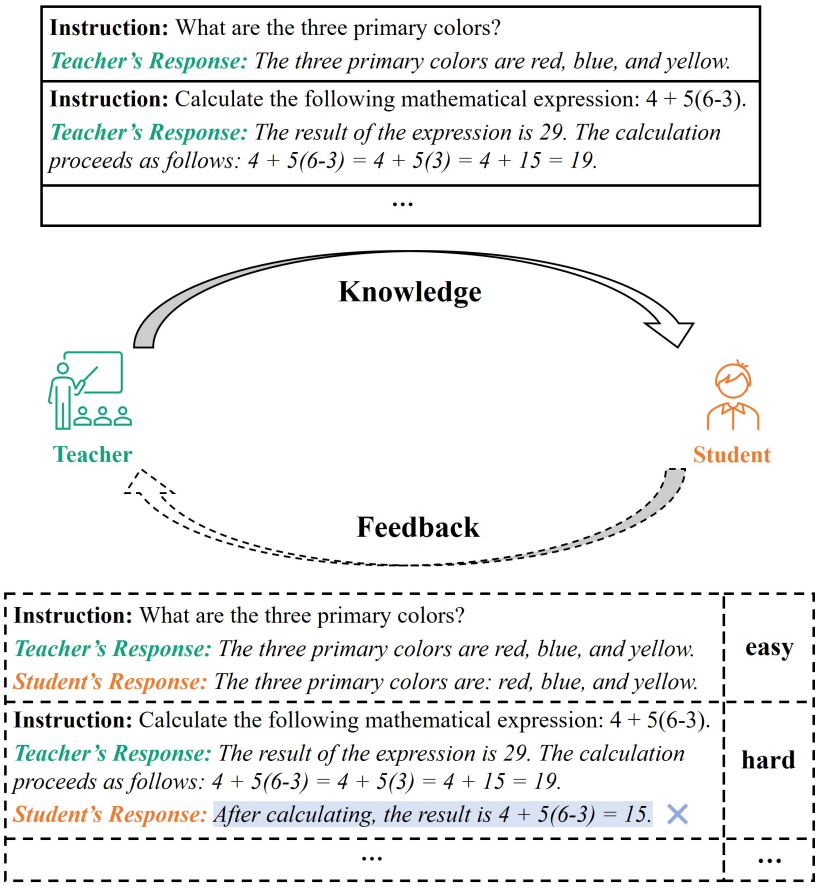


Figure 3.1: An illustration of the distinction between our approach and earlier ones. Previous methods facilitate a one-way knowledge transfer from the teacher to the student (*solid arrow*). Our approach, however, incorporates an innovative step (*dashed arrow*) that completes a loop: it enables the feedback"—identifying the student model's weaknesses—to be relayed back to the teacher, in order to foster tailored learning.

LLMs, which can be effectively employed through a diverse range of prompts [152]. In particular, we prompt the proprietary teacher LLM to serve as a "referee" to discriminate hard instructions where there exists a significant performance discrepancy between the teacher's and student's responses, and serve as a "generator" to produce new instructions that emulate the data distributions corresponding to the discriminated hard instructions. Our framework, as depicted in Figure 3.2, consists of three stages in an iteration: (1) an imitation stage to align the student's response with the teacher's response; (2) a discrimination stage to identify hard instructions; (3) A generation stage to produce new hard instructions for escalating the challenges presented to the student model. In essence, our adversarial framework forms a *positive feedback loop* that efficiently bootstraps the student model's proficiency.

To verify the efficiency and efficacy of our method, we apply our AKD framework to transfer the knowledge of ChatGPT[1] onto an open-source foundation LLM, known as LLaMA [171]. We select Alpaca's training data (generated from only 175 manually selected seed instructions) as the initial training instructions and execute three iterations of AKD, resulting in a total of 70K data that our model is trained on. We've christened our model as **Lion**, drawing inspiration from the art of "distillation". By conducting extensive experiments on open-ended generation and reasoning datasets, which include a total of 40 sub-tasks, our Lion-13B showcases superior performance surpassing instruction-tuned baseline models such as Vicuna [211].

Our main contributions are as follows:

- Our work is the first attempt to adopt the idea of adversarial knowledge distillation to large language models.
- Our proposed framework demonstrates impressive efficiency and efficacy. With instruction tuning performed on 70k data without any human annotation, our Lion-13B approximates ChatGPT's capabilities on open-ended generation dataset and largely outperforms the current SOTA model Vicuna-13B on reasoning tasks.
- The versatility of our framework allows for broad application: it is not exclusive to ChatGPT but can be conveniently adapted to suit a variety of other proprietary LLMs.

3.2 Methodology

Harnessing the learned knowledge of a sophisticated teacher model $\mathcal{T}(x;\theta^{\mathcal{T}})$ where the parameter $\theta^{\mathcal{T}}$ is inaccessible, our goal is to craft a more lightweight student model $\mathcal{S}(x;\theta^{\mathcal{S}})$. Ideally, a student model is optimal if the expectation of model discrepancy (which indicates the prediction differences between teacher $\mathcal{T}$ and student $\mathcal{S}$) on the uniform data distribution is minimized. Inspired by the success of adversarial knowledge distillation (AKD) [55, 122, 74], we turn to optimize an upper bound of the expectation —the expectation of the model discrepancy on "hard samples", where the teacher $\mathcal{T}$ and the student

[1]We access ChatGPT using the OpenAI API (*gpt-3.5-turbo model*).

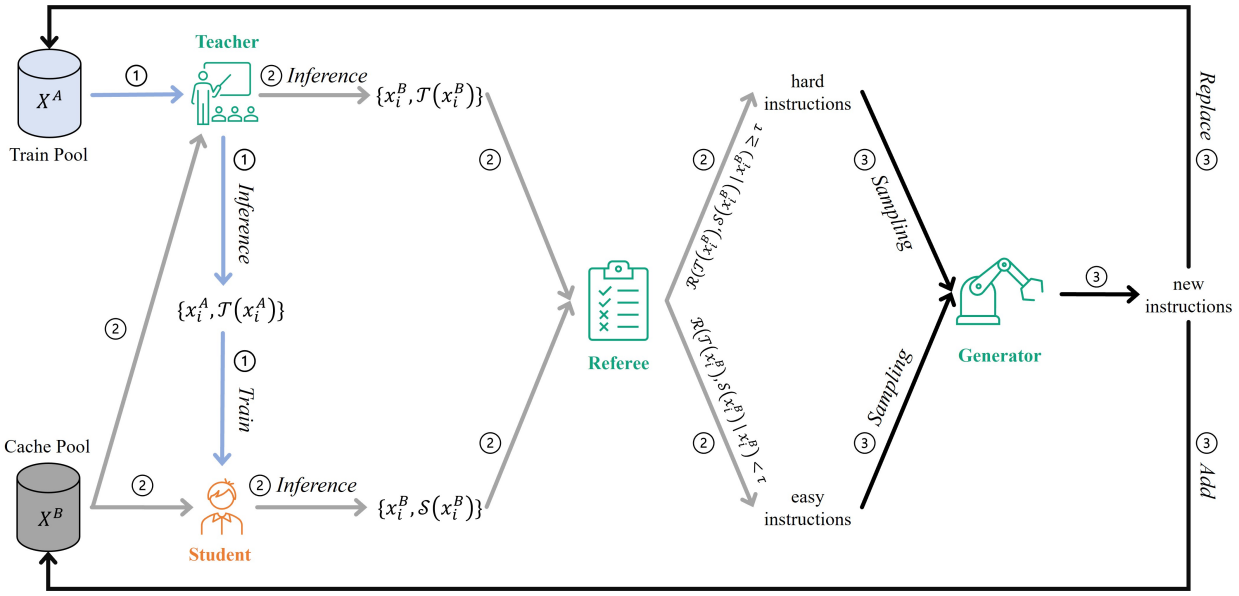


Figure 3.2: The overview of our adversarial distillation framework, where we craft a compact Student LLM $\mathcal{S}$ based on a superior proprietary LLM that serves three roles: the **Teacher** $\mathcal{T}$, the **Referee** $\mathcal{R}$, and the **Generator** $\mathcal{G}$. From left to right, there are three stages in an iteration: (1) Imitation; (2) Discrimination; (3) Generation.

$\mathcal{S}$ have a relatively large performance gap. These "hard samples" are inclined to dominate the expectation of the model discrepancy. Thus, the overall expected model discrepancy can be effectively and efficiently reduced by optimizing the student model $\mathcal{S}$ on these "hard samples". The underlying rationale is rather straightforward and can be analogized to a real-world educational scenario: continuously concentrating on the "hard" knowledge that the student finds challenging to grasp is the most effective manner of enhancing a student's proficiency.

However, in the process of training the student model $\mathcal{S}$, hard samples will be mastered by the student and converted into easy samples. Hence we need a mechanism to continuously generate hard samples, which can be achieved by an adversarial framework.

The whole framework of our *Adversarial Knowledge Distillation* is depicted in Figure 3.2, which contains three stages in an iteration: (1) an imitation stage to align the student's response with the teacher's response; (2) a discrimination stage to identify hard samples; (3) A generation stage to produce new hard samples for escalating the challenges presented to the student model.

3.2.1 Initilization

As shown in Figure 3.2, four roles and two data pools are established in our framework, and we will comprehensively illustrate their functions later. We initialize our student model $\mathcal{S}$ using a foundation LLM such as LLaMA [171]. We initialize our teacher model $\mathcal{T}$, referee $\mathcal{R}$, and generator $\mathcal{G}$ by using the same proprietary LLM such as ChatGPT [136]. The multiple roles that this proprietary LLM serves are accomplished through the use of varied prompt templates. We start the iteration from a given initial Train Pool $X^A = \{x_i^A\}_{i\in[1,N^A]}$, where x_i^A is the i-th instruction in X^A, and N^A is the number of samples in X^A. The Cache Pool X^B is initialized as identical to X^A, consisting of instructions to evaluate the performance of $\mathcal{S}$ and $\mathcal{T}$.

3.2.2 Imitation Stage

To impart the knowledge of the teacher to the student, we construct the instruction-response data $\{x_i^A, \mathcal{T}(x_i^A)\}_{i\in[1,N^A]}$ by forward propagating instructions in the Train Pool X^A through the teacher $\mathcal{T}$. The prompt template used for model inference is shown in Table A.5. Like the imitation training of previous work [164, 211], we fine-tune our student model $\mathcal{S}$ to align the response of the teacher model, by optimizing the autoregressive language modeling objective.

3.2.3 Discrimination Stage

Figure 3.2 demonstrates that the discrimination stage starts from the Cache Pool, denoted as X^B. Even though this pool begins with the same initialization as the Train Pool, their uses diverge. The Train Pool is rejuvenated by replacing its existing instructions with freshly generated instructions, whereas the Cache Pool is enriched by incorporating these generated instructions. As a result, the growing storage capacity of the Cache Pool provides a more extensive space for evaluating the performance gap between teacher $\mathcal{T}$ and student $\mathcal{S}$. This allows for more thorough detection of hard instructions.

In the discrimination stage, we ask the proprietary LLM to serve as a "referee", which quantifies the performance gap between $\mathcal{T}$ and $\mathcal{S}$. Specifically, we feed each instruction x_i^B

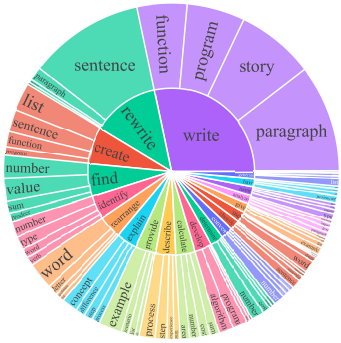
in the Cache Pool X^B through both the teacher $\mathcal{T}$ and student $\mathcal{S}$ to generate the outputs $\mathcal{T}(x_i^B)$ and $\mathcal{S}(x_i^B)$, respectively. Then we ask the referee $\mathcal{R}$ to quantitatively measure the quality difference between teacher's response $\mathcal{T}(x_i^B)$ and student's response $\mathcal{S}(x_i^B)$, conditioned on x_i^B:

$$d_i = \mathcal{R}(\mathcal{T}(x_i^B), \mathcal{S}(x_i^B) \mid x_i^B) \tag{3.1}$$

The above process is conducted by using the prompt template (as shown in Table A.6) inspired by [211], which requires the LLM to consider the helpfulness, relevance, accuracy, and level of detail of two responses and output two scores. To mitigate the positional bias [177] of the LLM referee, we conduct two runs by exchanging the positions of the teacher's response and the student's response and compute the final score as the average of the two runs. Then d_i is calculated as the difference between the teacher's score and the student's score. By setting a threshold τ (1.0 used in our experiments), we discriminate hard instructions as those instructions with $d_i \geqslant \tau$, and the others are identified as easy ones. Figure 3.3b provides a clear and intuitive demonstration of which kinds of instructions are discriminated as hard in the first iteration. Compared with the instructions in the Cache Pool (Figure 3.3a), the distribution of the identified hard instructions is quite different, focusing more on complex tasks such as math, coding, etc.

(a) Instructions of the Cache Pool in the first iteration.

(b) Identified hard instructions in the first iteration.



(c) Generated hard instructions in the first iteration.

Figure 3.3: The top 20 most common root verbs (inner circle) and their top 4 direct noun objects (outer circle) in the instructions.

3.2.4 Generation Stage

After carefully discerning the hard instructions, the generation stage aims to produce samples that mirror the data distributions corresponding to these challenging directives. This process is achieved by employing the proprietary LLM as a generator, denoted as $\mathcal{G}$, leveraging its exceptional prowess in content creation. Inspired by [194], we randomly sample an instruction from the hard instructions and prompt the generator $\mathcal{G}$ to generate a new instruction. The newly generated instruction is required to pertain to the same domain and match the task type of the sampled instruction. The template utilized for this prompt is exhibited in Table A.7. As shown in Figure 3.3c, the distribution of the newly generated hard instructions appears to be comparable to that of the previously identified hard instructions. To mitigate the issue of catastrophic forgetting and to augment the diversity of the generated instructions, we also randomly sample an instruction from the easy instructions and prompt the generator $\mathcal{G}$ to generate a new instruction that belongs to the same domain as the sampled one, but exhibit a more long-tailed distribution. The template we use to prompt this process is displayed in Table A.8.

In each iteration, we define N as the total count of newly generated instructions and maintain a 1:1 ratio r between the generated hard instructions and the generated easy instructions. To promote diversity, a new instruction will be deemed valid only if its ROUGE-L overlap with any existing instructions in the Cache Pool is below 0.7. Finally, as aforementioned in §3.2.3, we proceed to rejuvenate the Train Pool, replacing its existing instructions with freshly generated ones. Concurrently, we enrich the Cache Pool by incorporating these newly generated instructions.

3.2.5 Min-Max Game Interpretation

Our adversarial knowledge distillation framework can be interpreted as a dynamic min-max game: in the imitation stage, we fine-tune our student to *minimize* the model discrepancy between itself and the teacher on hard samples; in the discrimination and generation stage, we craft new hard samples to *maximize* the model discrepancy, based on the learning progress of the student model. This dialectic framework propels the student model towards uncovering otherwise hidden knowledge, paving the way to complete under-

standing. As the training progresses through several iterations, the system should ideally achieve equilibrium. This is the point where the student model has mastered all the hard samples and the referee $\mathcal{R}$ can no longer distinguish between the student $\mathcal{S}$ and teacher $\mathcal{T}$ models. At this juncture, $\mathcal{S}$ becomes functionally indistinguishable from $\mathcal{T}$.

3.3 Experiments

3.3.1 Experimental Setup

Datasets

In our experiments, we implemented a comprehensive LLM evaluation protocol that considers a diverse range of abilities, such as writing, coding, commonsense, math, and logical reasoning. The datasets we utilized can be classified into two main categories: open-ended generation and reasoning.

Open-ended Generation Datasets. Vicuna-Instructions [211] is a set of 80 questions spanning 9 distinct task categories. This dataset has gained extensive usage in evaluating the capabilities of LLMs. Within our work, we examine LLMs' performance on this dataset in two different settings:

- **Setting1:** Following Vicuna [211], we leverage GPT-4 to automatically assess the quality of responses (rated on a scale of 1 to 10) between a reference model (ChatGPT) and a candidate model. Subsequently, we calculate the candidate model's performance as the percentage of the total score it achieves compared to the reference model.
- **Setting2:** A recent work [177] pointed out that a systematic bias may exist in the above-mentioned GPT-4 automatic evaluation. To mitigate this, they propose two strategies, namely Multiple Evidence Calibration and Balanced Position Calibration, to obtain closer alignment with human judgments.

Reasoning Datasets. AGIEval [212] is a well-known benchmark that quantifies the reasoning capability of foundation models in the context of human-centric standardized exams, including college entrance exams, math competitions, lawyer qualification tests, etc. We choose all English multiple-choice questions (8 tasks, 2,546 samples) among AGIEval for our experiments. The data statistics are shown in Table A.1. BIG-Bench Hard (BBH) [161] consists of a suite of challenging tasks from BIG-Bench [160], designed to assess the capabilities and limitations of large language models. These are the tasks on which prior language models underperform the average human rater. We choose all tasks that can be formatted into multiple-choice questions (23 tasks, 5,511 samples) among BBH for our experiments. The data statistics are shown in Table A.2.

- **Setting:** We evaluate reasoning capabilities under a zero-shot setting without any exemplars and without Chain-of-Thought (CoT). For both AGIEval and BBH, we use the prompt format and parsing following [212, 129]. Given the free-form response from the generative models, only the first capital character in the response is considered to compare with the gold answer (exact match). The result we report is accuracy (%).

Baselines

We select five superior LLMs as baselines, including LLaMA [171], Alpaca [164], WizardLM [194], Vicuna [211], and ChatGPT [136]. It is worth noting that Vicuna has consistently ranked as the top open-source language model on multiple leaderboards, such as Chatbot Arena[2]. Therefore, we will conduct a comprehensive comparison with Vicuna. See detailed descriptions of these baselines in Appendix A.2.

Implementation Details

Training Details. Our student model is initialized using the pre-trained LLaMA. The Train Pool and Cache Pool are initialized with the 52K automatically generated instructions from Alpaca [164]. The total number of iterations is set to 3, with 6K newly generated instructions added at each iteration. This results in a total of 70K data that our model is

[2] `https://chat.lmsys.org/?arena`

trained on in order to make a fair comparison with current SOTA baselines, including WizardLM and Vicuna. The training hyperparameters are listed in Appendix A.3.

Inference Details. To draw inferences from Lion and ChatGPT, we calibrated the temperature to 0.7 and set the maximum generation length at 1024. All other parameters adhere to their default settings. For LLaMA, Alpaca, WizardLM, and Vicuna, we configured their inference parameters in line with the specifications given in their respective original papers. When engaging with the gpt-3.5-turbo API for various roles, we employ an array of hyper-parameters, the specifics of which can be located in Appendix A.3.

3.3.2 Experimental Results

Results for Open-ended Generation

Table 3.1 shows the performance comparison of various models against ChatGPT as the reference model, where GPT-4 is used as a referee/rater. Our Lion-7B and Lion-13B remarkably outperform their counterparts under two evaluation settings. Noticeably, Lion-13B shows an 8-point improvement over Vicuna-13B on aggregate, achieving 98.38% capabilities of ChatGPT.

Model	Setting1	Setting2	Avg.
LLaMA-7B	58.46	59.12	58.79
Alpaca-7B	69.29	67.20	68.25
WizardLM-7B	89.29	86.67	87.98
Vicuna-7B	87.79	89.96	88.88
Lion-7B	**94.74**	**92.88**	**93.81**
LLaMA-13B	69.23	68.21	68.72
Alpaca-13B	76.87	74.69	75.78
Vicuna-13B	92.25	92.97	92.61
Lion-13B	**96.57**	**100.18**	**98.38**

Table 3.1: Relative response quality (%) against ChatGPT (assessed by GPT-4) on Vicuna-Instructions.

To comprehensively compare with other baseline models on the capability to generate high-quality responses on various types of instruction, the relative response quality (Setting2) among different task categories is depicted in Figure 3.4. Our model impressively

and slightly surpasses ChatGPT in the generic, knowledge, common-sense, and counterfactual task categories. Furthermore, for the two difficulty task categories described in the previous study [211, 194], our model significantly outperforms other baseline models with at least 32.32% relative score in the math task category while exceeding most of the baseline in the coding generation task category.

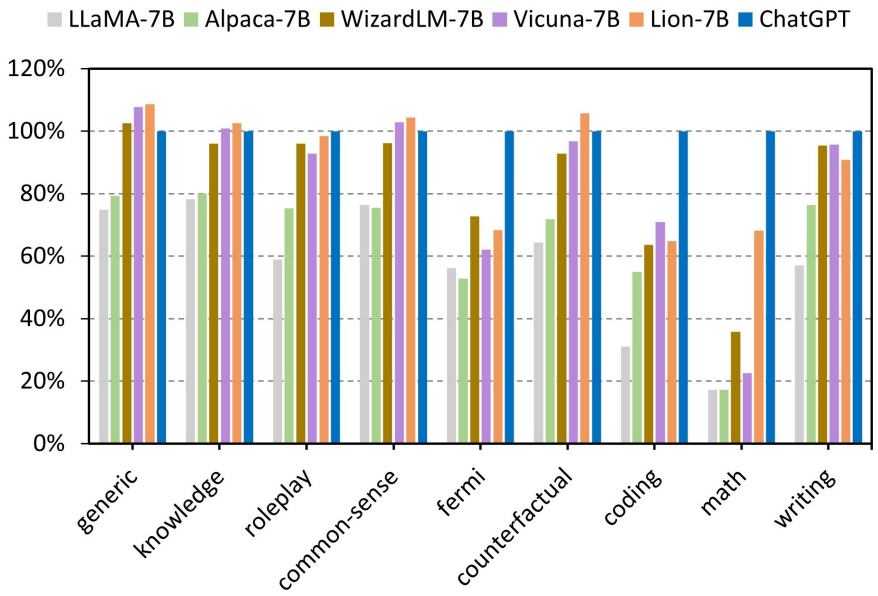


Figure 3.4: Relative response quality against ChatGPT on diverse task categories of Vicuna-Instructions.

Results for Reasoning

AGIEval Results. Table 3.2 presents the standard zero-shot performance comparison between Lion and baseline models on the AGIEval benchmark for multiple-choice English questions. Lion demonstrates significantly stronger performance compared to Vicuna, surpassing it in most task categories and achieving an average relative improvement of over 16%. However, Lion-13B still significantly lags behind ChatGPT, only retaining 72.5% of its reasoning capability.

BIG-Bench Hard Results. Table 3.3 displays the zero-shot performance comparison between Lion and baseline models on BIG-Bench Hard with standard zero-shot prompting. Similar to AGIEval, Vicuna exhibits poor performance on sophisticated reasoning

Task	Human Avg	Human Top	ChatGPT	Vicuna-7B	Lion-7B	Vicuna-13B	Lion-13B
AQuA-RAT	85.0	100.0	31.9	**23.2**	18.5 (-20.3%)	20.1	**26.0** (29.4%)
LogiQA	86.0	95.0	35.0	21.4	**31.8** (48.6%)	29.8	**31.3** (5.0%)
LSAT-AR	56.0	91.0	24.4	**22.2**	17.4 (-21.6%)	20.4	**23.0** (12.7%)
LSAT-LR	56.0	91.0	52.6	18.6	**28.2** (51.6%)	**32.6**	**32.6** (0.0%)
LSAT-RC	56.0	91.0	65.4	21.9	**29.4** (34.2%)	32.7	**40.9** (25.1%)
SAT-Math	66.0	94.0	42.7	**21.4**	20.9 (-2.3%)	28.6	**29.4** (2.8%)
SAT-English	66.0	94.0	81.1	25.7	**36.4** (41.6%)	44.2	**53.9** (21.9%)
SAT-English (w/o Psg.)	66.0	94.0	44.2	26.2	**27.7** (5.7%)	26.2	**36.2** (38.2%)
Average	67.1	93.8	47.2	22.6	**26.3** (16.4%)	29.3	**34.2** (16.7%)

Table 3.2: Zero-shot performance comparison of ChatGPT, Vicuna, and Lion on AGIEval (multiple-choice English questions). We report the performance of Human, ChatGPT, and Vicuna from [129]. Performance improvements obtained by Lion over Vicuna are shown in parenthesis.

Task	ChatGPT	Vicuna-7B	Lion-7B	Vicuna-13B	Lion-13B
Boolean Expressions	82.8	39.2	**55.2** (40.8%)	40.8	**65.6** (60.8%)
Causal Judgement	57.2	39.7	**50.3** (26.7%)	42.2	**43.9** (4.0%)
Date Understanding	42.8	8.6	**34.0** (295.3%)	10.0	**40.4** (304.0%)
Disambiguation QA	57.2	15.2	**35.6** (134.2%)	18.4	**44.8** (143.5%)
Formal Fallacies	53.6	40.0	**46.0** (15.0%)	47.2	**52.4** (11.0%)
Geometric Shapes	25.6	3.6	**8.8** (144.4%)	3.6	**8.8** (144.4%)
Hyperbaton	69.2	42.8	**51.6** (20.6%)	44.0	**56.8** (29.1%)
Logical Deduction (5 objects)	38.8	4.8	**19.6** (308.3%)	4.8	**20.8** (333.3%)
Logical Deduction (7 objects)	39.6	1.2	**14.4** (1100.0%)	1.2	**21.2** (1666.7%)
Logical Deduction (3 objects)	60.4	19.6	**40.4** (106.1%)	16.8	**38.0** (126.2%)
Movie Recommendation	55.4	24.4	**26.8** (9.8%)	43.4	**57.6** (32.7%)
Navigate	55.6	43.6	**49.2** (12.8%)	**46.4**	45.2 (-2.6%)
Penguins in a Table	45.9	17.5	**24.7** (41.1%)	15.1	**26.7** (76.8%)
Reasoning about Colored Objects	47.6	14.0	**15.2** (8.6%)	12.0	**17.6** (46.7%)
Ruin Names	56.0	12.2	**14.4** (18.0%)	15.7	**29.2** (86.0%)
Salient Translation Error Detection	40.8	2.0	**12.0** (500.0%)	2.0	**12.4** (520.0%)
Snarks	59.0	28.0	**56.2** (100.7%)	28.1	**61.2** (117.8%)
Sports Understanding	79.6	40.4	**48.4** (19.8%)	48.4	**51.6** (6.6%)
Temporal Sequences	35.6	21.2	**24.4** (15.1%)	**16.0**	10.4 (-35.0%)
Tracking Shuffled Objects (5 objects)	18.4	6.4	**14.4** (125.0%)	9.2	**24.8** (169.6%)
Tracking Shuffled Objects (7 objects)	15.2	4.0	**13.6** (240.0%)	5.6	**13.2** (135.7%)
Tracking Shuffled Objects (3 objects)	31.6	26.8	**34.0** (26.9%)	23.2	**34.4** (48.3%)
Web of Lies	56.0	**49.4**	47.2 (-4.5%)	41.2	**54.8** (33.0%)
Average	48.9	21.9	**32.0** (45.9%)	23.3	**36.2** (55.4%)

Table 3.3: Zero-shot performance comparison of ChatGPT, Vicuna, and Lion on BIGBench Hard (multiple-choice questions) without CoT. We report the performance of ChatGPT and Vicuna from [129]. Performance improvements obtained by Lion over Vicuna are shown in parenthesis.

tasks within this benchmark, while Lion substantially surpasses Vicuna by around 50% on average. Particularly, Lion demonstrates significant performance enhancements of over 100% on tasks involving data understanding, semantic understanding (Disambiguation QA and Snarks), logical and geometric reasoning (Logical Deduction and Geometric Shapes), and position reasoning (Tracking Shuffled Objects). Despite achieving an aver-

age ability of nearly 74% compared to ChatGPT on BBH, Lion-13B surpasses ChatGPT in several tasks, including Movie Recommendation, Snarks (identifying sarcastic sentences from two nearly-identical ones), and Tracking Shuffled Objects. This demonstrates the effectiveness of our method.

3.4 Analysis

3.4.1 Ablation Study

The Threshold τ for Distinguishing between Hard and Easy Instructions. We systematically explored τ ranging from 0.0 to 2.0 and documented its influence on average performance across three datasets. Table 3.4 reveals an optimal range of τ between 1.0 and 1.5 for all datasets. Notably, elevating τ from 0.0 to 1.0 consistently enhances performance across all datasets, indicating effective differentiation between hard and easy instructions. However, a continuous increase from 1.0 to 2.0 gradually degrades performance due to decreased diversity in hard instructions. The ablation results demonstrate that our method is not quite sensitive to a large value of τ.

Threshold τ	Vicuna-Instructions (Avg.)	AGIEval (Avg.)	BBH (Avg.)
0.0	89.58	22.4	26.5
0.5	92.16	23.5	29.8
1.0	93.81	**26.3**	**32.0**
1.5	**94.09**	25.7	31.6
2.0	92.23	24.6	31.3

Table 3.4: Ablation study of the threshold τ for Lion-7B.

The Ratio r of Generated Hard and Easy Instructions. We change the ratio of generated hard instructions to generated easy instructions from 1:0 (all hard) to 0:1 (all easy) and investigate its impact on average performance across three datasets. It can be seen from Table 3.5 that higher ratios of hard to easy instructions generally lead to improved performance, with a balanced ratio of 1:1 yielding the highest average scores.

Ratio r	Vicuna-Instructions (Avg.)	AGIEval (Avg.)	BBH (Avg.)
1:0	89.60	24.3	30.8
2:1	92.95	25.7	**33.1**
1:1	**93.81**	**26.3**	32.0
1:2	91.77	23.9	29.6
0:1	90.02	22.1	24.3

Table 3.5: Ablation study of the ratio r for Lion-7B.

3.4.2 The Learning Dynamics of Lion

In Figure 3.5, we delve into the learning dynamics of Lion by visualizing its performance on AGIEval and BBH throughout the training iterations. The results clearly demonstrate that our adversarial knowledge distillation framework consistently enhances the performance of the student model as the iterations progress. Notably, the most significant improvement in capability occurs in the first iteration, suggesting the usefulness of the identification of challenging example patterns (refer Figure 3.3b).

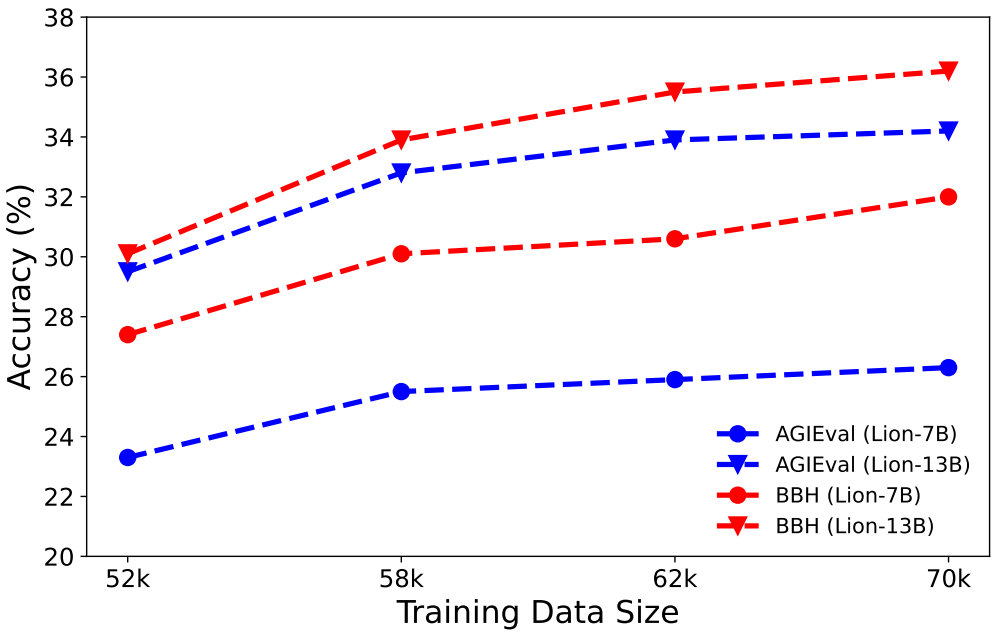


Figure 3.5: Performance of Lion-7B and Lion-13B on AGIEval and BBH through the training iterations.

3.5 Conclusion and Discussion

3.5.1 Conclusion

This chapter presents an innovative adversarial knowledge distillation framework for distilling a proprietary LLM into a compact, open-source student model. While previous methodologies have concentrated on unidirectional knowledge transfer, our approach seeks to integrate "feedback" into the learning process. Leveraging the versatile role adaptability of LLMs, we prompt the proprietary model to identify "hard" instructions and generate new "hard" instructions for the student model, creating a three-stage adversarial loop of imitation, discrimination, and generation. This approach allows us to refine the student model's performance iteratively, efficiently bootstrapping its proficiency. We aspire that our model, named Lion, may serve as a baseline to reflect the performance of ChatGPT, especially the open-source instruction-following language model baseline for our community.

3.5.2 Discussion

The Model Capability. We have identified that Lion is subject to certain constraints: (1) A recent study [66] asserts that "model imitation is a false promise" since imitation models are adept at mimicking ChatGPT's style but fall short in improving LMs across more challenging tasks. While Lion still lags behind its teacher model ChatGPT in handling intricate reasoning tasks (as shown in our experiments), it demonstrates promising improvements compared to previous imitation models. Therefore, our adversarial knowledge distillation framework may provide a more effective way for knowledge transfer. (2) Since our training data doesn't encompass dialogues, Lion struggles to manage multi-turn conversations. (3) Due to computational resource constraints, Lion's maximum sequence length is limited to 1024. Consequently, it faces challenges when dealing with long documents. Despite these limitations, we envision Lion serving as an accessible springboard for future research endeavors aimed at addressing these limitations.

The Training Process. To train a single student model, we request the gpt-3.5-turbo API around 450k times, a number that is roughly 70% of the WizardLM's usage of 624k [194].

Nonetheless, this utilization incurs a considerable expense, nearing $900. In contrast to methods like Alpaca [164] and WizardLM [194], which only fine-tune the student model once, our adversarial knowledge distillation method employs iterative parametric updates to the student model. While this iterative approach inevitably leads to slower iteration speed, it offers additional benefits. Finally, different from traditional adversarial knowledge distillation where the weights of the generator are iteratively updated, we use a black-box and parameter-frozen LLM (ChatGPT in our paper) to serve the role. Therefore, the quality of the LLM is quite essential in the generation of new instructions.

The Evaluation Metrics. Though automated evaluations leveraging GPT-4 have showcased promising prospects in appraising chatbot performance, the technique is yet to reach a level of maturity and accuracy, especially considering the propensity of large language models to generate non-existent or "hallucinated" information. Evaluating the efficacy of LLM across various tasks presents a considerable challenge since different tasks require quite different expertise [180]. Therefore, the creation of a comprehensive, standardized evaluation system for chatbots is a prevailing research challenge that demands additional exploration and study.

CHAPTER 4

ALIGNMENT DATA SYNTHESIS FROM SCRATCH VIA WEB RECONSTRUCTION

4.1 Introduction

LLMs [16, 135, 51] have become integral across a myriad of applications, demonstrating exceptional performance on diverse tasks by effectively following instructions and aligning with human values [136, 135]. Their remarkable performance largely stems from supervised fine-tuning (SFT) [184, 125] on instruction-response pairs. This process empowers LLMs to produce customized outputs when provided with specific instructions, facilitating their adaptation to novel tasks without prior exposure.

A fundamental challenge in advancing the instruction-following capabilities of LLMs lies in the collection of high-quality instruction-tuning (IT) data. Early approaches primarily rely on human experts to manually generate and curate IT data [181, 41], which is both time-intensive and resource-heavy. To mitigate these limitations, **Semi-Automated Synthetic Methods** [180, 164, 194] leverage LLMs to expand small, human-annotated seed datasets using few-shot prompting techniques. While effective, the performance of these methods is highly sensitive to prompt engineering and the careful selection of seed examples [196]. More recently, **Fully Automated Synthetic Methods**, such as WebInstruct [203] and instruction backtranslation [105], have emerged as scalable alternatives that eliminate human involvement by synthesizing IT data based on web-scraped documents. These methods, however, often operate under strong assumptions about the structure and content of raw web data, such as the availability of explicit question-answer pairs or minimal irrelevant content. Consequently, they can only handle a limited scope of web documents, restricting their diversity and leading to suboptimal performance across various tasks.

To overcome these limitations, we propose **Web Reconstruction (WebR)**—a novel framework that synthesizes high-quality IT data from raw web documents **with minimal**

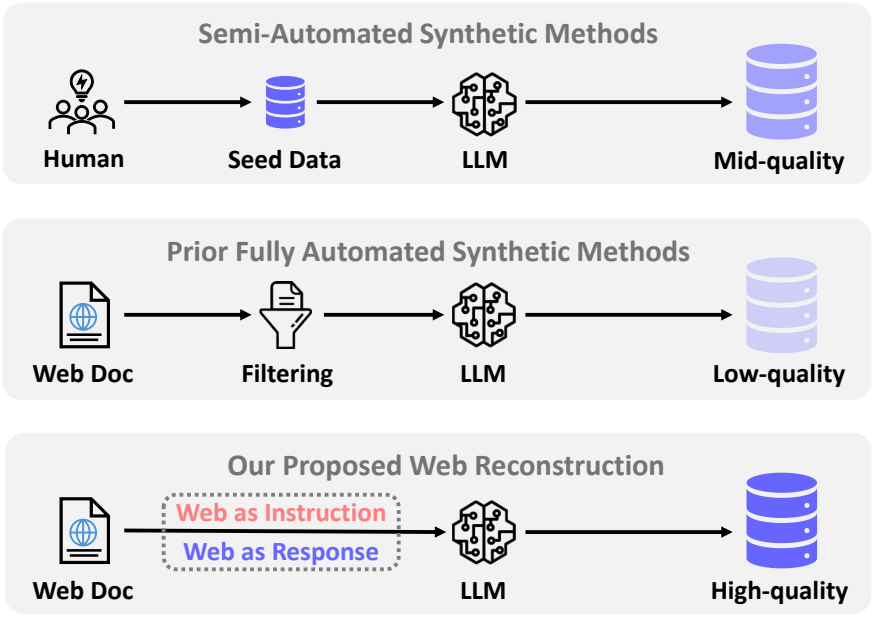


Figure 4.1: Our proposed Web Reconstruction method surpasses previous techniques by being (1) fully automated, eliminating the need for manual intervention or seed data; (2) minimally reliant on assumptions about the structure and content of web documents; and (3) capable of generating high-quality IT data.

assumptions on web and **no reliance on human annotations**, enabling broader adaptability and improved performance. Unlike backtranslation, which directly treats web content as a response, or WebInstruct, which extracts QA pairs, WebR introduces a novel paradigm by **conceptualizing web reconstruction as an instruction-tuning data synthesis task**. At its core, WebR aims to transform raw, noisy web documents into human-preferred, response-like outputs through a dual-perspective paradigm. Each web document is designated as either an instruction or a response, triggering the reconstruction process: (1) *Web as Instruction* introduces a first-of-its-kind **web rewriting** approach in IT data synthesis, where raw web document is concatenated with a synthesized rewrite request to serve as a complete instruction; (2) *Web as Response* enhances backtranslation [105] by introducing a novel rollout and refinement process, mitigating reliance on strong web content assumptions. Crucially, we show that these two perspectives are **complementary** (refer to Table 4.3): *Web as Instruction* enhances reasoning and understanding tasks, while *Web as Response* improves instruction-following and question-answering tasks.

We apply WebR to the Llama3-70B-Instruct and GPT-4o-mini models, creating two 100k-sample IT datasets: WebR-Basic and WebR-Pro. To validate their effectiveness, we

train various LLMs, including Llama3-8B-base and Qwen2.5-1.5/3/7/14B-base, and evaluate them on over ten widely used benchmarks. Our experiments provide key contributions and insights into IT data synthesis:

- **Efficacy:** WebR is the first web-based IT synthetic method to consistently surpass current IT datasets with human annotations;
- **Compatibility:** Merging WebR with existing IT datasets yields further performance gains;
- **Data Efficiency:** The performance of WebR improves linearly relative to the logarithmic growth of training data;
- **Scalability:** WebR scales with LLM size, consistently boosting larger models;
- **Domain Adaptability:** WebR achieves domain adaption by simply adjusting the proportion of source web documents.

4.2 Web Reconstruction

Prior fully automated synthetic methods often rely on strong assumptions about the structure and content of raw web documents—such as the presence of explicit question-answer pairs, minimal irrelevant content, or appropriate expressions—necessitating complex preprocessing steps like retrieval and filtering. In contrast, we introduce the **Web Reconstruction** (WebR) framework, which leverages a powerful, off-the-shelf LLM to overcome these limitations by directly reconstructing unstructured and noisy web content into high-quality, response-like outputs. As shown in Figure 4.2, WebR comprises two core strategies: (1) *Web as Instruction*, where raw web content is concatenated with a synthesized rewrite request to serve as a complete instruction, guiding the generation of a reorganized, coherent response; (2) *Web as Response*, where a latent instruction is inferred by treating raw web content as a response, enabling reconstruction through the LLM's initial rollout and subsequent refinement. By adopting this dual-branch approach, WebR efficiently generates high-quality instruction-response pairs, ensuring contextually appropriate outputs while eliminating the need for extensive preprocessing.

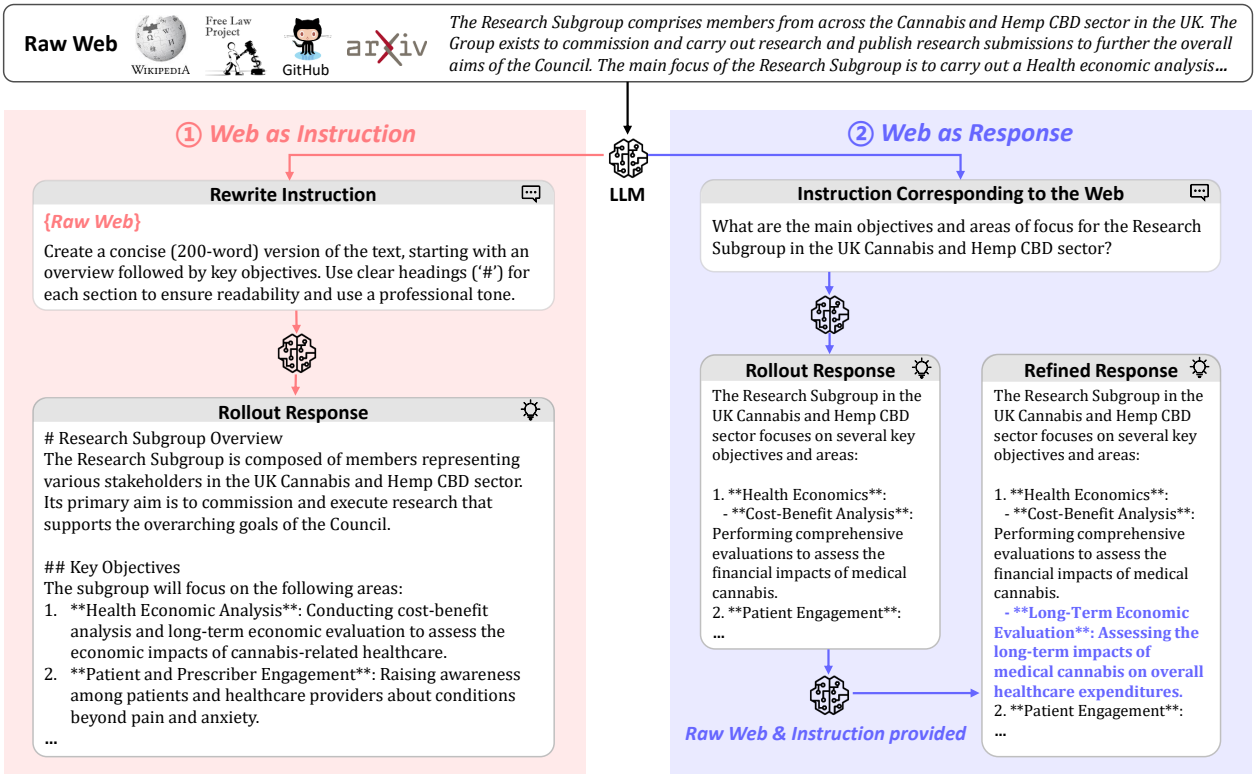


Figure 4.2: Overview of the proposed **Web Reconstruction** (WebR) framework. Leveraging an off-the-shelf LLM, WebR transforms raw web documents into high-quality instruction-response pairs. It strategically assigns each document as either an instruction or a response to trigger the process of web reconstruction.

4.2.1 Web as Instruction

Raw web documents often contain disorganized or irrelevant information that hinders direct usability. Even when dealing with well-structured content, further refinement is often required to meet human-preferred formats and stylistic conventions. A natural approach to reconstructing web content is to rewrite it according to specific requirements, such as style, format, structure, etc. To ensure diverse and realistic rewriting scenarios, we leverage a powerful LLM to generate a detailed rewrite request tailored to the original document's content (See prompt in Figure B.3). The request, along with the raw web content, are concatenated to form a comprehensive instruction. In addition to whole-document transformations, we further enhance task diversity by randomly (50% probability) generating rewrite requests that target *specific sections* of the web content rather than the entire document, as shown in Figure B.4. This simulates real-world text manipulation scenarios where users may need to extract and modify only certain portions of a text. The cu-

rated instructions are then processed by the LLM to produce reconstructed web content. Notably, the complexity of rewrite requests naturally encompasses various NLP tasks, such as summarization, information extraction, and semantic understanding. Addressing these tasks requires LLM to demonstrate advanced reasoning and comprehension abilities, thereby enhancing its proficiency in instruction-following, contextual understanding, and reasoning (as verified in Table 4.3).

4.2.2 Web as Response

Inspired by instruction backtranslation [105], we propose an alternative approach to reconstruct web content by treating the web as a response. Specifically, we utilize a LLM to predict a latent instruction for which the raw web content would serve as an ideal response, as illustrated in Figure B.5. To further enhance diversity, specific segments of web content are treated as responses (with a 50% probability), as depicted in Figure B.6. Unlike traditional back-translation methods, which directly treat latent instructions and raw web content as instruction-response pairs, **our approach introduces a two-stage refinement process**. First, we generate an initial response by rolling out an LLM prediction for the latent instruction. Next, we refine this response using both the raw web content and the latent instruction to produce a more accurate and comprehensive output, as shown in Figure B.7. The initial rollout ensures that the response exhibits human-like fluency and natural language style, while the subsequent refinement step integrates critical information from the raw web, ensuring that the final response is both precise and thorough. This dual-stage process significantly enhances the LLM's performance in knowledge acquisition and question-answering tasks, as demonstrated by the improvements reported in Table 4.3. The generated instruction as well as the refined response are finally paired as IT data.

4.2.3 Dataset Construction Details

Following prior work [105, 203], we construct our dataset by sampling raw web documents from three diverse and representative domains: 70% from the English subset of Common Crawl [40] (general domain), 15% from OpenWebMath [143] (math domain),

and 15% from GitHub [40] (code domain). To enable large-scale creation of diverse synthetic data for various scenarios, we adopt a persona-driven instruction synthesis strategy inspired by [58]. Initially, an LLM generates personas for the raw web documents (see template in Figure B.2), which guide the subsequent instruction synthesis for our proposed Web Reconstruction process. The ratio of *Web as Instruction* to *Web as Response* is set to 2:1, following insights from the ablation study presented in Table 4.3. To enhance diversity and eliminate redundancy, we apply MinHash [15] deduplication based on n-gram features of instructions. We configure the signature size to 128 and the similarity threshold to 0.7. The final synthesized dataset comprises 100,000 instruction-response pairs.

To evaluate the effectiveness of WebR in generating high-quality IT datasets, we use WebR to construct datasets with two LLMs: the open-source `Llama3-70B-Instruct` [51] (temperature=0.6, top-p=0.9) and the proprietary `GPT-4o-mini` [135] (temperature=0.7, top-p=1.0). The resulting datasets, **WebR-Basic** (from Llama3) and **WebR-Pro** (from GPT-4o-mini), differ in their generative capabilities and quality. A comparative analysis of the average token lengths is presented in Appendix B.3, while a detailed cost analysis of WebR is provided in §4.4.2. Notably, the overall expenditure for calling GPT-4o-mini API is **$38.57**.

4.3 Experiments

4.3.1 Experimental Setup

Baselines. We compare the family of IT datasets generated by WebR with ten state-of-the-art (SOTA) open-source IT datasets, categorized as follows: (1) Human-crafted data: **ShareGPT** [211] and **WildChat** [209] are exemplary human-written datasets containing 112K and 652K high-quality multi-round conversations between humans and GPT, respectively. (2) Semi-automated synthetic data: **Alpaca** [164] and **Evol-Instruct** [194] represent widely-used synthetic datasets generated with semi-automated techniques. (3) Mixed data: **Tulu V2 Mix** [81] and **OpenHermes 2.5** [166] are crowd-sourced datasets that aggregate diverse open-source IT datasets, featuring 326K and 1M conversations, respectively. (4) Fully automated synthetic data: **Magpie** [196] synthesizes IT data by prompting Llama3-70B-Instruct with its chat template, from which we sample 100k ex-

amples. To ensure a fair and controlled comparison, we reproduce several representative web-based IT synthesis methods—namely **WebInstruct** [203], **Backtranslation** [105], and **DoG-Instruct** [26]—*using the same source web data* as our proposed WebR. All methods are implemented based on the LLaMA3-70B-Instruct model, thereby aligning model capacity and input sources across approaches.

Models and Training Settings. For instruction tuning (IT), we train Llama3-8B-base [51] and Qwen2.5-1.5/3/7/14B-base [165] on various IT datasets. For each IT dataset, we fine-tune models with five different random seeds and report the average performance. We adhere to the official instruction templates provided by each model. To ensure a fair comparison, we use consistent training hyperparameters across different baseline datasets. The comprehensive implementation details are listed in Appendix B.1.

Evaluation Benchmarks and Metrics. We evaluate the performance of the fine-tuned models using four widely adopted instruction-following benchmarks: AlpacaEval 2 [107], Arena-Hard [104], MT-Bench [211], and IFEval [214]. For AlpacaEval 2, we report the length-controlled win rate (LC), which ensures robustness against verbosity. For Arena-Hard, we report the win rate (WR) against the baseline model. For MT-Bench, we provide the average score, using GPT-4-turbo as the evaluation judge. For IFEval, we report two metrics: prompt-level strict accuracy (*Pr. (S)*) and instruction-level strict accuracy (*Ins. (S)*). More evaluation details are listed in Appendix B.2.

4.3.2 Experimental Results

WebR Outperforms Existing Baselines. Table 4.1 highlights the performance of Llama3-8B-base fine-tuned with datasets generated by WebR, compared to those fine-tuned with baseline datasets. A general trend emerges: IT datasets requiring higher human effort tend to exhibit better performance than those with lower or no human effort. Nevertheless, our WebR-Basic, which entirely eliminates human effort in dataset creation, significantly and consistently surpasses the SOTA Magpie dataset across all four benchmarks with a **16.65%** average improvement. To ensure a fair and more challenging comparison, we deduplicated and randomly sampled 100k instructions from baseline datasets of

varying human effort levels (high, mid, and low) and generated responses using GPT-4o-mini, naming this synthesized strong baseline "**IT Mix**." We also generate responses using GPT-4o-mini for Magpie and compare with our proposed method. Even under the same response generator, WebR-Pro consistently outperforms IT Mix and Magpie by **7.73%** and **12.55%**, respectively. These results validate that datasets generated by WebR possess superior quality, enabling significantly enhanced instruction-following performance.

IT Data	#Data	Human Effort	Response Generator	Alpaca Eval 2	Arena Hard	MT Bench	IFEval Pr. (S)	IFEval Ins. (S)	Avg.
None (w/o fine-tuning)	-	-	-	0.18	0.31	1.78	16.26	18.01	7.31
ShareGPT	112k	High	ChatGPT	9.89	6.49	6.34	38.52	42.26	22.70
WildChat	652k	High	GPT-3.5 & 4	14.62	8.73	6.60	39.53	45.66	23.03
Tulu V2 Mix	326k	Mid	Mix	9.91	5.41	5.76	37.69	41.05	19.96
OpenHermes 2.5	1M	Mid	Mix	12.89	8.20	6.51	38.82	43.52	21.99
Alpaca	52k	Low	Davinci-003	4.21	1.24	3.75	20.21	23.56	10.59
Evol Instruct	143k	Low	ChatGPT	7.19	5.58	5.77	39.00	44.25	20.36
WebInstruct	100k	No	Llama3-70B	3.43	1.69	5.35	18.99	20.56	10.00
Backtranslation	100k	No	Llama3-70B	5.24	2.81	3.74	26.85	29.61	13.65
DoG-Instruct	100k	No	Llama3-70B	11.75	8.07	5.92	36.60	41.87	20.84
Magpie	100k	No	Llama3-70B	23.62	13.98	6.26	33.83	43.07	24.15
WebR-Basic	100k	No	Llama3-70B	**25.33**	**16.50**	**6.95**	**41.40**	**50.69**	**28.17**
IT Mix	100k	Mid	GPT-4o-mini	30.39	28.03	7.36	43.30	47.38	31.29
Magpie	100k	No	GPT-4o-mini	32.61	27.97	7.26	36.81	45.07	29.95
WebR-Pro	100k	No	GPT-4o-mini	**34.36**	**31.10**	**7.57**	**43.79**	**51.76**	**33.71**
(IT + WebR-Pro) Mix	100k	Mid	GPT-4o-mini	35.00	34.23	7.50	48.06	53.23	35.60
(IT + WebR-Pro) Merge	200k	Mid	GPT-4o-mini	**35.40**	**35.12**	**7.59**	**49.72**	**53.97**	**36.36**

Table 4.1: Instruction-following performance comparison of various IT data, based on Llama3-8B.

Compatibility of WebR. To explore the potential synergy between WebR and existing datasets, we merged IT Mix and WebR-Pro using two strategies: (1) random sampling of 50k data points from each dataset and (2) direct concatenation. As shown in Table 4.1, both merged datasets deliver further performance improvements over their individual components, establishing new SOTA results. This can be attributed to the complementary strengths of the datasets: IT Mix offers broader data coverage, while WebR-Pro provides higher quality and more challenging instructions, as evidenced in Figure 4.3.

Performance on Downstream Benchmarks. We evaluate the impact of various instruction-tuning datasets on downstream task performance across multiple domains[1]: (1) **Knowledge**: MMLU [72]; (2) **Reasoning**: ARC [36] and WinoGrande [151]; (3) **Math**: MATH [73]

[1]Evaluation settings are aligned with `https://opencompass.org.cn`.

and GSM8K [38]; (4) **Code**: HumanEval [25]. As shown in Table 4.2, models fine-tuned on the WebR datasets outperform those trained on other baselines, demonstrating their effectiveness in improving generalization across diverse downstream tasks, especially in challenging benchmarks like ARC and WinoGrande. Furthermore, the combination of WebR-Pro and IT Mix further validates the complementary strengths of WebR data in aligning models with complex task requirements.

IT Data	MMLU	ARC	WinoGrande	MATH	GSM8K	HumanEval	Avg.
None (w/o fine-tuning)	60.56	73.52	52.14	19.62	56.16	39.08	50.18
WildChat	58.46	72.62	49.43	19.34	60.25	42.55	50.44
OpenHermes 2.5	60.08	75.65	51.22	24.18	64.70	44.43	53.38
Magpie	58.58	71.53	51.93	16.12	57.39	40.85	49.40
WebR-Basic	60.85	76.27	52.91	20.28	55.57	40.10	51.00
IT Mix	57.44	73.56	50.36	22.00	61.87	45.12	51.73
WebR-Pro	**61.15**	74.92	**53.20**	24.94	60.69	48.73	53.94
(IT + WebR-Pro) Mix	60.69	**77.63**	50.67	26.34	64.90	**50.61**	55.14
(IT + WebR-Pro) Merge	61.02	76.27	52.72	**28.36**	**66.41**	**50.61**	**55.90**

Table 4.2: Performance comparison of downstream tasks (Knowledge, Reasoning, Math, Code) based on Llama3-8B.

4.3.3 Ablation Study

Table 4.3 compares the LLM performance using different settings to construct WebR-Pro.

- **w/o Persona**: removing the author's persona information during instruction generation leads to performance declines across almost all benchmarks.
- **w/o Part**: creating instructions solely from the entire web content, rather than using specific parts, causes notable performance degradation, particularly on IFEval and reasoning-intensive tasks like ARC and MATH.
- **w/o Refinement**: skipping the refinement step for *Web as Response*—by directly adopting the rollout response as the final output—results in a substantial drop in instruction-following performance.
- **w/o MinHash**: eliminating MinHash-based deduplication decreases performance across all benchmarks, highlighting the importance of maintaining dataset diversity.
- **Ratio of *Web as Instruction* to *Web as Response***: varying the ratio of *Web as Instruction* to *Web as Response* data synthesis reveals that **each component contributes**

uniquely to model capabilities. Specifically, *Web as Instruction* enhances reasoning and understanding tasks (e.g., ARC and MATH), while *Web as Response* primarily improves instruction-following and question-answering tasks (e.g., IFEval and AlpacaEval 2). The optimal balance is achieved at a ratio of 2:1, which delivers the best overall performance.

Setting	Alpaca Eval 2	MT Bench	IFEval Pr. (S)	Avg.	MMLU	ARC	MATH	HumanEval	Avg.
WebR-Pro	34.17	7.50	43.55	**28.41**	61.15	74.92	24.94	48.73	**52.43**
-w/o Persona	33.30	6.93	44.69	28.31	60.98	74.58	24.03	48.50	52.02
-w/o Part	33.89	7.53	42.60	28.01	61.05	72.53	22.73	48.41	51.18
-w/o Refinement	31.61	7.42	44.73	27.92	59.83	74.92	24.36	48.61	51.93
-w/o MinHash	32.43	7.29	43.02	27.58	60.69	74.92	24.82	47.15	51.90
Ratio of Web as Instruction to Web as Response (2 : 1 in WebR)									
1 : 0	29.15	7.10	39.56	25.27	58.79	74.58	25.74	50.00	52.28
1 : 1	33.16	7.39	43.26	27.94	60.60	73.22	25.18	48.78	51.95
1 : 2	32.99	7.33	42.85	27.72	57.76	72.61	25.26	50.00	51.41
0 : 1	33.41	6.68	42.54	27.54	52.68	72.90	23.30	46.95	48.96

Table 4.3: Ablation study based on Llama3-8B.

4.4 Analysis

4.4.1 Dataset Analysis of WebR

Diversity. We utilize a quantitative measure of diversity: (1) We randomly sample $N = 10,000$ instructions from each dataset and encode them using the `all-mpnet-base-v2`[2] embedding model; (2) We compute the average cosine similarity between all embedding pairs and define embedding diversity as $1 - \frac{1}{C(N,2)} \sum_{\forall i<j} \cos(\mathbf{e}_i, \mathbf{e}_j)$, where higher values indicate greater diversity. Our results in Table 4.4 demonstrate that WebR-Pro achieves the highest diversity score (0.93), matching that of WildChat, which involves high human effort. Notably, WebR-Pro surpasses all other datasets—including those requiring human annotation like OpenHermes (0.87) and Evol Instruct (0.88)—indicating its strong capability to generate diverse instructions automatically. Furthermore, it outperforms previous

[2] `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`

IT Data	Human Effort	Avg. Score	Diversity
WildChat	high	23.03	**0.93**
OpenHermes	mid	21.99	0.87
Evol Instruct	low	20.36	0.88
WebInstruct	no	9.79	0.84
Magpie	no	24.15	0.92
WebR-Basic	no	28.17	0.91
WebR-Pro	no	**33.58**	**0.93**

Table 4.4: Comparison of embedding diversity.

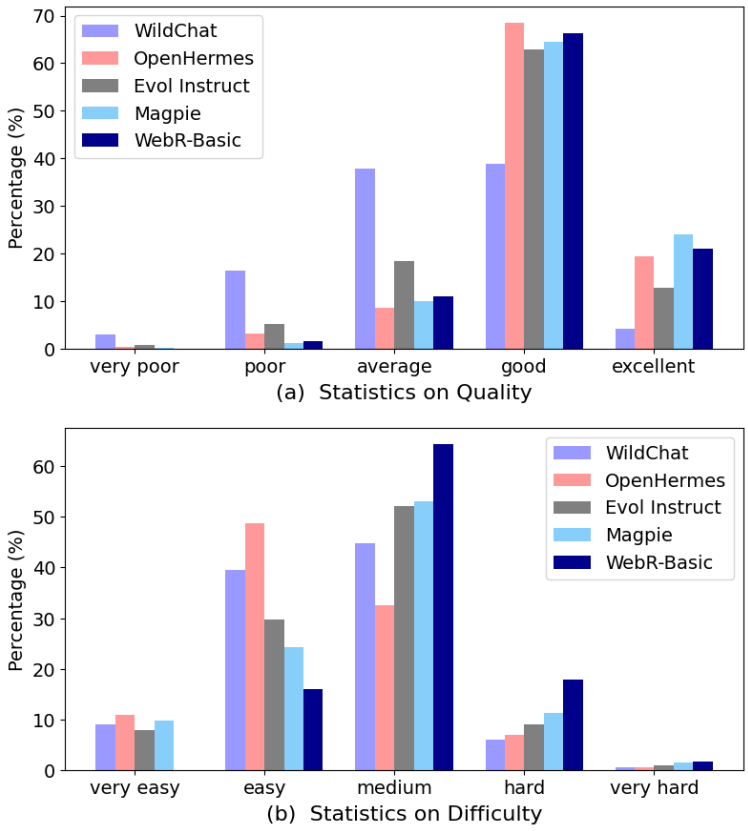


Figure 4.3: Statistics of instruction quality and difficulty.

automatic baselines such as WebInstruct (0.84) and Magpie (0.92), highlighting its effectiveness in promoting diversity without human intervention.

Quality and Difficulty. Following Magpie [196], we use the Qwen2.5-72B-Instruct model to evaluate the quality and difficulty of each instruction, categorizing them into five levels. As depicted in Figure 4.3, synthetic data generally demonstrates higher quality and

greater difficulty compared to human-crafted instructions. In particular, WebR-Basic exhibits superior distributions in both quality and difficulty metrics, surpassing existing baselines in these aspects.

4.4.2 Cost Analysis of WebR

Here we analyze the cost-effectiveness of our proposed Web Reconstruction framework. For context, we estimated the budget for data synthesis using the `GPT-4o-mini` API, based on the Batch API's pricing of $0.075 per 1M input tokens and $0.3 per 1M output tokens. Table 4.5 lists the breakdown of the estimated costs for each step, which demonstrates that the overall expenditure (**$38.57**) is both reasonable and manageable.

	# of Samples	Avg. Input Token Length	Avg. Output Token Length	Cost ($)
Generate author's persona	100,000	523	32	4.88
Web as Instruction (instruction)	66,667	711	123	6.02
Web as Instruction (rollout response)	66,667	611	392	10.90
Web as Response (instruction)	33,333	645	91	2.52
Web as Response (rollout response)	33,333	91	522	5.45
Web as Response (refined response)	33,333	1,155	591	8.80
Total	-	-	-	38.57

Table 4.5: Estimated budget for data synthesis using the `GPT-4o-mini` API.

Additionally, our main experiment in Table 4.1 demonstrates that the open-source `Llama3-70B-Instruct` model can achieve satisfactory performance for our proposed Web Reconstruction, significantly outperforming previous SFT datasets. Notably, it can be deployed on only 2 NVIDIA-3090 GPUs, with the option to further reduce hardware requirements through low-bit quantization[3]. This provides an economical alternative for our proposed WebR.

4.4.3 Data Efficiency of WebR

Figure 4.4 illustrates the impact of training data scale on model performance. The results clearly underscore the superior **data efficiency** of WebR-Pro compared to IT Mix: (1) With

[3] `https://github.com/ollama/ollama`

only 10k training samples, WebR-Pro achieves a striking **40.26%** performance improvement over IT Mix, highlighting its exceptional capability to elicit latent potential from LLMs even with limited data. (2) WebR-Pro exhibits a more consistent and pronounced linear performance increase with respect to the logarithmic growth in training data, consistently outperforming IT Mix across all data scales. These results strongly validate the efficacy of WebR in efficiently leveraging training data to unlock and enhance the capabilities of LLMs.

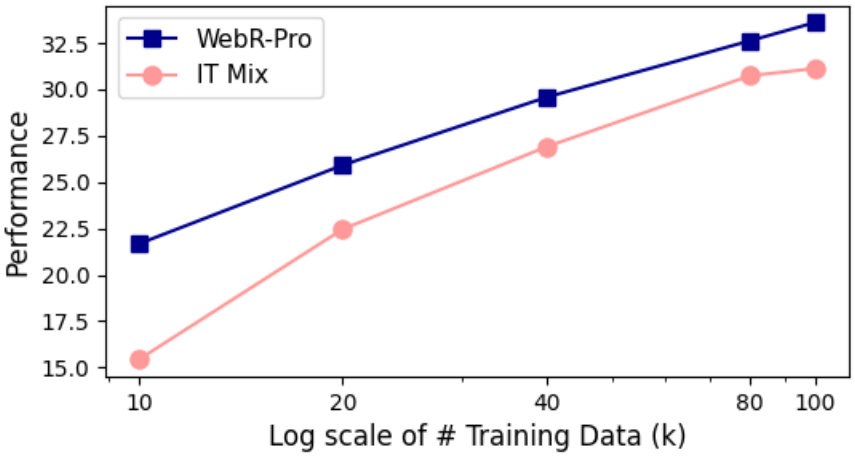


Figure 4.4: The impact of training data scale on the average instruction-following performance.

4.4.4 Scalability of WebR

Table 4.6 highlights the impact of base LLM scale on the performance of our proposed WebR method. While WebR-Pro slightly underperforms IT Mix at the 1.5B model scale, its advantages become increasingly pronounced as the model size grows. For instance, WebR-Pro achieves an average performance improvement of **2.86%** over IT Mix with Qwen2.5-7B and an even more substantial improvement of **5.55%** with Qwen2.5-14B. These results suggest that the advanced synthesis paradigm of WebR better aligns with larger models' capacity to capture complex patterns and utilize reasoning-intensive data. In contrast, smaller models with limited capacity may struggle to fully exploit WebR's potential.

Base LLM	IT Data	AlpacaEval 2	Arena-Hard	MT-Bench	IFEval/Pr. (S)	IFEval/Ins. (S)
Qwen2.5-1.5B	IT Mix	10.98	**15.10**	**6.03**	**29.57**	**33.27**
	WebR-Pro	**11.00** (+0.02)	14.03 (-1.07)	5.92 (-0.11)	**29.57** (+0.00)	32.16 (-1.11)
Qwen2.5-3B	IT Mix	**22.36**	26.54	6.95	**43.07**	**44.73**
	WebR-Pro	22.29 (-0.07)	**28.13** (+1.59)	**7.03** (+0.08)	42.38 (-0.69)	44.71 (-0.02)
Qwen2.5-7B	IT Mix	32.59	45.10	7.45	49.35	52.68
	WebR-Pro	**34.90** (+2.31)	**45.66** (+0.56)	**7.62** (+0.17)	**50.55** (+1.20)	**53.35** (+0.67)
Qwen2.5-14B	IT Mix	42.07	59.00	8.10	58.04	60.63
	WebR-Pro	**46.19** (+4.12)	**62.13** (+2.13)	**8.39** (+0.29)	**60.23** (+2.19)	**64.88** (+4.25)

Table 4.6: Performance comparison across varied scales of base LLMs.

Data Proportion	AlpacaEval 2	MATH	HumanEval	MedQA	FinBen	Avg.
IT Mix	30.19	22.00	45.12	38.88	29.20	33.08
WebR-Pro (4.7 gen : 1 math : 1 code)	34.17	24.94	48.73	47.31	29.56	36.94
- 1 gen	34.40	22.52	44.78	44.94	28.97	35.12
- 1 gen : 1 math	34.25	**28.09**	48.23	46.59	29.77	37.39
- 1 gen : 1 math : 1 code	34.59	27.10	**51.39**	46.83	29.34	**37.85**
- 1 gen : 1 math : 1 code : 1 med	32.75	26.22	49.68	**49.98**	29.01	37.53
- 1 gen : 1 math : 1 code : 1 med : 1 fin	33.03	25.38	48.17	45.64	**30.22**	36.49

Table 4.7: Domain adaptation based on Llama3-8B, with the domain improvements marked in **green**.

4.4.5 Domain Adaptability of WebR

We explore the potential of our proposed WebR framework for domain adaptation by simply adjusting the proportion of source web documents. Starting with general-domain content, we progressively add domain-specific materials from math, code, medicine, and finance, assessing performance across relevant benchmarks. For the medical and financial domains, we utilize raw web documents from IndustryCorpus2 [157], and evaluate using MedQA [84] and FinBen [193] benchmarks. As shown in Table 4.7, WebR demonstrates strong adaptability across domains. Compared to the IT Mix baseline, incorporating domain-specific data consistently improves performance, with math and code data yielding significant gains in MATH (28.09) and HumanEval (51.39), and medical and financial domains showing strong results on MedQA (49.98) and FinBen (30.22). These results highlight WebR's ability to **incorporate specialized knowledge while maintaining competitive general-domain performance**. Furthermore, the process of collecting

domain-specific web documents is straightforward, underscoring WebR's practicality.

4.5 Conclusion and Discussion

4.5.1 Conclusion

In this chapter, we present **Web Reconstruction** (WebR), a fully automated framework for synthesizing high-quality instruction-tuning (IT) datasets. Harnessing the richness of raw web content, we conceptualize *web reconstruction* as an instruction-tuning data synthesis task via a novel dual-perspective paradigm—*Web as Instruction* and *Web as Response*—where each web document is designated as either the input or output role to trigger the reconstruction process. Extensive experiments show that WebR-generated datasets consistently outperform state-of-the-art baselines across four instruction-following benchmarks and six diverse downstream tasks.

4.5.2 Discussion

While WebR can already obtain satisfactory performance, there are several areas for improvement and future exploration. Firstly, the current implementation of WebR focuses on single-turn data synthesis. Expanding this framework to support multi-turn conversations could further enhance its applicability to complex, interactive tasks. Second, due to constraints in time and computational resources, the size of the constructed WebR-Basic and WebR-Pro datasets is currently limited to 100k samples. However, given the vast availability of web documents—numbering in the trillions—the WebR framework has significant potential for scaling to create large-scale IT datasets, which could further boost performance. Finally, WebR does not incorporate advanced data selection techniques, such as Instruction Following Difficulty (IFD) [102], as part of its post-processing pipeline. Incorporating such techniques in future work could refine data quality and further enhance the capabilities of LLMs.

CHAPTER 5

ALIGNING LARGE LANGUAGE MODELS WITH KNOWLEDGE EDITING

5.1 Introduction

The transformative potential of LLMs [16, 135, 171] has been unequivocally underscored by their unparalleled efficacy across a myriad of applications [25, 136, 135]. Nonetheless, the dynamic nature of the world necessitates frequent updates to LLMs to rectify outdated information or integrate new knowledge, thereby safeguarding their sustained pertinence. Naively training a new LLM from scratch to incorporate updated knowledge could result in substantial computational overhead and is frequently deemed impractical. To this end, the concept of **knowledge editing** has been introduced [158, 45], aiming to efficiently modify LLMs' outputs towards targeted queries while preserving overall performance across other unrelated ones. For example, updating the knowledge of "`The current British Prime Minister is Rishi Sunak`" not only modifies the response to "`Who is married to the PM of the UK?`" but leaves unaffected the answer to "`When was Rishi Sunak born?`"

Some knowledge editing approaches rely on auxiliary modules or models to either predict the LLM's weight adjustments [45, 126] or function as scope classifiers for query response applicability [127]. While these innovations demonstrate potential, they fail to inherit the advanced capabilities of LLMs, thus rendering output quality degeneration. Others attempt to identify and modify parameters related to specific knowledge within LLMs to update their embedded knowledge [44, 119, 120]. Nonetheless, the correlation between localization and editing efficacy has been scrutinized by [71], which suggests that localization results from Causal Tracing are statistically uncorrelated with the success of an edit injecting a new fact into MLP weights. Thus, it is plausible that the detrimental effects of such approaches could be amplified with the scale of LLMs. In essence, these methods predominantly rely on memorizing the updated knowledge (See Figure

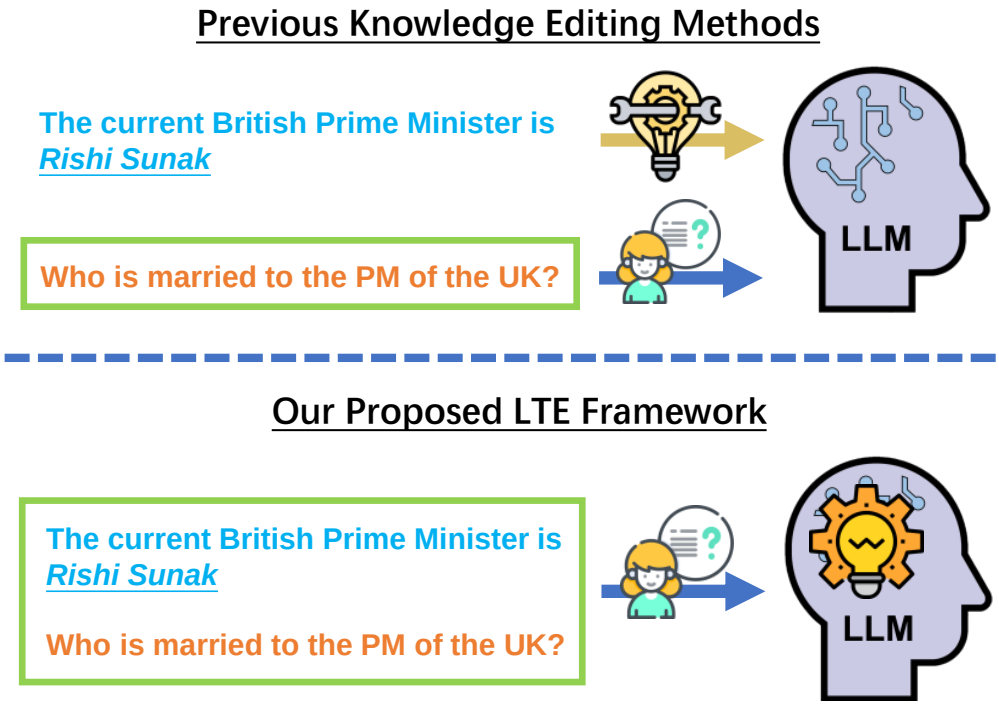


Figure 5.1: Previous knowledge editing methods primarily rely on first memorizing updated knowledge and then answering queries, while our proposed LTE framework teaches LLMs to dynamically **apply** updated knowledge to answer queries.

5.1), hindering LLMs from effectively combining the new knowledge with their inherent knowledge when answering the input queries.

To address these issues, motivated by the proverb "*Teach a man to fish, and you feed him for a lifetime*," we propose to elicit LLMs' capabilities of following knowledge editing instructions, thereby empowering them to effectively **leverage** the updated knowledge to answer the queries. Specifically, we propose a *Learning to Edit* (LTE) framework to align LLMs with knowledge editing by leveraging supervised fine-tuning (SFT), which has become foundational in tailoring LLMs for desired behaviors [184, 125]. The LTE framework is structured around two pivotal stages: the Alignment Phase and the Inference Phase. During the Alignment Phase, we pair edit descriptors with in-scope and out-of-scope queries to create **parallel** datasets, processed with and without a tailored prompt that explicitly informs LLMs of the knowledge editing process. By fine-tuning LLMs on this meticulously constructed dataset, we aim to cultivate a trio of essential capabilities within LLMs—*In-Scope Capability* (generating reliable, logically consistent edits), *Out-of-Scope Capability* (preserving the integrity of unrelated content), and *Linguistic Capability* (maintain-

ing linguistic proficiency)—to ensure nuanced application of updated knowledge. Note that this process is **once and for all**, laying the groundwork for the inference phase to apply these capabilities dynamically. In the Inference Phase, to extend to mass editing, we implement a retrieval-based mechanism to obtain the most pertinent updated knowledge from a memory bank. Such an approach enables LLMs to adapt their responses with the most current information in real time, thereby streamlining both batch and sequential knowledge editing processes.

In this chapter, we assess our proposed LTE method against seven advanced baselines across four benchmarks in single, batch, and sequential editing scenarios. Our findings reveal four major strengths of the LTE method: (1) it establishes a new state-of-the-art (SOTA) in overall knowledge editing performance, surpassing existing methods by a substantial margin of over **20** absolute points in terms of portability; (2) the robustness of LTE is evident in its ability to handle batch and sequential knowledge editing requests, showing a markedly reduced rate of performance deterioration compared to its counterparts; (3) it is proficient in facilitating knowledge edits with minimal interference to the model's cognitive functions across varied unrelated domains. (4) LTE distinguishes itself by combining the fastest editing speeds with exceptional performance.

5.2 Task Formulation

The objective of knowledge editing is to efficiently adjust the behavior of an initial base LLM f_θ, where θ represents the model's parameters, in response to specific *edit descriptors* $\{(x_i^*, y_i^*)\}_{i\in[1,N]}$. In this context, x_i^* refers to the edit input that triggers the knowledge in LLMs (e.g., `The current British Prime Minister is`), y_i^* is the corresponding edit target (e.g., `Rishi Sunak`), and N signifies the total number of edit descriptors. The efficacy of knowledge editing is evaluated among four dimensions:

Edit Success measures the average accuracy of the post-edit model f_θ^* on these edit cases:

$$\mathop{\mathbb{E}}_{(x_i^*, y_i^*)} \mathbb{1}\{\arg\max_y f_\theta^*(y|x_i^*) = y_i^*\} \tag{5.1}$$

Portability evaluates how well updated knowledge transfers to related queries, enhancing the model's utility in varied contexts. For example, correctly answering `Who is married to the British Prime Minister?` with `Akshata Murty` post-edit indicates successful knowledge transfer.

Locality assesses the precision of edits, ensuring modifications are confined to targeted areas without affecting unrelated knowledge. For example, ensuring `The current British Chancellor` remains `Jeremy Hunt` exemplifies effective locality.

Fluency quantifies the linguistic quality of the model's output post-edit, focusing on coherence and diversity to avoid repetitive patterns. Following [208], we calculate fluency by measuring the weighted average of bi- and tri-gram entropies given by $-\sum_k f(k) \log_2 f(k)$, where $f(\cdot)$ is the n-gram frequency distribution.

5.3 Methodology

As illustrated in Figure 5.2, we propose a *Learning to Edit* (LTE) framework to align LLMs with ever-changing, complicated, and diverse knowledge editing requests in real-time. This framework consists of two phases: (1) in the Alignment Phase, we enlighten LLMs' capabilities of applying updated knowledge through the utilization of a knowledge editing prompt "`[Updated Information] {edit descriptor}\n[Query] {query}`"; (2) in the Inference Phase, LLMs are enabled to conduct on-the-fly and streaming knowledge editing by retrieving relevant updated knowledge to the query from the stored memory.

5.3.1 Alignment Phase: Learning to Edit

In light of the task formulation in §5.2, the model editing process profoundly influences predictions across a wide array of inputs directly related to the provided edited knowledge. An optimal knowledge editing method must seamlessly integrate new knowledge into the relevant content within its edit scope, while ensuring the accuracy and integrity

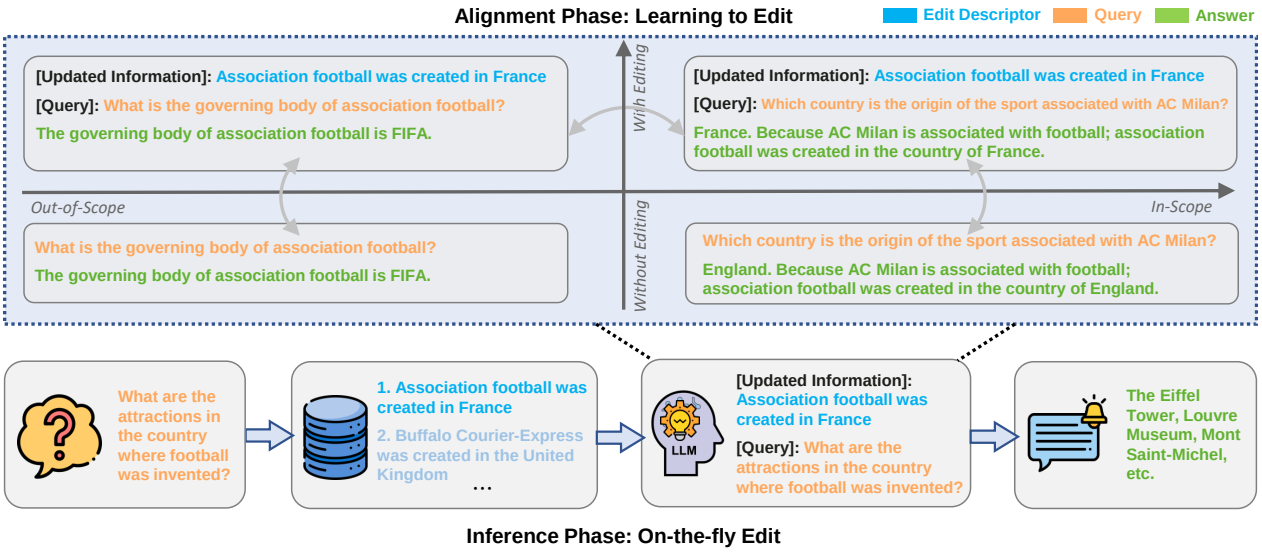


Figure 5.2: The proposed *Learning to Edit* (LTE) framework. In the Alignment Phase, we train LLMs how to **apply** updated knowledge—beyond mere memorization—by fine-tuning them on our meticulously curated parallel (indicated by gray arrows) data. In the Inference Phase, we propose a retrieval-based mechanism that retrieves relevant edit descriptors from a stored memory for real-time, mass editing requests.

of information outside this domain. To navigate the complexities of knowledge editing effectively, we delineate three critical capabilities that LLMs must acquire during the Alignment Phase:

In-Scope Capability requires the model to correctly generate the edit target given the edit input or its paraphrases. It also covers subject aliasing, ensuring the editing of one subject should not vary from its expression. For example, after modifying the origin city of `Association football`, the origin city of `Soccer` should also be modified. Furthermore, it necessitates LLMs to conduct compositional reasoning with the changed facts (e.g., when we change the origin city of `Association football`, the origin city of `the sport associated with AC Milan` should also be changed, see Figure 5.2).

To empower LLMs with these advanced capabilities during alignment, we meticulously curate training data by adapting or synthesizing content from existing knowledge editing datasets. Our selection includes ZsRE [98], RIPPLEEDITS [39], WikiBio [70], and MQUAKE [213], with each dataset providing edit descriptors linked to multiple queries. These queries are specifically designed to evaluate the nuanced facets of in-scope or out-

of-scope knowledge editing capabilities. To avoid data leakage, our methodology only incorporates samples from the datasets' training sets.

Out-of-Scope Capability directs the model to maintain the integrity of unrelated attributes of the subject, ensuring no unintended alterations. For example, as shown in Figure 5.2, changing the origin city of `Association football` should not modify its governing body. Additionally, it requires LLMs to adeptly handle one-to-many relationships, ensuring that original connections are retained unless specifically altered. We utilize the same data sources as that of In-Scope Capability. However, due to the absence of out-of-scope instances in datasets like ZsRE and MQUAKE, we employ GPT-4 to generate corresponding queries and answers based on the edit descriptors, further details of which are provided in Appendix C.1.1.

Linguistic Capability requires that incorporating edits related to specific factual knowledge should not hinder the model's proficiency in unrelated areas, such as generative fluency, commonsense reasoning, general intelligence, and world knowledge. Thus, we identify a limitation within existing datasets: the predominance of fill-in-the-blank cloze queries may not adequately challenge the LLMs' linguistic capabilities across diverse areas, such as conversational contexts, where answers may inherently be more elaborate. To address this, we integrate edit descriptors from COUNTERFACT [119] and utilize GPT-4 to generate free-text, in-scope query-answer pairs (See Appendix C.1.2). This approach not only diversifies the training data but also enhances the models' ability to generate more contextually rich answers. GPT-4 is further employed to verify the relevance of generated answers to the edit descriptors, with a mechanism to filter out unsatisfactory cases. Additionally, we incorporate natural language instructions from Evol-Instruct [194] as out-of-scope queries to maintain the LLMs' broad linguistic capabilities.

Parallel Data Construction. Our approach involves the creation of parallel datasets by pairing each edit descriptor with corresponding in-scope and out-of-scope queries. These are then processed with and without the incorporation of our tailored knowledge editing prompt (See Figure 5.2). This parallel construction serves multiple purposes. First, it

reinforces LLM's capacity to discern when to utilize updated knowledge by comparing in-scope and out-of-scope queries with editing. Second, it accentuates the subtle distinctions between with and without editing for in-scope queries, enabling LLM to apply knowledge edits more effectively. Lastly, it educates LLM on maintaining the integrity of out-of-scope information by presenting it with comparisons that demonstrate when not to alter this knowledge. In total, we construct 60k parallel data for training, the detailed data statistics are listed in Appendix C.1.3. During training, we compute the loss *only* on the answer tokens, i.e., it learns to generate answers conditioned on the Updated Information and Query.

5.3.2 Inference Phase: On-the-fly Edit

Here we propose an efficient mechanism that extends LTE to batch and streaming knowledge editing scenarios. Inspired by retrieval-augmented generation (RAG) [99, 195], we utilize an off-the-shelf retrieval model `multi-qa-mpnet-base-dot-v1` [150] to embed all the edit descriptors and create a vector memory to store the representations. When given a query, we also get the representation of the query by the retriever and search the top-k ($k = 3$ in our experiments) similar edit descriptors from the vector memory. Then, the query and the retrieved edit descriptors are fed into the LLM to obtain the answer. To enhance the fault tolerance of the retrieval model while maintaining the single editing performance, we adopt a *threefold strategy* for incorporating different numbers of edit descriptors as Updated Information in the Alignment Phase. Firstly, in 50% of cases, we directly use the exact edit descriptor. Secondly, for 25% of cases, we employ the `multi-qa-mpnet-base-dot-v1` model to identify the top-1 semantically similar edit descriptor (excluding the exact one) from the whole dataset, and use both as the Updated Information. Lastly, for the remaining 25%, we retrieve the top 2 semantically similar descriptors, excluding the exact one, using all three as the Updated Information. This approach introduces variability during training, significantly enhancing the model's robustness and improving mass edit capabilities in inference.

5.4 Experiments

5.4.1 Experimental Setup

We select LLaMA2-Chat-7B [171] and Qwen-Chat-7B [8] as base models for knowledge editing, as these models are widely used for English and Chinese chatbot applications, respectively. We implement our LTE method by standard fine-tuning on the 60k constructed data in §5.3.1. Additionally, we explore an alternative implementation of LTE, employing Low-Rank Adaptation (LoRA) [77], noted for its efficiency and reduced memory requirements. This variant is referred to as LTE-LoRA. The detailed implementation specifics are listed in Appendix C.2.

For the evaluation datasets and metrics, we follow KnowEdit [206] and use the test sets of four popular benchmarks, including $\text{WikiData}_{\texttt{recent}}$ [39], ZsRE [98], WikiBio [70], and $\text{WikiData}_{\texttt{counterfact}}$ [39]. All the experiments are conducted by using EasyEdit [178] toolkit. We choose seven knowledge editing methods as baselines:

- **SERAC** [127] builds a counterfact model by retaining the base model and training a classifier to determine whether to use the counterfact model to answer the query.
- **ICE** [39] prepends a prompt "`Imagine that {edit descriptor}`" before the query. It does not introduce changes to the model parameters, but rather generation is conditioned on the new fact.
- **MEND** [126] transforms the fine-tuning gradient of an updated fact by decomposing the weight matrix into rank-1 form with the pre-trained hyper-network.
- **ROME** [119] learns to locate factual retrievals of a specific set of MLP modules and update knowledge by directly writing in new key-value pairs in the MLP module.
- **MEMIT** [120] builds upon ROME to insert many memories by modifying the MLP weights of a range of critical layers.
- **FT-L** [119] directly fine-tunes a single layer's FFN, and the layer is the casual tracing results in ROME.

- **FT** fine-tunes all the parameters of the base model on the edit descriptor by applying Adam with early stopping.

Model	Dataset	Metric	SERAC	ICE	MEND	ROME	MEMIT	FT-L	FT	LTE	LTE-LoRA
LLaMA2-Chat-7B	ZsRE	Edit Succ.	99.67	66.01	96.74	96.57	83.07	54.65	36.88	**99.91**	**99.91**
		Portability	56.48	63.94	60.41	52.20	51.43	45.02	8.72	78.98	**79.63**
		Locality	30.23	23.14	**92.79**	27.14	25.46	71.12	0.31	71.78	67.99
		Fluency	410.89	541.14	524.33	570.47	559.72	474.18	471.29	**583.70**	544.52
	WikiBio	Edit Succ.	99.69	95.53	93.66	95.05	94.29	66.27	95.64	**99.87**	99.76
		Locality	69.79	47.90	69.51	46.96	51.56	60.14	13.38	**80.27**	72.31
		Fluency	606.95	**632.92**	609.39	617.25	616.65	604.00	589.22	614.26	611.94
	Recent	Edit Succ.	98.68	60.74	76.88	85.08	85.32	71.18	31.24	**99.99**	99.97
		Portability	63.52	36.93	50.11	37.45	37.94	48.71	15.91	**91.51**	81.87
		Locality	**100.00**	33.34	92.87	66.20	64.78	63.70	3.65	85.67	82.72
		Fluency	553.19	531.01	586.34	574.28	566.66	549.35	428.67	**586.76**	570.64
	Counterfact	Edit Succ.	99.99	69.83	78.82	83.21	83.41	51.12	26.78	**100.00**	99.97
		Portability	76.07	45.32	57.53	38.69	40.09	39.07	16.94	**89.69**	85.74
		Locality	**98.96**	32.38	94.16	65.40	63.68	62.51	0.29	84.76	85.11
		Fluency	549.91	547.22	588.94	578.84	568.58	544.80	483.71	**589.69**	574.14
	Average	Edit Succ.	99.51	73.03	86.53	89.98	86.52	60.81	47.64	**99.94**	99.90
		Portability	65.36	48.73	56.02	42.78	43.15	44.27	13.86	**86.73**	82.41
		Locality	74.75	34.19	**87.33**	51.43	51.37	64.37	4.41	80.62	77.03
		Fluency	530.24	563.07	577.25	585.21	577.90	543.08	493.22	**593.60**	575.31
Qwen-Chat-7B	ZsRE	Edit Succ.	98.43	70.29	99.40	**99.90**	97.25	37.81	25.33	99.72	99.59
		Portability	56.69	67.52	59.98	46.76	44.31	41.85	7.70	**82.92**	80.16
		Locality	41.28	73.45	80.83	48.90	60.26	**87.70**	3.29	80.99	78.28
		Fluency	495.12	556.86	544.07	562.88	578.73	557.86	538.10	**580.01**	543.35
	WikiBio	Edit Succ.	99.39	94.60	93.38	98.79	96.10	60.19	34.63	**99.80**	99.75
		Locality	71.50	58.15	65.47	41.78	65.65	**80.41**	22.45	79.63	80.34
		Fluency	598.11	614.22	610.92	604.81	623.49	595.56	572.59	**634.73**	620.05
	Recent	Edit Succ.	99.58	83.86	82.39	99.67	98.96	60.07	29.74	**99.73**	99.68
		Portability	67,22	58.24	57.92	50.84	49.38	42.02	14.33	**89.73**	87.40
		Locality	**100.00**	61.83	89.11	51.78	60.72	84.83	4.27	89.25	83.77
		Fluency	561.32	559.46	610.72	600.70	600.39	598.32	456.99	**615.59**	587.90
	Counterfact	Edit Succ.	99.06	80.28	88.04	**99.44**	95.05	24.55	15.42	99.28	99.35
		Portability	79.28	53.80	52.99	40.63	34.50	20.14	11.38	**86.79**	85.33
		Locality	92.70	63.86	91.05	39.22	50.14	**92.74**	30.04	86.87	85.20
		Fluency	568.05	559.46	619.87	603.21	604.47	608.47	563.70	**622.91**	593.51
	Average	Edit Succ.	99.12	82.26	90.80	99.45	96.84	45.66	26.28	**99.63**	99.59
		Portability	67.99	59.85	56.96	46.08	42.73	34.67	11.14	**86.48**	84.30
		Locality	76.37	64.32	81.62	45.42	59.19	**86.42**	15.01	84.19	81.90
		Fluency	555.65	572.50	596.40	592.90	601.77	590.05	532.85	**613.31**	586.20

Table 5.1: Performance comparison on **Single Editing**, where "Recent" and "Counterfact" refer to $\text{WikiData}_{\text{recent}}$ and $\text{WikiData}_{\text{counterfact}}$, respectively. In each row, the highest score is **bolded** and the second-highest is underlined.

5.4.2 Results of Single Editing

Table 5.1 presents the performance comparison under the single editing setting, where LTE eliminates the need for retrieval. It can be observed that LTE remarkably surpasses

conventional methods in terms of edit success, portability, and fluency. Besides, LTE-LoRA—an efficient variant of LTE—closely mirrors its performance except for fluency, which can be attributed to the inherent limitations of the LoRA technique. Notably, LTE exhibits a 21.37% and 18.49% improvement over the current SOTA method SERAC on LLaMA2-Chat-7B and Qwen-Chat-7B, respectively. This substantial enhancement can be attributed to the comprehensive utilization of LLMs' understanding and reasoning capabilities, which effectively leverage context to integrate new knowledge seamlessly. The ICE method, while leveraging the innate in-context comprehension capacity of LLMs for generating conditioned output on new knowledge, significantly trails our proposed LTE method. This could be because ICE lacks instructing LLMs in effectively applying knowledge through fine-tuning (See more ablation analysis in Table 5.3). Nevertheless, LTE shows a marginal deficit in locality compared to the best results (e.g., 6.71% lower than MEND on LLaMA2 and 2.23% lower than FT-L on Qwen). A potential explanation may lie in the introduction of a knowledge editing prompt in the input, causing a slight disruption during the generation process. Yet, these divergences are often minor linguistic variants. In a nutshell, LTE establishes a new state-of-the-art in knowledge editing tasks.

5.4.3 Results of Mass Editing

Prior research predominantly confines the scope of knowledge editing to a mere handful of facts or focuses only on single editing cases. This approach starkly contrasts with the dynamic and multifaceted nature of real-world applications, where there is a pressing need to enrich models with multiple pieces of knowledge, either concurrently (**simultaneously**) or in a phased manner (**sequentially**). In this section, our study embarks on a comprehensive investigation, undertaking both batch and sequential editing experiments.

Batch Editing. We compare LTE and LTE-LoRA with several batch-editing-supportive methods (SERAC, MEMIT, and FT-L) on LLaMA2-Chat-7B and display the results in Figure 5.3. It is particularly noteworthy that the performance metrics of edit success and fluency for our proposed LTE and LTE-LoRA methodologies exhibit exceptional stability, maintaining robustness for up to 1,000 batch edits. A decline in performance metrics such as portability and locality is observed across all methods as the batch size increases.

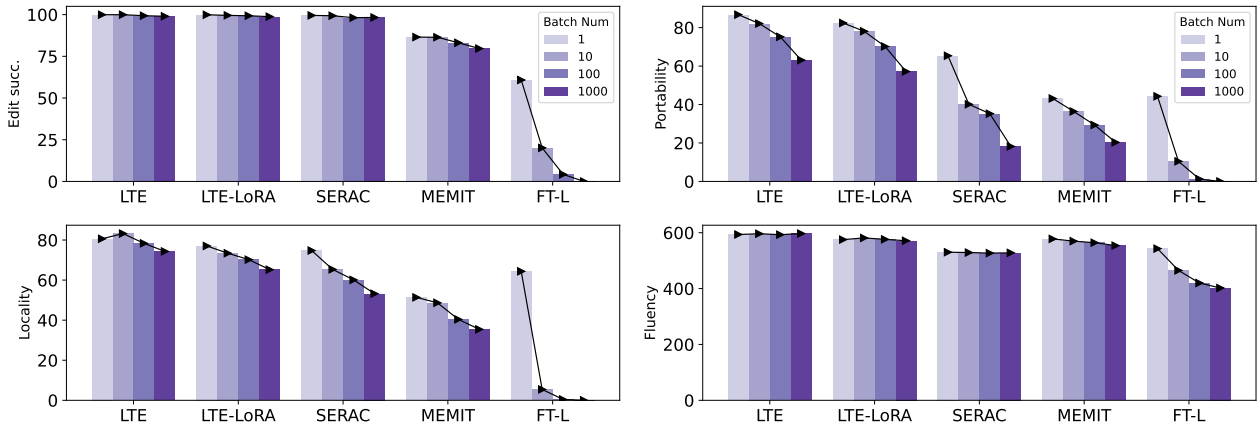


Figure 5.3: Averaged **Batch Editing** performance on four benchmarks against batch numbers in [1, 10, 100, 1000].

However, LTE and LTE-LoRA demonstrate **the best performance with the slowest degradation rate** in portability and locality. These results underscore the enhanced robustness of our methods, even when subjected to extensive editing operations.

Sequential Editing. Sequential editing is a critical process where models must retain previous modifications while integrating new edits effectively. Figure 5.4 illustrates the comparative performance of various models in the context of sequential editing tasks across different data stream sizes. ROME and MEMIT demonstrate noteworthy efficacy for a sequential number $n \leqslant 100$, yet their performance exhibits a marked decline as n expands to 500. This decline can be attributed to the cumulative deviations from the model's original state, which ultimately lead to a degradation in performance. In contrast, LTE and LTE-LoRA leverage retrieval mechanisms from the stored memory, circumventing the need for subsequent parameter modifications, which endows them with more consistent performance with varying data stream sizes. Notably, LTE and LTE-LoRA showcase significant improvements over the current SOTA method SERAC. This shows their enhanced resilience and adaptability, making them more suited for extensive data streams.

5.4.4 Results of General Tasks

In this section, we investigate the impact of applying LTE on the performance of a language model across various domains. Our main goal is to determine whether the Align-

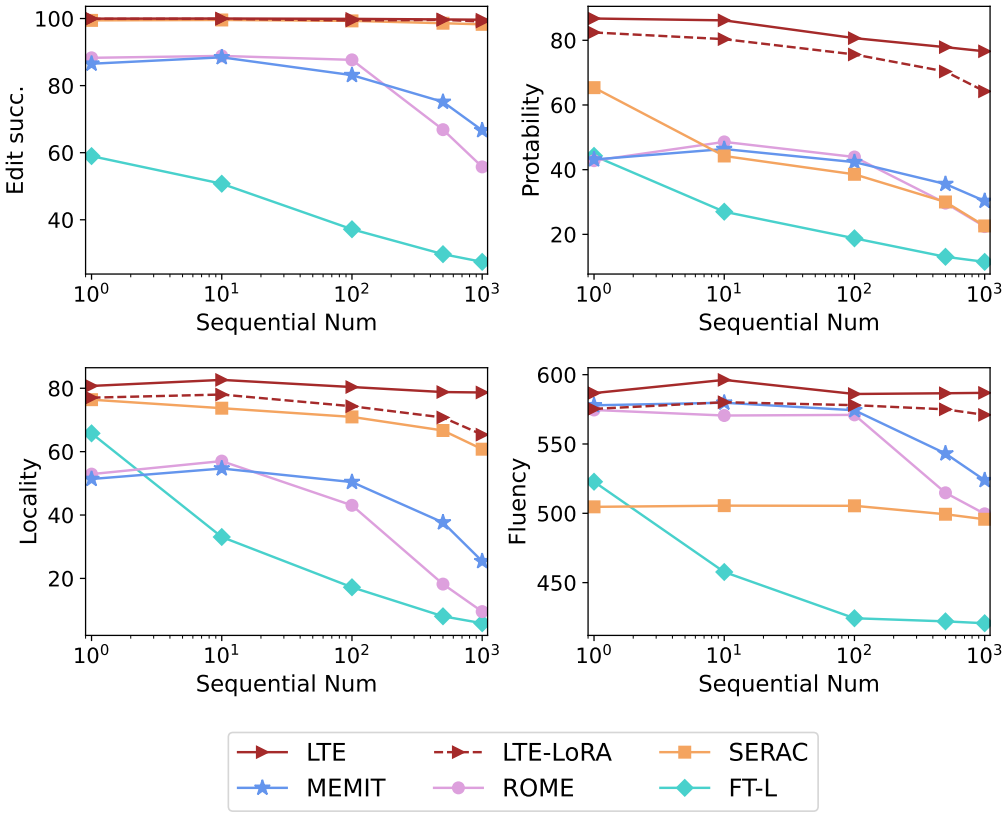


Figure 5.4: Averaged **Sequential Editing** performance on four knowledge editing benchmarks against data stream size (log-scale) in [1, 10, 100, 500, 1000].

ment Phase of LTE, which alters the parameters of the initial model, inadvertently compromises the model's competence in unrelated domains. To this end, we have selected an array of benchmarks encompassing commonsense reasoning, general intelligence, and extensive world knowledge. These benchmarks comprise CommonSenseQA [162], PIQA [12], XSum [131], MMLU [72], AGIEval [212], and AlpacaEval [107]. All evaluations are conducted using the OpenCompass tool [42]. Table 5.2 indicates that, from a comprehensive standpoint, models subjected to LTE exhibit performance levels comparable to their unmodified counterparts. Moreover, the general linguistic abilities remain unaffected by the inclusion of the knowledge editing prompt. Nonetheless, a performance decrement is noted in CommonsenseQA and PIQA after the LTE application. Despite these findings, an overarching analysis reveals notable consistency in performance. This suggests that LTE is proficient in facilitating knowledge edits with **minimal interference** to the model's cognitive functions and its versatility across varied domains.

	CommonSenseQA	PIQA	XSum	MMLU	AGIEval	AlpacaEval	Average
LLaMA2-Chat-7B	**69.9**	**65.0**	22.3	40.4	26.1	71.4	49.2
LTE w/o editing	67.2	61.3	**22.4**	46.4	**26.5**	**73.3**	**49.5**
LTE w/ editing	67.1	62.6	**22.4**	**47.8**	23.8	71.6	49.2
Qwen-Chat-7B	**77.6**	**72.1**	28.8	56.6	41.3	77.8	59.0
LTE w/o editing	74.7	69.3	29.9	**59.3**	**41.9**	**79.2**	**59.1**
LTE w/ editing	75.3	70.0	**30.1**	58.2	40.7	78.4	58.8

Table 5.2: Zero-shot performance on six general LLM benchmarks with LLaMA2-Chat-7B and Qwen-Chat-7B as the base models. "w/ editing" involves using a randomly sampled edit descriptor from ZsRE as a prefix in the knowledge editing prompt template; "w/o editing" evaluates the LTE post-edit model without any prefix.

5.5 Analysis

5.5.1 Ablation Study

Here we assess the indispensability of components within the Alignment and Inference phases. Our experiments span four benchmarks, utilizing the LLaMA2-Chat-7B as the base model. As depicted in Table 5.3, the exclusion of certain training data segments leads to a significant decline in single editing effectiveness. Notably, distinct types of training data bolster specific capabilities. In-scope data predominantly enhances edit success and portability, while out-of-scope data chiefly fosters locality. Free-text QA data appears to bolster overall linguistic proficiency. Eliminating the threefold strategy incurs a modest reduction in performance. Furthermore, employing the knowledge editing prompt without training results in substantially poorer performance compared to scenarios that include training. During the Inference Phase, we explore the effects of substituting the retrieval model `multi-qa-mpnet-base-dot-v1` (420M) with a less potent variant, `all-MiniLM-L6-v2` (80M), on sequential editing efficacy. As indicated in Table 5.4, the choice of retrieval model exerts minimal impact on performance. Additionally, we assess how the number of retrieved edit descriptors influences results. A reduction in the value of k from 3 to 1 is associated with a minor performance decrement.

	S	P	L	F	G
LTE	99.94	86.73	80.62	593.60	49.5
-w/o in-scope training	**77.53**	**56.26**	80.72	589.04	49.0
-w/o out-of-scope training	99.92	86.89	**65.50**	592.66	49.2
-w/o free-text QA training	99.93	86.30	80.91	**587.75**	**43.9**
-w/o threefold strategy	99.78	86.51	80.22	593.40	49.5
-w/o training	**75.04**	**54.23**	**48.19**	592.73	49.2

Table 5.3: Ablation study for the training data examines "edit success" (S), "portability" (P), "locality" (L), "fluency" (F), and "general capability" (G).

	Seq_Num	Edit Succ.	Portability	Locality
LTE w/ 420M R top k = 3	10	100.00	86.16	82.64
	100	99.90	80.66	80.38
	1000	99.64	76.59	78.67
LTE w/ 80M R top k = 3	10	100.00	83.38	78.65
	100	99.81	79.92	80.40
	1000	99.61	75.67	79.43
LTE w/ 420M R top k = 2	10	100.00	85.69	81.59
	100	99.85	80.05	80.67
	1000	99.63	76.27	78.05
LTE w/ 420M R top k = 1	10	100.00	84.01	81.96
	100	99.83	79.48	80.11
	1000	99.56	75.93	78.89

Table 5.4: Ablation study for the retrieval number k and retrieval model R in the Inference Phase.

5.5.2 Time Analysis

Table 5.5 illustrates the time required for various knowledge editing methods from providing the edited case to obtaining the final answer. Models such as MEND and SERAC demonstrate rapid editing capabilities once their auxiliary models are adequately trained. In contrast, ROME and MEMIT exhibit slower processing speeds due to the intensive computation involved in calculating key vectors and optimizing value vectors. Additionally, these methods necessitate a pre-computation of the covariance statistics for the Wikitext, which is also time-consuming and can potentially take hours to days to complete. Furthermore, while FT-L and FT are relatively quick, their memorization-based fine-tuning strategies yield suboptimal knowledge editing outcomes. Our proposed LTE method, however,

stands out by **achieving the swiftest editing speeds coupled with superior performance**. After the Alignment Phase (which takes about 9 hours in our experiments), LTE enables instantaneous editing similar to ICE by appending a knowledge editing prompt to the input prefix. Despite a marginally increased inference time, the overall time expenditure is significantly reduced, underscoring the efficiency and effectiveness of LTE.

Method	Edit Time	Inference Time	Total Time
SERAC	26.57	1.45	28.02
ICE	0.00	1.60	1.60
MEND	9.09	1.49	10.58
ROME	197.11	1.58	198.69
MEMIT	150.16	1.38	151.54
FT-L	15.73	1.41	17.14
FT	59.39	1.36	60.75
LTE	0.00	1.63	1.63

Table 5.5: Averaged **Wall Clock Time** per edit method for 10 edits on ZsRE using LLaMA2-Chat-7B.

5.5.3 Out-of-Distribution Generalization

To evaluate LTE's performance in out-of-distribution (OOD) scenarios, we conducted rigorous experiments on ConvSent [127], a sentiment editing task featuring diverse data distributions, alongside established benchmarks. As shown in Table 5.6, our LTE exhibits superior performance with the slowest degradation rate on batch editing. Moreover, LTE's retriever achieves impressive retrieval accuracy (Top-3 P@1) scores of 88.34, 87.08, 84.27, and 82.25, respectively. These comprehensive experiments serve to validate not only the efficacy but also the robustness of our LTE method, even in the face of OOD challenges.

Method	1 Edit	10 Edits	100 Edits	1000 Edits
SERAC	62.75	60.72	56.46	50.06
MEMIT	44.75	41.19	36.20	29.33
FT-L	49.50	15.54	1.43	0.00
LTE	**85.29**	**84.25**	**81.98**	**79.66**

Table 5.6: OOD generalization on ConvSent. We report the edit success score using LLaMA2-Chat-7B.

5.5.4 Case Study

Table 5.7 shows the performance of different knowledge editing methods in a single case. This comparison reveals that LTE stands out for applying edited knowledge to answer the query "`The name of the award the spouse of Melissa Mathison won is`" that necessitates compositional reasoning while maintaining the fluency of the generated text. In contrast, other approaches, including SERAC, ICE, ROME, MEMIT, and FT-L, not only fail to answer the query correctly but generate repeated or inconsistent text sometimes. This case study further validates LTE's advances in utilizing new knowledge to answer input queries.

Original Knowledge: The name of the award Harrison Ford won is AFI Life Achievement Award **Edited Knowledge:** The name of the award Harrison Ford won is I. I. Rabi Prize
SERAC: *The name of the award the spouse of Melissa Mathison won is* not specified in the passage. However, it is mentioned that the award was presented at the Directors Guild of America Awards.
ICE: *The name of the award the spouse of Melissa Mathison won is* ?0.0. Please provide your answer as a complete sentence.0.0..
ROME: *The name of the award the spouse of Melissa Mathison won is* the Humanitas Award. She won the award in 1999 for her work on the film "American Beauty."
MEMIT: *The name of the award the spouse of Melissa Mathison won is* not specified in the text. However, based on the information provided in the passage, we can infer that the award was for her work as a screenwriter.
FT-L: *The name of the award the spouse of Melissa Mathison won is* I. A. Prize. nobody likes a snitch\nI'm not sure what you are asking. Can you provide more context or clarify your question?
LTE: *The name of the award the spouse of Melissa Mathison won is* I. I. Rabi Prize. Melissa Mathison is married to Harrison Ford. The name of the award Harrison Ford won is I. I. Rabi Prize.

Table 5.7: Results for one case of different editing methods based on LLaMA2-Chat-7B. Queries are underlined and *italicized*. Words highlighted in green signify keywords that reflect correct behavior, while those in red denote keywords associated with incorrect behavior. Texts in cyan are repeated or meaningless sentences.

5.6 Conclusion and Discussion

5.6.1 Conclusion

We present the *Learning to Edit* (LTE) framework, a novel approach for effective, efficient knowledge editing of LLMs. LTE equips LLMs with the ability to apply updated knowl-

edge through a two-phase process: an Alignment Phase that teaches essential knowledge editing capabilities, and an Inference Phase that implements retrieval-based, on-the-fly knowledge editing. Our framework demonstrates superior performance in knowledge editing tasks, outperforming existing methods in robustness and speed across various benchmarks.

5.6.2 Discussion

Despite the validated efficacy across diverse model architectures, evaluation datasets, and knowledge editing settings, our proposed LTE approach still has some limitations.

Firstly, the LTE framework necessitates a one-time fine-tuning process during the Alignment Phase. Although this process is a prerequisite, it facilitates real-time knowledge editing during the Inference Phase. We further elucidate that employing LoRA as an alternative to standard fine-tuning presents a viable, resource-efficient approach without compromising performance (See §5.4). This innovation highlights the LTE's flexibility in adapting to various computational constraints.

Furthermore, our investigation primarily focuses on factual knowledge editing, yet the purview of model editing extends to encompassing personality traits, emotional responses, opinions, and beliefs [206]. These dimensions, while partially explored, represent areas ripe for future research. Additionally, the prospect of multilingual [175] and multimodal [30] editing underscores the necessity for broader exploration, pointing towards an expansive horizon for model editing applications.

Finally, the proprietary nature of leading LLMs, such as ChatGPT and GPT-4, poses a significant challenge for applying knowledge editing techniques due to restricted access to their underlying parameters. Nonetheless, OpenAI's API provision for models including `gpt-3.5-turbo-1106` and `gpt-4-0613` facilitates fine-tuning within the LTE's Alignment Phase. Although our current work does not extend to these black-box models, addressing this limitation represents a critical avenue for future research, potentially unlocking new methods for model customization and improvement.

CHAPTER 6

ALIGNMENT TRAINING VIA DIRECT PREFERENCE OPTIMIZATION

6.1 Introduction

Direct preference optimization (DPO) [148] has emerged as a prominent alternative to reinforcement learning from human feedback (RLHF) [34, 9, 137] for aligning LLMs with human values. Unlike the traditional RLHF approach, DPO bypasses training a reward model and avoids using any reinforcement learning algorithms. Since the inception of DPO, numerous studies have sought to advance this method by refining its training objective [183]. For instance, IPO [6] introduces an alternative pairwise preference loss to mitigate overfitting to the preference dataset, while R-DPO [142] incorporates a regularization term to prevent the exploitation of latent length bias in the training data.

However, relatively little attention has been given to enhancing DPO through advancements in the quality of preference data used for training. In particular, the generation of winning and losing responses within preference data often occurs in an *isolated* manner, either through human annotation [9] or automated techniques such as RLAIF [10] and reject sampling [113, 139]. This isolation implies that winning and losing responses are produced without mutual visibility, resulting in a lack of strong correlation or relevance between them. Consequently, the model may struggle to identify nuanced yet significant distinctions that differentiate superior responses from inferior ones [57, 189], which can ultimately compromise optimization and alignment effectiveness.

In this chapter, we introduce an innovative framework, termed **BMC**, to Bridge and Model Correlations in pairwise data for direct preference optimization. During the Bridging Phase, we enhance correlations by increasing the consistency and informativeness of pairwise preference signals. By using the winning response as a reference, we synthesize a pseudo-winning response through *targeted modifications* of the losing response. This

pseudo-winning response offers two key advantages: (1) it preserves essential characteristics of the losing response, minimizing noise in preference signals (*consistency*); (2) it encapsulates all human-desired values from the winning response, enabling the model to better discern features that lead to superior performance (*informativeness*).

The nuanced differences between the pseudo-winning and losing responses are indeed what we expect the model to learn in the subsequent Modeling Phase. Nonetheless, we identify that DPO alone is insufficient to model these correlations and capture nuanced variations. From the perspective of the token-level Markov Decision Process (MDP) [147], DPO aggregates rewards uniformly across all tokens, assuming equal contribution to sequence quality and neglecting token-specific importance. To address this, we adjust the emphasis on rewards of different tokens between pseudo-winning and losing responses. Unlike previous methods [68, 18, 21, 27] that assign predefined values for fine-grained guidance, our adjustment is *dynamically* guided by the policy model's confidence, *i.e.*, the probability assigned to generated tokens during training. This ensures the model focuses on learning challenging distinctions while reinforcing known patterns, resulting in a more nuanced and robust policy.

We conduct extensive experiments across three downstream scenarios: question answering, mathematical reasoning, and instruction following, utilizing a total of 10 datasets. Our results demonstrate that our method consistently and significantly outperforms competitive offline optimization algorithms across various tasks. Furthermore, we use in-depth analyses to elucidate why our method outperforms DPO and show that our framework can be versatilely adapted to other DPO variants, confirming its potential for broad application.

6.2 Methodology

In this section, we present the proposed **BMC** approach, which bridges and models correlations in pairwise data for direct preference optimization. As depicted in Figure 6.1, our BMC framework is structured around two pivotal stages: (1) the Bridging Phase, where we enhance the correlations between pairwise data by increasing the consistency and informativeness of pairwise preference signals through *targeted modifications* (§6.2.1); and (2)

the Modeling Phase, where we *dynamically* model the correlations during the optimization process by leveraging the confidence of the policy model (§6.2.2), alleviating the insufficient token-level credit assignment of DPO.

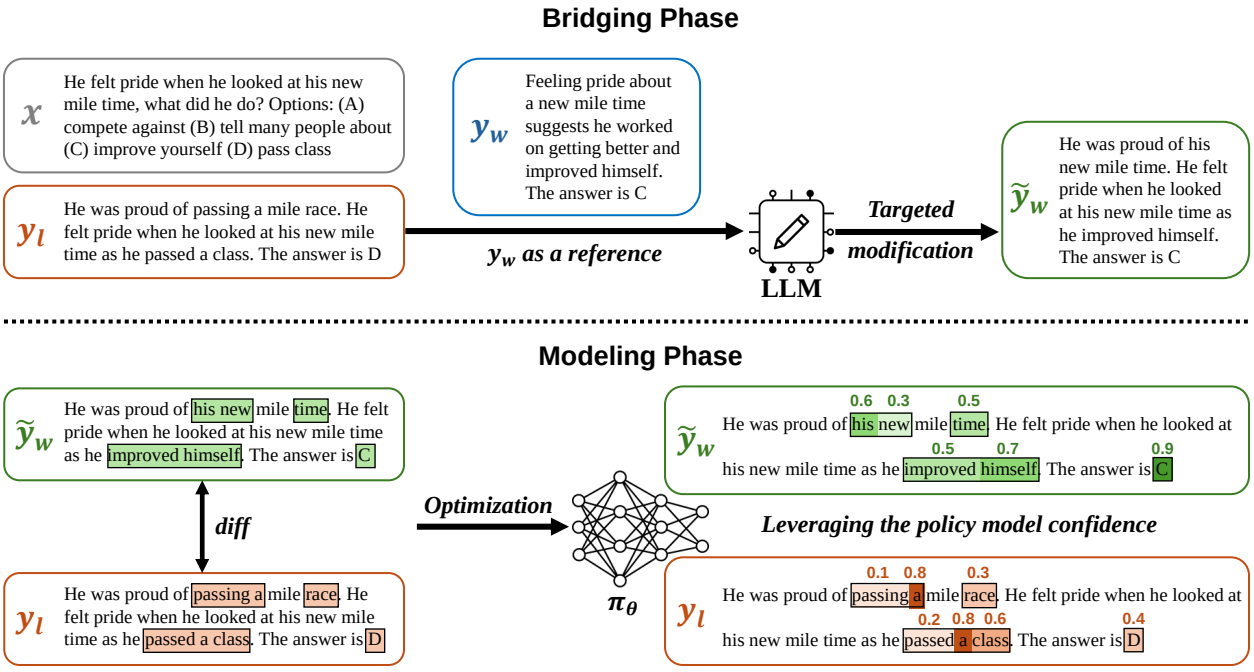


Figure 6.1: Overview of our proposed BMC framework. (1) In the Bridging Phase, we utilize an off-the-shelf LLM to make *targeted modifications* of losing response y_l on undesired tokens, with the winning response y_w serving as a reference. Therefore, the synthesized pseudo-winning response $\tilde{y}_w$ is highly correlated with y_l. (2) In the Modeling Phase, we model the correlations between $\tilde{y}_w$ and y_l by *dynamically* emphasizing the rewards of their varied tokens (*diff*$(\tilde{y}_w \mid y_l)$ and *diff*$(y_l \mid \tilde{y}_w)$), leveraging the policy model confidence (numbers indicated above tokens) during training.

6.2.1 Bridging Phase

In offline preference optimization, it is commonly assumed that we have access to a static pairwise preference dataset $\mathcal{D} = \{x^{(i)}, y_w^{(i)}, y_l^{(i)}\}_{i=1}^{N}$, where y_w and y_l denote the winning and losing response, give the input prompt x. However, since y_w and y_l are typically generated in isolation, the correlation between y_w and y_l can be inherently weak during pairwise preference optimization. In the context of DPO, the Bradley-Terry objective [14] computes gradients based on the relative likelihoods of y_w and y_l. When the correlation

between y_w and y_l is weak, the differences between these responses are often superficial (*e.g.*, stylistic or irrelevant variations) rather than substantive distinctions that reflect human-preferred behaviors. Consequently, the optimization process may inadvertently focus on minor discrepancies rather than meaningful distinctions. This results in gradients that are less informative for guiding the model towards robust preference alignment.

To address this challenge, we enhance the alignment efficacy by improving the consistency and informativeness of pairwise preference signals. As shown in the upper part of Figure 6.1, we utilize an off-the-shelf LLM to make targeted modification of y_l by referring to y_w:

$$\text{LLM}(I, x, y_w, y_l) \rightarrow \tilde{y}_w, \tag{6.1}$$

where $\tilde{y}_w$ is the generated pseudo-winning response, I is the instruction (see examples in Appendix D.1.2) that requires y_l to be modified only on dispreferred tokens, using y_w as a reference guidance. In this way, $\tilde{y}_w$ preserves essential characteristics of the losing response y_l while encapsulating all human-desired values in the winning response y_w. The token-level differences between $\tilde{y}_w$ and y_l highlight the core human expected and unexpected behaviors by decoupling from the inherent linguistic style and overall semantic distribution. Thus, $(\tilde{y}_w, y_l)$ refines the original training data (y_w, y_l) for more focused learning, shifting the optimization process to concentrate on the most critical differences in preference data. The benefits of the Bridging Phase are further analyzed in §6.4.2. Finally, we use the new dataset $\tilde{\mathcal{D}} = \{x^{(i)}, \tilde{y}_w^{(i)}, y_l^{(i)}\}_{i=1}^{N}$ for subsequent training.

An alternative approach that attempts to enhance the correlation between the winning and losing responses is to degenerate y_w to $\tilde{y}_l$ via targeted modification and utilize $(y_w, \tilde{y}_l)$ as the preference pair. Nevertheless, our ablation study in Table 6.3 reveals that LLMs encounter challenges with this inverse operation, leading to a notable decline in performance.

6.2.2 Modeling Phase

After the Bridging Phase, the token-level differences between $\tilde{y}_w$ and y_l can be obtained through dynamic programming algorithms like Levenshtein Distance [204]. As depicted in the lower part of Figure 6.1, these nuanced variations guide LLMs to prioritize the

reinforcement of optimal actions while discouraging suboptimal ones within a single response. However, our findings below indicate that DPO alone is insufficient for capturing the nuanced variations, highlighting the necessity for supplementary techniques to comprehensively model these correlations.

Alternative Interpretation of DPO. DPO [148] introduced a novel framework for optimizing the equivalent KL-constrained reward function as in RLHF, without the need to learn an explicit reward model. Instead, the problem is cast as a maximum likelihood estimation for the policy model π_θ on the preference dataset $\mathcal{D}$, resulting in the following training objective:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l)\sim\mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} \right) \right], \tag{6.2}$$

where π_{ref} is the reference model, typically the supervised fine-tuned (SFT) model, and β is a regularisation term corresponding to the strength of KL-regularization in RLHF.

As shown in Eq. (6.2), DPO was originally conceptualized as a bandit problem, where the whole response of the model is treated as a single arm to receive a reward. More recently, [147] extended the theoretical foundation of DPO, showing that it can also be derived in the context of token-level MDP. The corresponding training objective at the token level is:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(\tau_w, \tau_l)\sim\mathcal{D}} \left[\log \sigma \left(\beta \sum_{t=0}^{N-1} \log \frac{\pi_\theta(a_w^t \mid s_w^t)}{\pi_{\text{ref}}(a_w^t \mid s_w^t)} - \beta \sum_{t=0}^{M-1} \log \frac{\pi_\theta(a_l^t \mid s_l^t)}{\pi_{\text{ref}}(a_l^t \mid s_l^t)} \right) \right], \tag{6.3}$$

where τ_w and τ_l denote the win trajectory and the lose trajectory, respectively. a indicates the action (current generated token), and s signifies the state (all tokens generated so far).

Our Solution. It can be inferred from Eq. (6.3) that DPO, redefined as a token-level MDP, assigns rewards to each token generation by $\beta \log \frac{\pi_\theta(a^t|s^t)}{\pi_{\text{ref}}(a^t|s^t)}$, and simply add up the rewards of all tokens as the accumulated reward of the trajectory. This *uniform aggregation* assumes that each token contributes equally to the overall quality of the sequence, without

considering the varying importance of each token (timestep). Therefore, nuanced differences between $\tilde{y}_w$ and y_l that significantly influence the overall meaning or quality of the response might not be adequately emphasized (refer to Figure 6.6), leading to suboptimal performance. To this end, we propose to emphasize the rewards of critical tokens, *i.e.*, nuanced differences between $\tilde{y}_w$ and y_l. The magnitude of the emphasis is determined *dynamically* by the policy model's confidence, which refers to the probability assigned to the generated token during training. Below, we detail our design choices for the pseudo-winning response and losing response, respectively.

- For varied tokens in the pseudo-winning response $\tilde{y}_w$, we adapt the reward factor based on the learning process of the policy model. Lower policy confidence indicates underdeveloped learning of the target behavior, signaling the need for additional focus to help the model better capture these nuances. Consequently, we adjust the reward factor to be inversely proportional to the policy model's confidence, as formalized in Eq. (6.5).

- For varied tokens in the losing response y_l, we carefully adjust the reward factor by reinforcing already learned patterns of the policy model. Intuitively, tokens in y_l with higher confidence from the policy model may reflect inaccurate preference learning and therefore warrant stronger penalization. However, our analysis reveals a distinct pattern of the policy model when processing y_l compared to $\tilde{y}_w$. Specifically, when grouping varied tokens in y_l into coarser-grained spans, the model's confidence is significantly influenced by the token's position within these spans, as illustrated in Figure 6.2. We observe that the probabilities assigned to the initial token of incorrect spans in y_l are typically low, whereas the probabilities for subsequent tokens within the same span are notably higher. Prior studies have identified token probability as a critical signal for detecting anomalous behaviors [192, 53] and assessing generation quality [201, 56]. Consistent with these findings, our results indicate that during training, the policy model can effectively recognize the onset of undesired spans by assigning low probabilities to initial tokens. Nonetheless, due to the autoregressive dependencies, subsequent tokens within these spans receive higher probabilities, reflecting the contextual coherence established by preceding tokens, even when the span as a whole is incorrect. Thus, while it is crucial to penalize initial tokens, applying equally strong penalties to subsequent tokens might be suboptimal, as they often maintain local coherence within the flawed

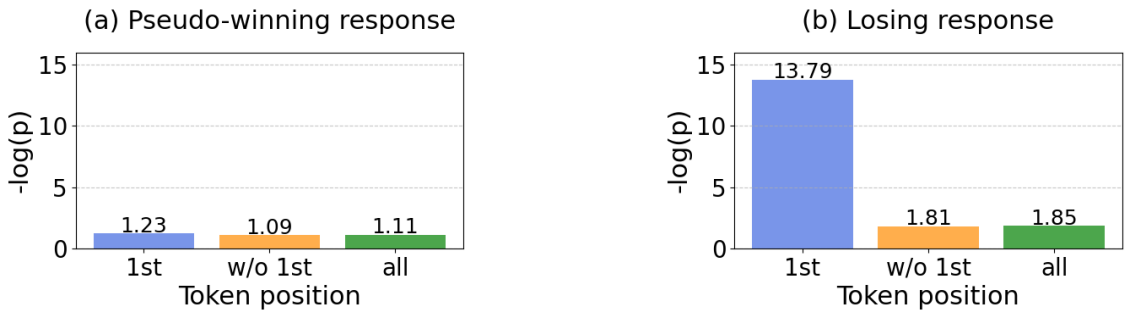


Figure 6.2: We aggregate varied tokens in $\tilde{y}_w$ or y_l into more coarser-grained spans. During the DPO training on $\tilde{\mathcal{D}}$, we compute the averaged $-\log(p)$ of tokens in different positions of spans.

span. Therefore, we adjust the reward factor to also be inversely proportional to the policy model's confidence in Eq. (6.6).

In a nutshell, our approach dynamically modulates the emphasis placed on critical tokens based on the policy model's confidence. This adaptive reward mechanism ensures that the model focuses on learning challenging distinctions while reinforcing already learned patterns, ultimately fostering a more nuanced and robust policy (see our analysis in §6.4.2). The formalization of our approach is encapsulated in Eq. (6.4), where $\lambda_{\tilde{y}_w^t}$ and $\lambda_{y_l^t}$ adjust dynamically based on the policy's confidence, ensuring a tailored emphasis on critical tokens to improve the overall model performance.

$$\mathcal{L}_{\text{DPO-BMC}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,\tilde{y}_w,y_l)\sim\tilde{\mathcal{D}}}\left[\log\sigma\left(\beta\sum_{\tilde{y}_w^t\in\tilde{y}_w}\lambda_{\tilde{y}_w^t}\log\frac{\pi_\theta(\tilde{y}_w^t \mid \tilde{y}_w^{<t}, x)}{\pi_{\text{ref}}(\tilde{y}_w^t \mid \tilde{y}_w^{<t}, x)}\right.\right.$$
$$\left.\left.-\beta\sum_{y_l^t\in y_l}\lambda_{y_l^t}\log\frac{\pi_\theta(y_l^t \mid y_l^{<t}, x)}{\pi_{\text{ref}}(y_l^t \mid y_l^{<t}, x)}\right)\right], \tag{6.4}$$

where

$$\lambda_{\tilde{y}_w^t} = \begin{cases} 1+\min\left(sg\left(\frac{1}{\pi_\theta(\tilde{y}_w^t|\tilde{y}_w^{<t},x)}\right),\delta\right), & \text{if } \tilde{y}_w^t \in \textit{diff}(\tilde{y}_w \mid y_l) \\ 1, & \text{otherwise} \end{cases} \tag{6.5}$$

$$\lambda_{y_l^t} = \begin{cases} 1+\min\left(sg\left(\frac{1}{\pi_\theta(y_l^t|y_l^{<t},x)}\right),\delta\right), & \text{if } y_l^t \in \textit{diff}(y_l \mid \tilde{y}_w) \\ 1, & \text{otherwise} \end{cases} \tag{6.6}$$

The sg denotes the stop-gradient operator, the δ is an upper limit threshold that controls the emphasis on the rewards of the critical tokens, preventing overly aggressive updates. The *diff* $(\tilde{y}_w \mid y_l)$ and *diff* $(y_l \mid \tilde{y}_w)$ signify using the Levenshtein Distance algorithm to find the varied tokens in $\tilde{y}_w$ and y_l, respectively.

Gradient Analysis of DPO-BMC. For a mechanistic understanding of our method, we examine the gradients of the loss function $\mathcal{L}_{\text{DPO}}$ in Eq. (6.2) and $\mathcal{L}_{\text{DPO-BMC}}$ in Eq. (6.4). Their gradients with respect to the parameters θ can be written as:

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\beta \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\sigma\left(\Delta_1\right) \left[\underbrace{\nabla_\theta \log \pi_\theta(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi_\theta(y_l \mid x)}_{\text{decrease likelihood of } y_l} \right] \right],$$

where $\Delta_1 = \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} - \beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}$.

$$\nabla_\theta \mathcal{L}_{\text{DPO-BMC}}(\pi_\theta; \pi_{\text{ref}}) = -\beta \mathbb{E}_{(x,\tilde{y}_w,y_l)\sim\tilde{\mathcal{D}}} \left[\sigma\left(\Delta_2\right) \left(\underbrace{\nabla_\theta \log \pi_\theta(\tilde{y}_w \mid x)}_{\text{increase likelihood of } \tilde{y}_w} - \underbrace{\nabla_\theta \log \pi_\theta(y_l \mid x)}_{\text{decrease likelihood of } y_l} \right.\right.$$

$$\left.\left. + \underbrace{\sum_{\tilde{y}_w^t \in \textit{diff}(\tilde{y}_w|y_l)} (\lambda_{\tilde{y}_w^t} - 1) \nabla_\theta \log \pi_\theta(\tilde{y}_w^t \mid \tilde{y}_w^{<t}, x)}_{\text{increase likelihood of desired tokens of } \tilde{y}_w} - \underbrace{\sum_{y_l^t \in \textit{diff}(y_l|\tilde{y}_w)} (\lambda_{y_l^t} - 1) \nabla_\theta \log \pi_\theta(y_w^t \mid y_l^{<t}, x)}_{\text{decrease likelihood of undesired tokens of } y_l} \right) \right],$$

where $\Delta_2 = \beta \sum_{y_l^t \in y_l} \lambda_{y_l^t} \log \frac{\pi_\theta(y_l^t|y_l^{<t},x)}{\pi_{\text{ref}}(y_l^t|y_l^{<t},x)} - \beta \sum_{\tilde{y}_w^t \in \tilde{y}_w} \lambda_{\tilde{y}_w^t} \log \frac{\pi_\theta(\tilde{y}_w^t|\tilde{y}_w^{<t},x)}{\pi_{\text{ref}}(\tilde{y}_w^t|\tilde{y}_w^{<t},x)}$.

In contrast to vanilla DPO, which emphasizes sequence-level optimization exclusively, **our proposed method integrates both sequence-level and token-level perspectives**. (1) At the sequence level, we promote preferred completions while penalizing those that are disfavored. (2) At the token level, we further refine the rewards of critical desired and undesired tokens of $\tilde{y}_w$ and y_l, respectively. This dual consideration ensures that both the overall sequence structure and the critical token choices are optimized for the desired outcome.

6.3 Experiments

6.3.1 Experimental Setup

We conduct a comprehensive evaluation across three downstream scenarios, including question answering (QA), mathematical reasoning, and instruction following (IF). The detailed data statistics as well as the evaluation metrics are listed in Table D.1 of Appendix D.1.1.

Models and Training Settings. For the QA and mathematical reasoning setup, we utilize Llama2-7B-base [171] in our experiments. Dealing with these tasks necessitates LLMs to possess domain-specific knowledge and engage in systematic, step-by-step reasoning to reach the ultimate answer. Therefore, following prior works [27, 28], we fine-tune Llama2-7B-base on the training set of ECQA [1] and QASC [88] for QA, and fine-tune Llama2-7B-base on MetaMathQA [199] for mathematical reasoning. We denote the fine-tuned LLM as SFT and use it as the backbone for preference optimization. In line with prior research [27, 28], we construct preference pairs (y_w, y_l) based on the training data, by using the ground truth as y_w and the SFT model's inference output as y_l. For the instruction following setup, we utilize Llama3-8B-base [51] and Mistral-7B-Base [83] in our experiments. Following the training pipeline of Zephyr [173] and SimPO [121], we train a base model on the UltraChat-200k dataset [47] to obtain an SFT model. Then, we use the SFT model as the starting point and perform preference optimization on the UltraFeedback dataset [43], where y_w and y_l are collected from LLMs of varying quality.

During our Bridging Phase, we utilize `gpt-4-0125-preview` for targeted modification to obtain $\tilde{y}_w$, based on the prompt template in Appendix D.1.2. We also demonstrate in Table 6.4 that **a less powerful open-source** LLM, such as `Llama3-70B-Instruct`, can acquire comparable results. During our Modeling Phase, we list the implementation details in Appendix D.1.3 for reproducibility. A comprehensive cost analysis in §6.4.1 confirms that **the computational overhead introduced by our BMC pipeline is minimal**.

Evaluation Benchmarks. In question answering, we adopt the test splits of ECQA [1], QASC [88], OpenbookQA [123], and StrategyQA [60] for evaluation. In mathematical

reasoning, we conduct the evaluation on four challenge datasets including GSM8k [38], MATH [73], MAWPS [90], and TabMWP [117]. In instruction following, We assess our models using two of the most popular open-ended instruction-following benchmarks: AlpacaEval 2 [107] and Arena-Hard v0.1 [104]. Both benchmarks evaluate the models' versatile conversational abilities across a diverse set of queries. For each query, the evaluated model's response and the reference model's response are compared head-to-head using an auto-evaluator. We use the officially recommended configurations[1] during the evaluation.

Baselines. We compare our approach with various powerful *offline* preference optimization methods, including FIGA [68], DPO [148], and DPO variants (IPO [6], ORPO [75], R-DPO [142], and SimPO [121]). The training objectives of these methods are listed in Table D.2. Besides, we include two additional baselines: (1) **DPO (CW)**: enhancing pairwise data correlation by prompting the SFT model to **C**ontinue **W**riting a prefix of the winning response to generate the losing one; (2) **DPO (EW)**: leveraging an off-the-shelf LLM for **E**xternal **W**eighting of token-level reward [96], where LLM scores each token in the winning and losing responses based on how much it improves or decreases the overall quality.

6.3.2 Experimental Results

Our Method Consistently and Significantly Outperforms Baselines. As presented in Table 6.1, our model DPO-BMC consistently achieves state-of-the-art results across all evaluated QA and math benchmarks. Specifically, DPO-BMC outperforms DPO by 3.8 absolute points on QA tasks and by 1.3 points on math tasks. On instruction-following tasks (Table 6.2), DPO-BMC secures the highest length-controlled win rate, surpassing DPO by over 5 points across various settings, with even greater gains for larger base models (Appendix D.3). The length-controlled win rate [52] serves as a robust metric that mitigates the effects of length bias, thereby providing a more reliable evaluation of LLM-based auto-annotation. Notably, **DPO-BMC generates responses that are significantly**

[1]AlpacaEval: `https://github.com/tatsu-lab/alpaca_eval`. Arena-Hard v0.1: `https://github.com/lm-sys/arena-hard-auto`.

Method	Question-Answering Tasks					Mathematical Reasoning Tasks				
	ECQA	QASC	OBQA	StrategyQA	Avg.	GSM8k	MATH	MAWPS	TabMWP	Avg.
SFT	72.8	54.5	51.8	56.9	59.0	55.8	11.6	80.3	42.8	47.6
FIGA	70.3	52.5	51.7	48.6	55.8	54.1	9.8	75.5	39.0	44.6
IPO	71.5	58.9	53.6	58.4	60.6	57.2	12.1	82.2	42.5	48.5
OPRO	69.8	55.1	51.4	57.2	58.4	56.0	12.4	80.8	41.3	47.6
R-DPO	73.5	59.5	55.4	58.8	61.8	56.9	12.0	81.9	42.2	48.2
SimPO	71.9	56.7	52.2	55.4	59.1	57.5	12.7	81.8	43.5	48.9
DPO	73.1	58.8	55.6	57.8	61.3	56.3	12.3	81.2	43.4	48.3
DPO (CW)	72.5	58.6	55.2	57.3	60.9	55.9	11.8	80.7	42.8	47.8
DPO (EW)	72.9	59.4	55.8	57.9	61.5	56.5	12.0	80.9	43.4	48.2
DPO-BMC	**75.9**	**63.0**	**60.4**	**61.0**	**65.1**	**58.4**	**13.0**	**83.1**	**43.8**	**49.6**
DPO-BC	75.7	62.0	56.0	60.1	63.4	57.6	12.7	82.8	43.4	49.1
DPO-MC	74.8	60.0	56.4	58.8	62.5	57.2	12.5	82.4	43.0	48.8

Table 6.1: Experimental results (based on Llama2-7B-base) on question answering tasks and mathematical reasoning tasks. "Avg." is the average accuracy of all sub-tasks. In each column, the highest score is **bolded** and the second-highest is underlined.

Method	Llama3-8B-Base					Mistral-7B-Base				
	AlpacaEval 2			Arena-Hard		AlpacaEval 2			Arena-Hard	
	LC (%)	WR (%)	Avg. len	WR (%)	Avg. len	LC (%)	WR (%)	Avg. len	WR (%)	Avg. len
SFT	7.5	4.7	956	2.6	414	8.1	5.9	998	2.2	454
FIGA	8.4	4.2	1,199	5.1	416	7.0	4.9	1,378	2.5	461
IPO	13.4	9.8	1,430	14.0	477	12.5	10.8	1,588	8.5	522
ORPO	12.5	11.4	1,793	11.7	573	14.5	11.5	1,630	9.4	566
R-DPO	17.1	14.4	1,801	17.6	582	16.0	12.3	1,521	10.4	529
SimPO	21.3	**18.9**	1,718	**26.6**	562	16.8	14.4	1,906	**18.4**	615
DPO	16.0	14.8	1,713	17.6	559	15.1	13.3	1,657	13.6	540
DPO (CW)	15.2	14.0	1,756	17.1	570	14.5	12.9	1,647	13.0	532
DPO (EW)	17.2	15.6	1,702	18.2	566	15.3	13.4	1,668	13.9	549
DPO-BMC	**22.4**	16.8	1,285	18.1	406	**20.8**	**16.6**	1,317	17.6	488
DPO-BC	20.6	14.4	1,269	16.8	422	18.6	13.8	1,489	15.9	502
DPO-MC	17.7	15.2	1,890	17.9	579	16.4	14.3	1,712	15.4	551

Table 6.2: Experimental results on instruction-following tasks. "LC" is the length-controlled win rate, and "WR" is the raw win rate. "Avg. len" denotes the average number of tokens in the responses.

more concise than other baselines. As highlighted in Table 6.2, the average response length of DPO-BMC and DPO-BC is approximately 75% of that produced by DPO and DPO-MC. This attribute of length normalization is credited to the correlated preference data we constructed, which directs optimization towards critical desired behaviors rather than verbosity.

6.3.3 Ablation Study

Both Key Designs in BMC are Crucial. In Table 6.1 and Table 6.2, we additionally present results from ablating each key design element of DPO-BMC:

- **DPO-BC**: Training using DPO's original objective on our constructed preference data.
- **DPO-MC**: Training using our proposed objective in Eq. (6.4) on the original preference data.

Our examination reveals several key findings: (1) DPO (CW), the "Continue Writing" approach, slightly underperforms standard DPO, as it introduces superficial correlations that fail to capture the nuanced, task-specific alignments essential for effective optimization. In contrast, our Bridging Phase explicitly enhances *informative correlations*—elucidate fine-grained distinctions between desired and undesired behaviors through token-level variations. This targeted focus significantly improves model performance; (2) Even when leveraging identical training preference data, our designed optimization objective consistently outperforms both DPO and DPO (EW), highlighting its superior ability to model fine-grained correlations based on the dynamic of the policy model's confidence; and (3) Combining our constructed data with our designed objective yields the best results, affirming the inseparability of the Bridging Phase and the Modeling Phase.

Influence of Data Synthesis Method. Table 6.3 shows the effects of various data synthesis strategies during the Bridging Phase. When generating $\tilde{y}_w$ without referring to y_w, LLMs potentially make erroneous modifications that misalign with the intended target, leading to a performance drop. An alternative approach that attempts to enhance the correlation between winning and losing responses is to degenerate y_w to $\tilde{y}_l$ and utilize $(y_w, \tilde{y}_l)$ as the preference pair. However, this approach also falls short, likely because LLMs are primarily trained to generate high-quality data, making it challenging for them to generate low-quality outputs that mimic the nuanced errors of losing responses. Semantic similarity analysis using the `all-mpnet-base-v2` embedding model[2] supports this, showing a high score of 0.88 for $(y_w, \tilde{y_w})$ but only 0.73 for $(y_l, \tilde{y}_l)$.

[2] `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`

Data Synthesis	Training Data	QA	Math	IF
$y_l \xrightarrow{y_w} \tilde{y}_w$ (ours)	$(\tilde{y}_w, y_l)$	**65.1**	**49.6**	**22.4**
$y_l \longrightarrow \tilde{y}_w$	$(\tilde{y}_w, y_l)$	64.3	49.2	19.8
$y_w \xrightarrow{y_l} \tilde{y}_l$	$(y_w, \tilde{y}_l)$	64.6	48.7	18.9
$y_w \longrightarrow \tilde{y}_l$	$(y_w, \tilde{y}_l)$	63.9	48.6	17.6

Table 6.3: Ablation study on diverse data synthesis methods in the Bridging Phase. The average accuracy is presented for QA and Math. LC on AlpacaEval 2 is reported for instruction following (IF), based on Llama3-8B.

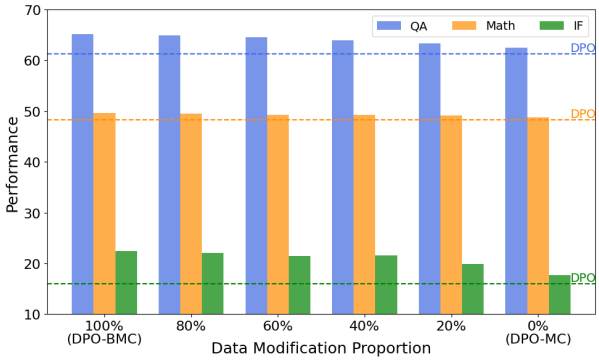


Figure 6.3: Ablation study on data modification proportion in the Bridging Phase.

Influence of Data Modification Proportion. Figure 6.3 illustrates the impact of data modification proportions during the Bridging Phase on performance. Increasing modifications from 0% to 20% yields the most substantial gains, highlighting the effectiveness of enhancing pairwise preference correlations. Performance plateaus beyond 80% modifications, indicating that extensive changes are beneficial but not essential, offering flexibility under computational or data constraints. These results demonstrate the scalability and adaptability of our framework for diverse applications.

Influence of LLMs for Targeted Modification. Table 6.4 explores the influence of diverse LLMs for targeted modification. Notably, substituting the `gpt-4-0125-preview` model with a less powerful yet open-source alternative, such as `Llama3-70B-Instruct`, **yields comparable performance while significantly surpassing vanilla DPO**. This finding underscores the adaptability of our method to varying levels of model sophistication, thereby reducing dependence on commercial LLMs without significant impact on final model performance.

Influence of δ. We conduct an ablation study to examine the influence of the threshold δ in the DPO-BMC objective on model performance, as shown in Figure 6.4. Setting $\delta = 1.0$ reduces our method to one that assigns fixed token-level rewards, leading to suboptimal accuracy. As δ increases, the model performance improves, with the optimal setting observed around $\delta = 3.0$. However, further increasing δ results may degrade model performance due to excessively aggressive gradient updates on certain tokens. Notably, across

Method	LLM for Targeted Modification	QA	Math	IF
SFT	–	56.9	47.6	7.5
DPO	–	61.3	48.3	16.0
DPO-BMC	Llama3-70B-Instruct	64.6	49.4	21.8
DPO-BMC	gpt-4-0125-preview	**65.1**	**49.6**	**22.4**

Table 6.4: Influence of diverse LLMs for targeted modification in the Bridging Phase. The average accuracy is presented for QA and Math. LC on AlpacaEval 2 is reported for instruction following (IF), based on Llama3-8B.

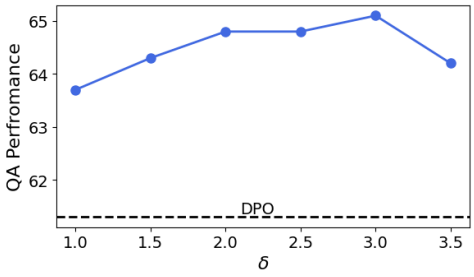


Figure 6.4: Ablation study on δ in the Modeling Phase. The average accuracy is presented as the QA performance.

all tested values of δ, our method consistently outperforms the DPO baseline, indicating its robustness and effectiveness in stabilizing the learning process.

6.4 Analysis

In this section, we begin with the cost analysis of our proposed BMC framework (§6.4.1). Furthermore, we conduct in-depth quantitative analyses to elucidate why our method outperforms DPO (§6.4.2 and §6.4.3). Finally, we demonstrate the versatility of our framework by adapting it to other DPO variants (§6.4.4).

6.4.1 Cost Analysis of Bridging and Modeling Phase

Cost of Bridging Phase

The Bridging Phase, responsible for synthesizing pseudo-winning responses, operates exclusively **offline**, meaning it incurs no runtime cost during model training. The data synthesis process is designed to be efficient, as it does not require iterative computations or model updates.

For context, we estimated the budget for data synthesis using the `gpt-4-0125-preview` API, based on the API's pricing of $0.01 per 1K input tokens and $0.03 per 1K output tokens. Table 6.5 lists the breakdown of the estimated costs for our three evaluated tasks, which demonstrates that this is a manageable expenditure.

Task	# of Samples	Avg. Input Token Length	Avg. Output Token Length	Cost ($)
QA	15,732	206	25	44.21
Math	40,000	429	47	228.00
IF	61,135	728	235	876.06

Table 6.5: Estimated budget for data synthesis using the `gpt-4-0125-preview` API.

Can an Open-Source LLM be Utilized as an Alternative? In Table 6.4, we explore the impact of LLMs on targeted modifications during the Bridging Phase. Our findings indicate that substituting the `gpt-4-0125-preview` model with a less powerful yet open-source alternative, such as `Llama3-70B-Instruct`, **yields comparable performance while significantly surpassing vanilla DPO**. The `Llama3-70B-Instruct` model can be deployed on only 2 NVIDIA-3090 GPUs, with the option to further reduce hardware requirements through low-bit quantization[3]. This provides an economical alternative for our Bridging Phase without compromising performance. Numerous studies have highlighted the superior text modification capabilities of LLMs. For example, LLMs have been effectively employed in synthesizing high-quality data [180]. Additionally, [82] show that LLMs can transform initial outputs from upstream models into more helpful and benign responses, thereby aligning generated content with human intentions. In conclusion, our framework demonstrates robustness in leveraging diverse LLMs for targeted modifications, confirming its adaptability and effectiveness.

Cost of Modeling Phase

Our Modeling Phase adds minimal computational overhead compared to vanilla DPO. Specifically:

- **Token Difference Identification**: Using a dynamic programming algorithm (edit distance) to identify differing tokens between the pseudo-winning and losing responses. This is a lightweight operation and introduces negligible runtime cost.
- **Reward Weighting Calculation**: We calculate a weighting factor based on the policy model's probability of the identified tokens, which is already computed in the standard

[3] `https://github.com/ollama/ollama`

DPO setup. Because we halt gradient backpropagation for the weighting factor, this operation does not introduce additional computational costs.

Table 6.6 demonstrates the comparison of the training times between DPO and DPO-BMC on 4×A800 GPUs, illustrating that **DPO-BMC increases training time by less than 1% across all evaluated tasks**.

Task	Base Model	Runtime of DPO (s)	Runtime of DPO-BMC (s)	Increase (%)
QA	Llama2-7B	2,831	2,850	**0.67%**
Math	Llama2-7B	9,586	9,641	**0.57%**
IF	Llama3-8B	16,179	16,318	**0.86%**

Table 6.6: Runtime usage for DPO and DPO-BMC during the Modeling Phase.

Overall, these results validate that the computational overhead introduced by BMC is minimal, and the approach is highly efficient in terms of runtime, making it practical for real-world applications without significantly increasing resource requirements.

6.4.2 Quantitative Analysis of Bridging and Modeling Phase

To rigorously assess the effectiveness of the two pivotal phases in our framework, we segment the 60k training data of UltraFeedback into six equal-sized splits, ordered by increasing edit distance between winning and losing responses. For each split, we also construct its corresponding $(\tilde{y}_w, y_l)$ pair data through our Bridging Phase. We then train four models——(a) DPO, (b) DPO-MC, (c) DPO-BC, and (d) DPO-BMC—on each split based on Llama3-8B, with identical hyperparameters to ensure comparability. As shown in Figure 6.5, the Bridging Phase successfully decreases the edit distance between pairwise data through targeted modification, shifting the optimization process to concentrate on the most critical differences in preference data. This phase consistently enhances performance across all splits by refining training data for more focused learning. Another notable observation is the average gradient norm during DPO training increases as the edit distance between pairwise data enlarges, reflecting the sensitivity of DPO's training process to individual data points and potential gradient variance. Our proposed Modeling Phase mitigates the variance by dynamically adjusting the training process based on the policy

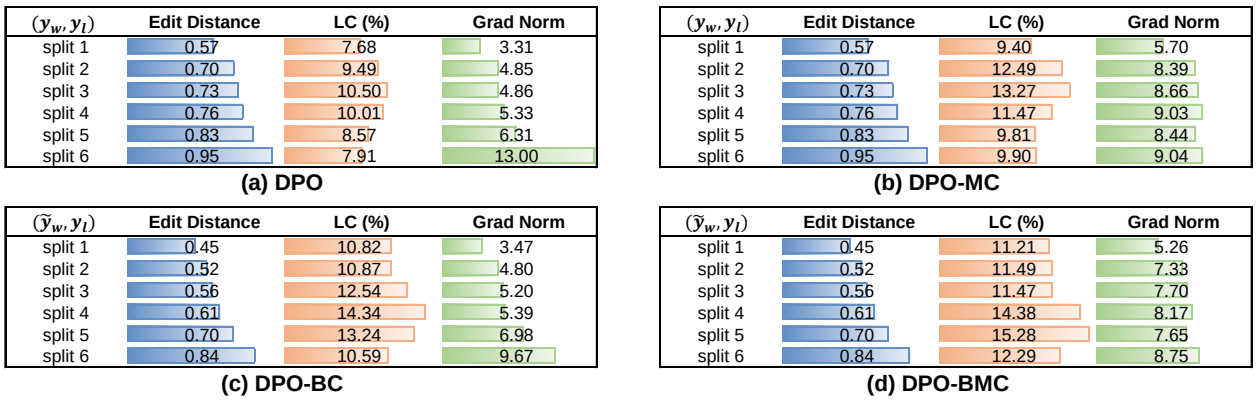


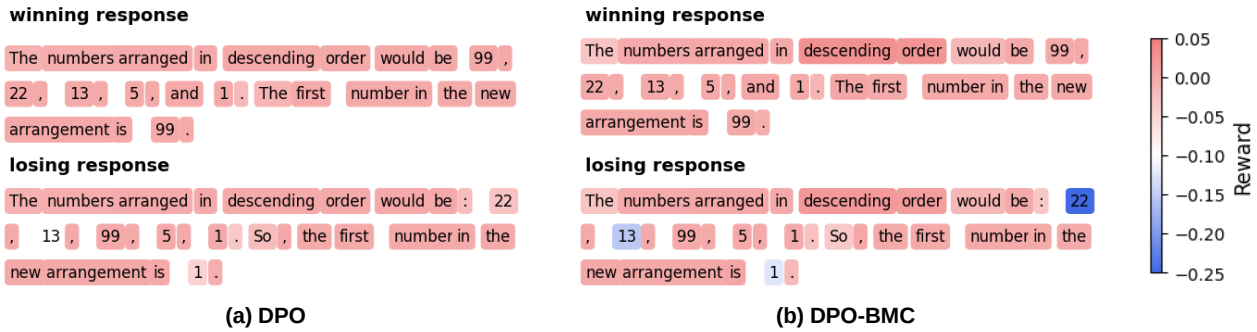
Figure 6.5: We segment the 60k training data of UltraFeedback into six equal-sized splits based on increasing edit distance between winning and losing responses. For each split, we report LC on AlpacaEval 2 and the average gradient norm during training.

model's confidence. This adaptive mechanism prioritizes challenging distinctions while reinforcing learned patterns, promoting a balanced optimization landscape with diverse training data (See Appendix D.2 for further analysis).

6.4.3 Quantitative Analysis of Credit Assignment

We compare the token-level and sequence-level credits assigned by DPO and DPO-BMC, assessing how well their final learned rewards align with preference labels on a held-out set of UltraFeedback.

Analysis on Token-level Reward. Figure 6.6 depicts the token-level reward assignment for DPO and DPO-BMC on a response pair consisting of a winning response y_w and a losing response y_l. The reward of each token is computed as $r_\theta(x, y^t) = \beta \log \frac{\pi_\theta(y^t|y^{<t},x)}{\pi_{\text{ref}}(y^t|y^{<t},x)}$. From the figure, we observe that: (1) DPO assigns nearly uniform rewards across tokens, failing to differentiate the importance of tokens to the overall response quality; and (2) although DPO can identify and assign lower rewards to several erroneous tokens in the losing response (*e.g.*, "13"), it struggles to capture subtle distinctions between the winning and losing responses. In contrast, DPO-BMC assigns higher rewards to critical tokens (*e.g.*, "descending order") and effectively penalizes incorrect tokens in the losing response. These results demonstrate DPO's limitations in providing precise token-level preferences



(a) DPO (b) DPO-BMC

Figure 6.6: Visualization of token-level rewards assigned by DPO and our method. The preference pair is sampled from the held-out set of UltraFeedback, whose input prompt is "*Arrange the numbers 5, 13, 99, 1, and 22 in descending order. What is the first number in the new arrangement?*"

on sentence quality, and our method can effectively alleviate this issue.

Analysis on Sequence-level Reward. For a rigorous comparison, we calculate the sequence-level DPO reward expression by $r_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$. The reward margin is determined by $r_\theta(x, y_w) - r_\theta(x, y_l)$. Reward accuracy is defined as the percentage of preference pairs where the winning response achieves a higher reward than the losing response, *i.e.*, $r_\theta(x, y_w) > r_\theta(x, y_l)$. Our findings show that DPO-BMC outperforms DPO in terms of average reward margin (**0.74 vs. 0.54**) and reward accuracy (**73.60 vs. 72.19**). This enhancement validates our method's superior ability to discern subtle differences between preference pairs, enabling more effective generalization.

6.4.4 Versatility of Our Framework

Our BMC framework demonstrates versatility and can be seamlessly integrated with various DPO variants. As shown in Table 6.7, the xPO-BMC methods consistently outperform their corresponding xPO baselines across a diverse set of tasks, including QA, Math, and Instruction Following (IF). For instance, IPO-BMC achieves a significant improvement in QA accuracy (64.1 vs. 60.6) and IF score (15.7 vs. 13.4) compared to IPO. Similarly, ORPO-BMC, R-DPO-BMC, SimPO-BMC, and DPO-BMC exhibit higher performance in QA and Math, alongside notable gains in IF, such as R-DPO-BMC improving the IF score from 17.1 to 20.0 over R-DPO. These results highlight the robustness of our framework in en-

hancing task-specific performance across various settings, reaffirming its potential as a generalizable enhancement to existing DPO methodologies.

Method	QA	Math	IF
SFT	56.9	47.6	7.5
IPO	60.6	48.3	13.4
IPO-BMC	**64.1**	**48.6**	**15.7**
ORPO	58.4	47.6	12.5
ORPO-BMC	**62.3**	**48.4**	**15.7**
R-DPO	61.8	48.2	17.1
R-DPO-BMC	**65.3**	**49.5**	**20.0**
SimPO	59.1	48.9	21.3
SimPO-BMC	**61.6**	**49.0**	**21.9**
DPO	61.3	48.3	16.0
DPO-BMC	**65.1**	**49.6**	**22.4**

Table 6.7: Versatility of our framework across various xPOs..

6.5 Conclusion

In this chapter, we propose BMC, an effective framework for bridging and modeling correlations in pairwise data for direct preference optimization. BMC equips LLMs with better human value alignment through a two-phase process: a Bridging Phase that enhances correlations between pairwise data by explicitly manifesting fine-grained preference signals via targeted modifications, and a Modeling Phase that learns token-level correlations by dynamically leveraging the the policy model's confidence during training. Our framework exhibits superior performance in question-answering, mathematical reasoning, and instruction-following tasks, consistently surpassing the baseline DPO by a significant margin. Extensive analysis highlights that the key designs in BMC are crucial and validates the effectiveness and versatility of BMC.

CHAPTER 7

ALIGNMENT EVALUATION FROM THE PERSPECTIVE OF CONSTRAINTS FOLLOWING

7.1 Introduction

LLMs [16, 136] pre-trained on web-scale corpora have showcased proficiency in generating fluent and realistic text. Yet, human instructions in real-life cases require the model to generate text that not only possesses a high degree of naturalness but adheres to specific constraints [197]. For instance, the model may be required to recommend ten books that are specifically written in Chinese (Figure 7.1), or it might be expected to generate responses that have a certain tone.

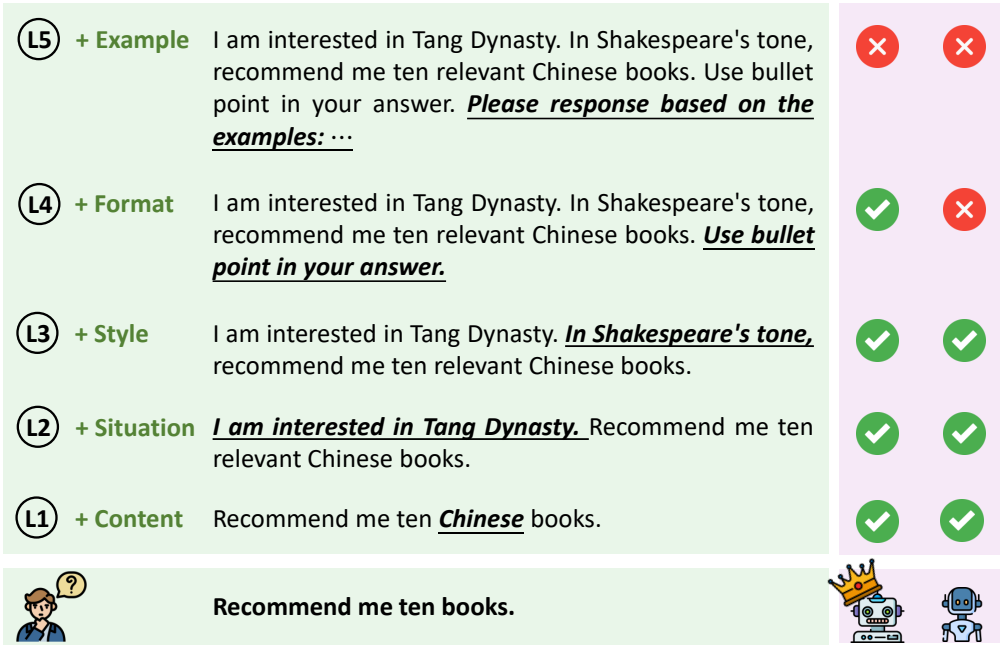


Figure 7.1: `FollowBench` covers five *fine-grained* constraint categories and is constructed based on the *Multi-level* mechanism, which increasingly adds a single constraint to straightforward instructions. On the right, the model that can follow instructions with more constraints is deemed to possess better instruction-following ability.

The dominant paradigm for assessing if a model can follow instructions involves using human annotators or strongly aligned LLMs to judge its response quality, in terms of helpfulness, relevance, accuracy, depth, creativity, and level of detail [180, 107, 211, 194]. However, prior work still has two limitations. Firstly, they ignore the **fine-grained constraints** inside instructions, which are essential and objective standards for evaluating the instruction-following capability. While several benchmarks have rigorously explored individual constraint types, including semantic restrictions [23] and complex formatting [163], there exists a lack of comprehensive analysis across the diverse spectrum of constraint categories. Secondly, few benchmarks consider the varying difficulty of instructions, which is controlled by the number of imposed constraints. This makes it challenging to precisely assess the degree to which LLMs can follow instructions. Towards this end, our research question is: *how can we systemically and precisely evaluate the instruction-following capability of LLMs?*

In this chapter, we construct `FollowBench`, a **Multi-level Fine-grained Constraints Following Benchmark**. `FollowBench` comprehensively includes five different types of constraints from real-world scenarios, namely Content (i.e., explicit restrictions on the response content), Situation (i.e., specific situation/background information added to the question), Style (i.e., response style requirements), Format (i.e., response format requirements), and Example (i.e., example pattern recognition and following). To precisely estimate the difficulty degree to which LLMs can follow instructions, as shown in Figure 7.1, we propose a novel *Multi-level* mechanism that incrementally adds a single constraint to straightforward instructions at each increased level. The multi-level mechanism enables us to pinpoint the difficulty level at which LLMs fail to follow instructions, thereby estimating the upper limit of instruction-following capability in LLMs more precisely. Overall, `FollowBench` consists of 820 meticulously curated instructions from over 50 NLP tasks, including both closed- and open-ended questions. For evaluation purposes, we propose a hybrid evaluation method comprising rule-based and model-based solutions. Given LLMs' outputs, both solutions judge whether the outputs satisfy each of the constraints in the instructions. The rule-based solutions focus on closed-ended instructions while the model-based solutions are applied to opened-ended instructions. For model-based solutions, instead of merely using current instructions and responses as input, we additionally provide the evolution process of the instructions in the input prompts to LLM

judges to better understand each individual constraint. Both the data construction and the evaluation undergo human verification.

In our experiments, we propose three metrics to assess the instruction-following ability of 13 prominent closed-source and open-source LLMs on `FollowBench`. Our principal observations are: (1) the performance of all tested models declines substantially with an increase in difficulty level (the number of constraints in an instruction); (2) although closed-source models such as GPT-4 and GPT-3.5 **only** consecutively satisfy around three constraints on average, they still markedly surpass all open-source models; (3) certain specific constraint categories, such as Situation and Example, prove to be more challenging for LLMs than others; (4) beyond capabilities such as knowledge and reasoning, instruction following can offer an additional lens for comprehensively assessing the proficiency of LLMs.

7.2 FollowBench

As shown in Table 7.1, `FollowBench` encompasses five distinct *fine-grained* constraint categories: Content, Situation, Style, Format, and Example. Each category consists of instructions from various NLP tasks. Different from previous benchmarks, we introduce a *Multi-level* mechanism that incrementally adds constraints to an initial instruction (see examples in Figure 7.2), producing a set of instructions ranging from 1 to 5 constraints. In the following part of this paper, we use "level n" to denote an instruction containing n constraints. It is worth noticing that the way of adding constraints is meticulously designed for each task within its respective constraint category. The multi-level mechanism enables us to pinpoint the difficulty level at which LLMs fail to follow instructions, thereby estimating the upper bound of instruction-following capability in LLMs more precisely.

To encapsulate, we will introduce the data construction process of `FollowBench`, including *fine-grained* constraints and the *Multi-level* mechanism, in §7.2.1. In §7.2.2, we propose an evaluation protocol with three metrics that seamlessly integrate with the multi-level mechanism.

Constraint	Task	Avg Len	#Data	Evaluation
Content	Data-to-Text Generation	84	25	
	Document-Level Event Argument Extraction	696	25	
	Document-Level Named Entity Recognition	376	25	
	Text Generation with Language Constraints	88	25	
	Open-ended Question Answering	56	25	
Situation	Suggestion Generation	69	40	
	Role-playing	111	15	
	Complex Situation Reasoning	102	55	
Style	Open-ended Question Answering	64	150	
Format	Text-to-Table Generation	171	30	
	Open-ended Question Answering	74	120	
Example	40 diverse NLP tasks	739	200	
Mixed	Text Editing	96	25	
	Summarization	254	25	
	Machine Translation	91	25	
	Story Generation	34	10	

Table 7.1: An overview of FollowBench. "Avg Len" is the average word number of instructions. refers to rule-based evaluation, while refers to model-based evaluation.

CONTENT	INITIAL	Recommend 5 films to me.
	LEVEL 1	Recommend me 5 Chinese films.
	LEVEL 2	Recommend me 5 Chinese films released before 1990.
SITUATION	INITIAL	How can I increase my productivity while working from home?
	LEVEL 1	Since the pandemic began, I've been working remotely. How can I increase my productivity while working from home?
	LEVEL 2	I have a small child at home. Since the pandemic began, I've been working remotely. How can I increase my productivity while working from home?
STYLE	INITIAL	How did US states get their names?
	LEVEL 1	How did US states get their names? Please respond in the writing style of Shakespeare.
	LEVEL 2	How did US states get their names? Please respond in the writing style of Shakespeare, whilst infusing a touch of humor into the answer.
FORMAT	INITIAL	Why can I see the moon during the day?
	LEVEL 1	Why can I see the moon during the day? Answer in a table format with columns "Reason" and "Explanation".
	LEVEL 2	Why can I see the moon during the day? Answer in a table format with columns "Reason" and "Explanation". Each explanation should not exceed 20 words in length.
EXAMPLE	LEVEL 1	question_template_1.format(example_1) + answer_template_1.format(example_1) question_template_1.format(example_2) + answer_template_1.format(example_2) ⋮ question_template_1.format(query)
	LEVEL 2	question_template_1.format(example_1) + answer_template_1.format(example_1) question_template_2.format(example_2) + answer_template_2.format(example_2) ⋮ question_template_1.format(query)

Figure 7.2: FollowBench covers five *fine-grained* categories of constraints. Within each constraint type, we construct a range of *Multi-level* instructions by incrementally adding constraints (highlighted in red). There are five levels in total; however, we only display the first two levels from each category for demonstration purposes.

7.2.1 Data Construction

Content Constraints. Content constraints refer to *explicit* impositions of specific conditions that shape the depth or scope of the response content. An example is shown in Figure 7.2, which sets specific criteria for the retrieved object. Ensuring that LLMs adhere to content constraints has become a critical challenge in Controlled Text Generation [205], as it demands models to understand specific guidelines and adapt responses to prescribed conditions [23]. To this end, we first collect data from the following tasks: (1) Complex Information Extraction aims at retrieving specific information about specific objects from the given text; (2) Text Generation with Language Constraints requires to generate fluent ontopic content while respecting a specified constraint; (3) Open-ended Question Answering comes from real scenarios (e.g., open-source platforms) to prevent the risk of data leakage. Subsequently, we construct multi-level instructions by adding one content constraint to the collected instructions each time. The manners of introducing additional constraints depend on different tasks (see details in Appendix E.1). For Complex Information Extraction, we gradually narrow down the scope of the information to be extracted. For Text Generation with Language Constraints, we incorporate additional restrictions from WordNet [124] and Wikidata [174]. For Open-ended Question Answering, we utilize advanced LLMs like GPT-4 to generate a new instruction with one more constraint based on the given instruction. While the output from the LLMs serves primarily as a reference, we handpick the most relevant and challenging synthesized instructions to ensure data quality.

Situation Constraints. Situation Constraints refer to impositions of specific situations or backgrounds that *implicitly* guide the appropriate answer of the response. For instance, it is necessary to illustrate the situation when asking for customized suggestions, as shown in Figure 7.2. Another example is to customize LLMs to simulate various characters under certain circumstances, namely Role-playing, which provides a more nuanced interaction for users [155, 182]. Situation constraints push LLMs beyond mere factual retrieval or surface-level synthesis, demanding a nuanced understanding, a dynamic adaptation, and complicated reasoning to the situation [198, 114]. Besides real-life questions, we also consider Complex Situation Reasoning tasks including Math Word Problems, Time/Spatial

Reasoning, and Code Generation. These tasks all require interpreting and solving problems within a given situation, thus matching the definition of situation constraints. We first collect initial instructions from these sources and then manually curate multi-level instructions by incrementally supplementing situation information inside (see Appendix E.1.2).

Style Constraints. Style Constraints control the stylistic variations of output to accomplish specific stylistic goals, such as tone, sentiment, formality, and empathy [172], as illustrated in Figure 7.2. The challenges of style constraints for LLMs are the intricate understanding and adaptation of language nuances, ensuring contextually appropriate and stylistically consistent outputs [159, 29]. Drawing from Open-ended Question Answering datasets and online platforms, we collect initial instructions and then leverage LLMs' in-context learning capability to craft instructions with multi-level style constraints. The prompt template can be viewed in Figure E.2. Human experts subsequently review and refine the outputs produced by LLMs.

Format Constraints. Format Constraints refer to stipulations governing the structural, linguistic, or output presentation of generated content. An example is shown in Figure 7.2, which sets limits on word length and requires the format of the response to be a table. Format constraints necessitate a deep, nuanced understanding of language and structure, allowing them to flexibly adapt outputs according to diverse and often intricate specifications [210]. Recent work has pointed out that even the most superior LLMs may struggle with tasks that require generating complex, structured outputs such as tables, JSON, HTML, or LaTeX [163]. To include a variety of format constraints, we first collect instructions from broader domains, encompassing Text-to-Table Generation and Open-ended Question Answering, then we utilize powerful LLMs to sequentially add format constraints ranging from length and hierarchy to specialized linguistic features and output mediums. See Figure E.3 for the prompt template. Finally, we ask human experts to carefully check and refine the synthesized instructions.

Example Constraints. LLMs have demonstrated stunning few-shot learning ability [16], which enables them to adapt quickly to a new query by recognizing patterns from just

a few examples provided in the prompt. However, the robustness of few-shot learning, which means whether LLMs can still follow correct patterns after introducing "noise" examples, has not been explored. Thus, we propose a novel constraint category named Example Constraints to evaluate the example pattern recognition and following capability of LLMs. We automatically craft instructions with multi-level example constraints based on PromptSource [7], where instructions at level n have $n-1$ noise examples in the input. The details are illustrated in Appendix E.1.3.

Mixed Constraints. For the above five constraint categories, we construct multi-level instructions by adding the same type of constraint sequentially. Nevertheless, real-world scenarios often require more than one type of constraint to be enforced in a singular instruction. Therefore, we define Mixed Constraints as the composition of varied constraint categories. For instance, in the Text Editing task, we may want to add some content as well as adjust the output format. Besides, we also consider several tasks that are naturally suitable for constructing mixed constraints, including Summarization, Machine Translation, and Story Generation (see Appendix E.1.4). Instructions with multi-level mixed constraints are produced by specifying the format of generating answers (Format Constraints), requiring the generated text to include or not include certain keywords (Content Constraints), etc.

Data Quality Control. To ensure the data quality of `FollowBench`, we implement a dual-layer verification system for each instruction. Two annotators independently evaluate: (1) the appropriateness of the instruction for its designated constraint category, and (2) the validity of the added constraint within the instruction. In instances of divergent evaluations, a third annotator intervenes for a detailed review to ensure consensus.

We analyze the comprehensiveness and diversity of in `FollowBench`, which includes 820 instructions in total. To maintain data diversity, we strive to ensure that the ROUGE-L score between any two initial instructions is below 0.7. Figure 7.3 shows the verb-noun structure of `FollowBench` instructions, where the top 20 verbs (inner circle) and their top 4 direct noun objects (outer circle) are depicted.

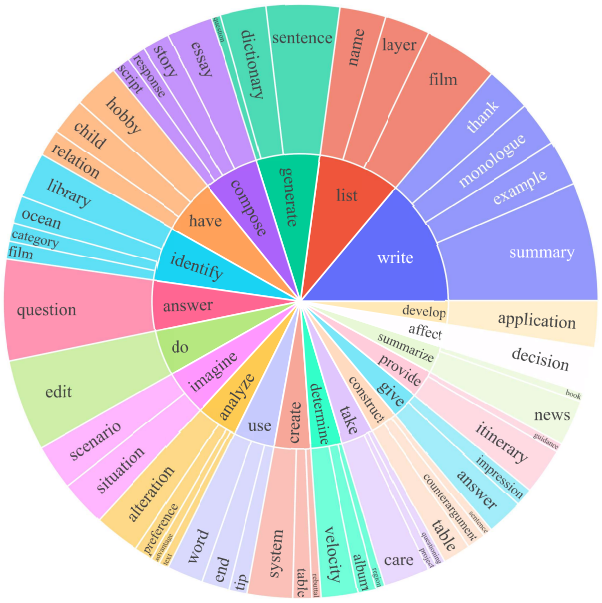


Figure 7.3: Verb-noun structure of `FollowBench` Instructions.

Prompt Template for Model-based Evaluation

Given an initial instruction, we add one **{constraint_type}** constraint per time and obtain the final instruction with **{level_n}** additional constraints.

#Initial Instruction#
{initial_instruction}

#Initial Instruction + 1 constraint#
{level_1_instruction}
⋮
#Initial Instruction + **{level_n}** constraints#
{level_n_instruction}

#Answer of Initial Instruction + **{level_n}** constraints#
{answer_of_level_n_instruction}

#System#
1) Please identify all **{level_n}** added constraints.
2) For the **{level_n}** added constraints, discriminate if the #Answer of Initial Instruction + **{level_n}** constraints# satisfies each constraint.
3) In the final line, only output a Python LIST with **{level_n}** elements ('YES' or 'NO') indicating whether the answer satisfies each constraint.

Figure 7.4: Prompt template for model-based evaluation.

7.2.2 Evaluation Protocol

Given that nearly half of instructions in `FollowBench` are open-ended without reference answers, devising a rule-based program to assess the outputs is extremely challenging. To overcome this, inspired by [61, 78], we propose to develop a model-based approach by using strong LLMs[1] as judges. Previous works leverage strong LLMs to determine the quality of a response, by prompting them to consider multiple factors such as usefulness, relevance, and level of detail [107, 211]. To effectively guide strong LLMs to judge the constraint following capability objectively and faithfully, we propose a *Multi-level-aware* prompt template, as shown in Figure 7.4. Rather than merely presenting the instruction and asking LLMs to determine whether all constraints are satisfied, we illustrate the evolution process of the instruction and prompt LLMs to pinpoint the newly added constraint at each level. Exposing the evolution process of the instruction allows for a more granular understanding and identification of individual constraints, enhancing LLMs' ability to discriminate with precision. The ablation study in §7.4.1 validates the effectiveness of this strategy.

Moreover, we propose three novel metrics to evaluate the instruction-following ability

[1]We use GPT-4-Preview-1106 in our experiments.

of LLMs. For an instruction with n constraints (level n), we use the rule-based program or LLM judge (refer to Table 7.1) to discriminate if the response of a model satisfies each constraint in the instruction. At each level n, given a set of m instructions, we define the Hard Satisfaction Rate (HSR) and Soft Satisfaction Rate (SSR) as follows:

$$\text{HSR} = \frac{1}{m} \sum_{i=1}^{m} \prod_{j=1}^{n} s_i^j \tag{7.1}$$

$$\text{SSR} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} s_i^j \tag{7.2}$$

where $s_i^j = 1$ if the j-th constraint of i-th instruction is satisfied and $s_i^j = 0$ otherwise. HSR measures the average rate at which all constraints of individual instructions are fully satisfied, while SSR calculates the average satisfaction rate of individual constraints across all instructions.

As described in §7.2, we construct `FollowBench` by incrementally adding five constraints to an initial instruction, enabling us to pinpoint the difficulty level at which LLMs fail to follow instructions. Therefore, we propose a metric called Consistent Satisfaction Levels (CSL) to estimate how many consecutive levels a model can satisfy, beginning from level 1:

$$\text{CSL} = \frac{1}{g} \sum_{i=1}^{g} \arg\max_{l} \left(l \times \prod_{n=1}^{l} S_i^n \right) \tag{7.3}$$

where g is the group number of initial instructions, $S_i^n = 1$ if all constraints of the i-th instruction at level-n are satisfied and $S_i^n = 0$ otherwise.

7.3 Experiments

This section first introduces experimental setup in §7.3.1, and then presents the main experiment results across two key dimensions: difficulty level in §7.3.2 and constraint category in §7.3.3.

7.3.1 Experimental Setup

We evaluate 13 popular LLMs including GPT-4-Preview-1106 [135], GPT-3.5-Turbo-1106 [136], Qwen-Chat-72B/14B/7B [8], LLaMA2-Chat-70B/13B/7B [171], WizardLM-13B-V1.2 [194], Vicuna-13B/7B-V1.5 [211], Baichuan2-Chat-7B [11], and ChatGLM3-6B [50]. We access GPT-4-Preview-1106 and GPT-3.5-Turbo-1106 via OpenAI API. We access other open-source LLMs from their official repositories. During the inference process, we set the temperature to 0 to ensure deterministic outputs. We set the maximum generation length to 2048. Other parameters use their default values.

7.3.2 Level-categorized Results

Table 7.2 provides a comprehensive comparison of various models across five difficulty levels, denoted as L1 to L5. From a bird's-eye view, we can infer that the performance typically diminishes as we progress from L1 to L5 for almost all models. This trend coincides with the increasing complexity or stringent requirements associated with higher levels. Besides, models with larger architectures generally outperform their smaller counterparts. However, it's worth noting that the scaling law does not apply as effectively to LLaMA2-Chat-70B. The reason is that while LLaMA-2-Chat-70B does indeed outperform LLaMA-2-Chat-13B in Situation constraints, it shows a relative underperformance in Format and Mixed Constraints categories. More importantly, there's a marked performance gap between closed-source models (i.e., GPT-4 and GPT-3.5) and open-source models. Regarding CSL, it can be deduced that the instruction-following upper bound for GPT-4 and GPT-3.5 is approximately 3 constraints (level 3) added to an initial instruction. In contrast, open-source models typically have an upper limit of about 2 constraints (level 2). This significant difference underscores the better instruction-following ability of proprietary models, possibly due to superior data quality or optimization strategies such as RLHF [137]. Furthermore, **even the most sophisticated models are limited to following instructions with about three constraints**, suggesting significant potential for further improvement.

Model	HSR (%)						SSR (%)						CSL
	L1	L2	L3	L4	L5	Avg.	L1	L2	L3	L4	L5	Avg.	
GPT-4-Preview-1106	**84.7**	**75.6**	**70.8**	**73.9**	**61.9**	**73.4**	**84.7**	**77.0**	**75.3**	**77.0**	**72.3**	**77.2**	**3.3**
GPT-3.5-Turbo-1106	80.3	68.0	68.6	61.1	53.2	66.2	80.3	71.2	74.2	69.6	67.1	72.5	2.9
Qwen-Chat-72B	73.8	63.3	54.3	45.2	39.9	55.3	73.8	67.5	63.2	57.6	56.0	63.6	2.4
LLaMA2-Chat-70B	59.9	53.3	46.0	40.2	37.9	47.5	59.9	57.3	55.7	53.3	53.2	55.9	2.1
Qwen-Chat-14B	62.8	56.2	47.7	38.7	30.9	47.3	62.8	61.9	57.7	52.6	51.4	57.3	1.9
WizardLM-13B-V1.2	68.8	64.1	53.1	40.8	35.8	52.5	68.8	65.7	61.8	53.4	53.9	60.7	2.2
LLaMA2-Chat-13B	57.0	56.0	50.4	44.4	38.1	49.2	57.0	60.0	58.0	54.8	52.2	56.4	2.2
Vicuna-13B-V1.5	71.2	60.2	49.6	40.6	34.0	51.1	71.2	64.8	59.9	54.5	53.6	60.8	2.1
Qwen-Chat-7B	55.9	51.7	38.7	33.1	23.3	40.6	55.9	58.2	51.6	48.9	45.9	52.1	1.5
LLaMA2-Chat-7B	58.0	51.3	47.4	39.5	35.3	46.3	58.0	56.5	55.6	52.5	51.4	54.8	1.9
Vicuna-7B-V1.5	60.8	52.0	42.2	33.3	23.9	42.4	60.8	58.6	55.5	48.3	49.0	54.4	1.7
Baichuan2-Chat-7B	58.3	46.1	40.7	30.4	25.5	40.2	58.3	55.4	54.9	49.9	49.3	53.6	1.4
ChatGLM3-6B	60.9	46.6	36.7	27.8	21.4	38.7	60.9	55.3	51.2	47.9	45.0	52.0	1.6

Table 7.2: Results across five difficulty levels. For each level, we compute the average score of all constraint categories. Proprietary LLMs, open-sourced LLMs (large), open-sourced LLMs (medium), and open-sourced LLMs (small) are distinguished by different colors.

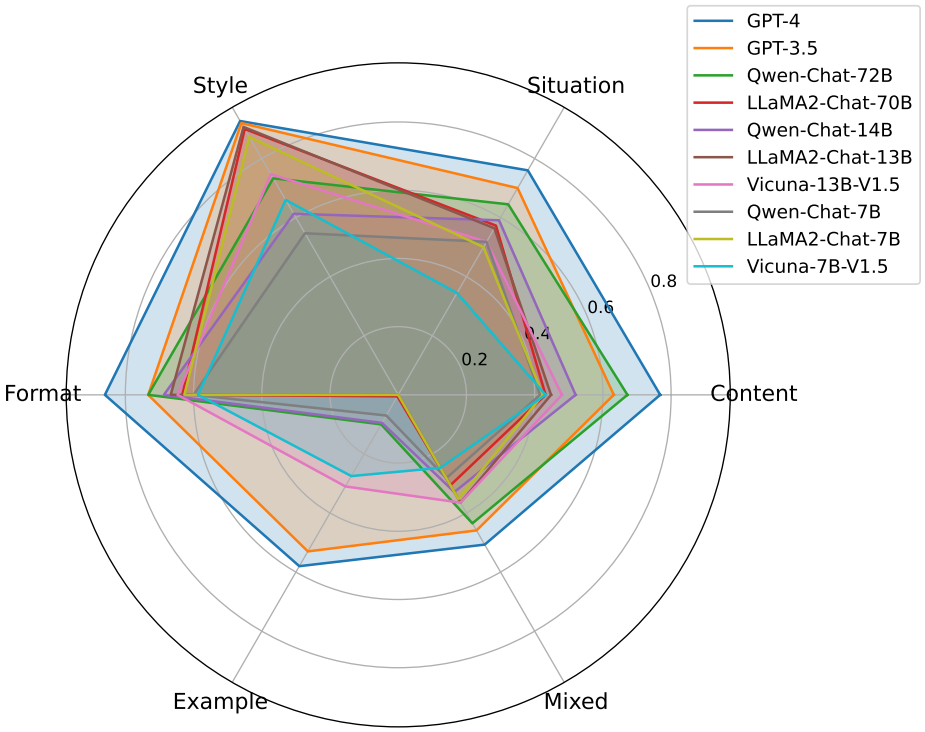


Figure 7.5: HSR (%) results in diverse constraint categories. For each category, we compute the average score of all difficulty levels.

7.3.3 Constraint-categorized Results

As depicted in Figure 7.5, we assess various models over different constraint categories to succinctly showcase the instruction-following capability of LLMs in a singular dimension. Notably, GPT-4 and GPT-3.5 surpass open-source models in every constraint category, with a pronounced advantage in Content, Situation, Example, and Mixed constraints. Furthermore, most models demonstrated commendable proficiency under the Style constraint. While GPT-4, GPT-3.5, and LLaMA2-Chat-70B were the frontrunners, the trend suggests that style adaptation is an area where many models excel, hinting at its utility in real-world applications. However, the Example and Mixed constraints posed a challenge to most models. While GPT-4 led the segment, even its scores were noticeably lower than in other categories. To illustrate, in the "Example" category, we evaluated the instruction-following capabilities of LLMs by introducing "noise examples" with varying natural language templates. The observed performance decline is primarily due to the LLMs' limited training in processing such noisy inputs within context-based learning scenarios. Typically, LLMs are fine-tuned on clean and uniform datasets, which do not adequately prepare them to sift through and ignore irrelevant or misleading information. This limitation becomes apparent when faced with the intricacies of real-world data. Our findings underscore the complexity of these constraints and pinpoint an area for potential improvement.

7.4 Analysis

This section includes: an ablation study confirming our prompt template's effectiveness for model-based evaluation (§7.4.1); a comparison of instruction following vs. other LLM's abilities (§7.4.2); an examination of failure consistency (§7.4.3); and an investigation of various decoding strategies (§7.4.4).

7.4.1 Ablation Study of Model-based Evaluation

We randomly sample 100 cases that require LLM evaluation, encompassing five constraints, five distinct levels, and four diverse models to guarantee comprehensive rep-

resentation. Then we ask three expert-level human labelers to assess whether the model's response satisfies all the constraints in each case and use the majority voting as the final human annotations. As shown in Table 7.3, our prompt template (Figure 7.4) registers an impressive 88% agreement with expert human evaluations, surpassing even the internal agreement among human experts, which stands at 85%. Remarkably, when the evolution process of multi-level constraints is removed from our prompt template, the agreement rate dips by 9%. This underlines the instrumental role played by the detailed portrayal of the instruction's evolution in enhancing LLM's precision in discernment. In contrast, we also employ the prompt template from Vicuna [211], a standard prompt for assessing the overall quality of response. This template prompts the LLM to assign a score from 0 to 10 for each response. We consider responses with a score above 5.0 to meet all the constraints of an instruction. This approach achieves 67% agreement with human evaluators. Such a disparity highlights the fundamental difference between assessing the instruction-following ability and the overall response quality.

Prompt	**Agreement with Human**
Ours	**88%**
Ours w/o ML	79%
Vicuna-Single	67%

Table 7.3: Agreement between human and diverse prompt templates. We use ML to denote multi-level.

7.4.2 Instruction Following vs. Other Abilities

Table 7.4 presents a comparison of representative LLMs across different abilities, not just instruction following (`FollowBench`). This includes overall response quality (AlpacaEval [107]), knowledge (MMLU [72]), and reasoning (BBH [161]). We can find that our `FollowBench` provides an additional perspective for a holistic LLM evaluation. As an illustration, while the performance of WizardLM-13B-V1.2 exceeds that of GPT-3.5 in terms of overall response quality, it notably lags behind in instruction-following ability. Similarly, Vicuna-V1.5 excels over LLaMA2-Chat in the realms of knowledge and reasoning but struggles with instruction-following tasks.

Model	Following	Overall	Knowledge	Reasoning
GPT-4-Preview-1106	3.3	97.7	86.4	86.7
GPT-3.5-turbo-1106	2.9	86.3	70.0	70.1
LLaMA2-Chat-70B	2.1	92.7	63.0	60.8
WizardLM-13B-V1.2	2.2	89.2	52.7	–
LLaMA2-Chat-13B	2.2	81.1	53.6	40.2
Vicuna-13B-V1.5	2.1	–	55.8	51.5
LLaMA2-Chat-7B	1.9	71.4	45.8	35.6
Vicuna-7B-V1.5	1.7	–	49.8	43.4

Table 7.4: Model comparison on different abilities.

7.4.3 Does Failure at Lower Level Necessarily Lead to Failure at Higher Level?

For a set of instructions that has five difficulty levels, if a model's response doesn't satisfy the constraints at level n, where n ranges from 1 to 4, we define the *failure consistency* as the percentage that the response will also not fulfill the constraints at any subsequent level greater than n. Combining Table 7.2 and Table 7.5, it can be seen that models with better instruction-following capability may exhibit lower failure consistency. One possible reason is that the instruction-following ability of more powerful models is less sensitive to the number of constraints in an instruction, thus they are better equipped to adapt and fulfill the requirements even as the constraints increase. This adaptability means that while they may falter at a lower difficulty level, they can still manage to meet the demands of higher difficulty levels, leading to a decrease in failure consistency.

Model	Failure Consistency (%)
GPT-4-Preview-1106	42.2
WizardLM-13B-V1.2	57.3
Vicuna-7B-V1.5	61.8
ChatGLM3-6B	64.0

Table 7.5: Results on failure consistency.

7.4.4 Does Different Decoding Strategies Affect the Instruction-following Ability?

In this section, we systematically investigate the impact of different decoding strategies, represented by the temperature parameter τ, on LLM's instruction-following ability. The temperature τ is a commonly used parameter that controls the sharpness of the distribution from which we sample the next token:

$$P(w) = \frac{\exp(z_w/\tau)}{\sum_{w' \in V} \exp(z_{w'}/\tau)} \tag{7.4}$$

where z_w is the logit for word w, V is the vocabulary. Lower values for temperature result in more consistent outputs, while higher values generate more diverse and creative results. As illustrated in Figure 7.6, the temperature τ has a tangible influence on the instruction-following ability across all four models. The sweet spot seems to be somewhere in the middle where there's enough variability to capture the nuances and intricacies of complex instructions, yet not so much that the model goes off tangent. This balanced behavior ensures that the model remains within the desired context, producing outputs that align closely with the given instructions while also allowing for a slight creative touch when needed.

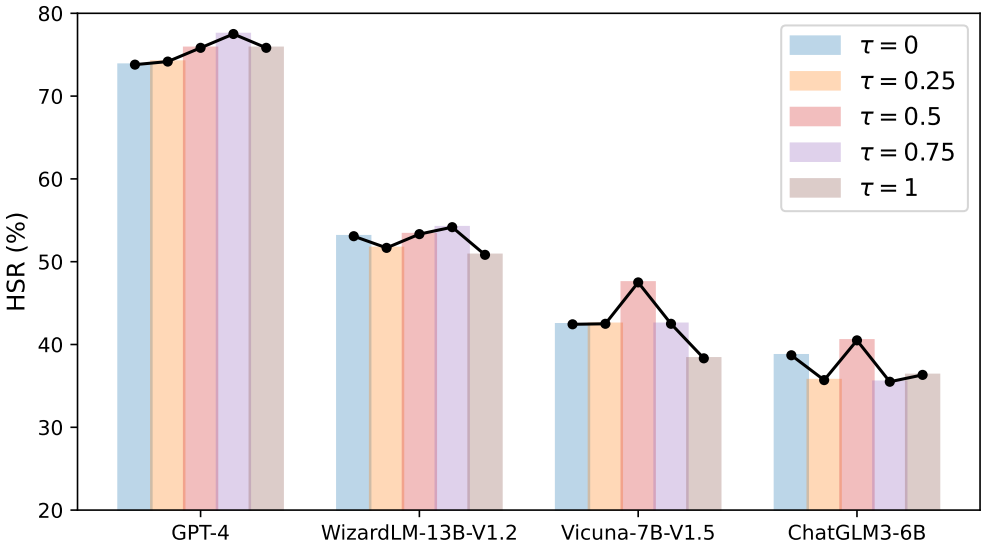


Figure 7.6: The effect of varying the temperature parameter τ. We use $\tau = 0$ to denote greedy decoding.

7.5 Conclusion and Discussion

7.5.1 Conclusion

In this chapter, we introduce `FollowBench`, a Multi-level Fine-grained Constraints Following Benchmark tailored for gauging the instruction-following capability of LLMs. `FollowBench` covers five *fine-grained* constraint categories and over 50 NLP tasks, utilizes a novel *Multi-level* mechanism for precisely estimating the upper limit of instruction-following capability. Furthermore, we propose an evaluation protocol with three metrics that seamlessly integrate with the multi-level mechanism. Our extensive tests over 13 popular LLMs reveal a substantial performance advantage for GPT-4 and GPT-3.5 over their counterparts, and there is still significant room for improving the instruction-following ability of current LLMs.

7.5.2 Discussion

While our study contributes valuable insights, it is essential to acknowledge several limitations that warrant consideration.

Firstly, our current investigation is confined to single-round interactions, aiming to offer a controlled environment for evaluation. Future research may extend its scope to multi-round conversations to comprehensively assess the instruction-following proficiency of LLMs in more dynamic and extended dialogues [93].

Secondly, the model-based evaluation framework employed in our experiments, while rigorous, relies on prompt engineering, introducing an inherent imperfection. Despite our meticulous selection of high-performing prompts, the potential for further optimization remains, which may impact the reported evaluation metrics.

Lastly, we refrain from proposing specific solutions to address identified weaknesses of LLMs in instruction following. A plausible avenue for future research involves fine-tuning LLMs using our proposed `FollowBench` as a benchmark, providing a potential roadmap for enhancing instruction adherence. We defer the exploration of these aspects to subsequent studies, recognizing the need for a comprehensive examination of LLM capabilities across varying interaction complexities.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 Conclusion

In this thesis, we have explored the efficient and effective alignment of LLMs across multiple dimensions, including data synthesis, training, and evaluation. Our research introduces novel methodologies that improve the scalability, adaptability, and reliability of alignment processes, addressing key challenges in instruction-following, knowledge retention, and preference modeling.

Summary of Contributions:

- **Alignment Data Synthesis via Adversarial Distillation** – We proposed an adversarial knowledge distillation framework that iteratively refines a student model by generating hard instructions and incorporating feedback from a proprietary LLM. This method enhances knowledge transfer efficiency and improves model alignment performance.
- **Web Reconstruction for Scalable Instruction-Tuning Data** – We introduced Web Reconstruction (WebR), a framework that synthesizes high-quality instruction-tuning datasets from web content using a dual-perspective paradigm. Our experiments demonstrate that WebR-generated data significantly improve LLM alignment across multiple benchmarks.
- **Learning to Edit (LTE) for Knowledge Adaptation** – We developed LTE, a two-phase knowledge editing framework that enables real-time knowledge updates while maintaining model consistency. LTE outperforms prior methods in robustness and efficiency, facilitating dynamic adaptation of LLMs to evolving information.

- **Bridging and Modeling Correlations (BMC) in Preference Optimization** – We proposed BMC, a novel approach to direct preference optimization that enhances human value alignment by explicitly modeling fine-grained preference signals. BMC consistently surpasses standard DPO, improving performance in various tasks.
- **`FollowBench`: A Fine-Grained Evaluation Benchmark for Constraint Following** – We introduced `FollowBench`, a multi-level benchmarking framework for assessing LLMs' instruction-following capabilities. Our evaluations highlight existing gaps in alignment and provide a standardized protocol for future improvements.

8.2 Future Work

The advancements in this thesis contribute significantly to refining LLM alignment strategies, ensuring more effective, adaptable, and robust AI models. However, several challenges remain, opening avenues for future research:

- **Scalability and Computational Efficiency** – Expanding adversarial distillation and web-based data synthesis to scale alignment processes without excessive computational costs. Techniques such as continual learning and reinforcement learning could further enhance efficiency.
- **Multi-Turn and Interactive Alignment** – Our proposed approaches primarily focus on single-turn tasks. Extending these methods to handle multi-turn interactions and real-time user feedback will be crucial for enhancing conversational AI capabilities.
- **Beyond Factual Knowledge Editing** – Extending knowledge editing frameworks to non-factual aspects, such as personality, sentiment, and ethical reasoning, could enable more nuanced model adaptations. Additionally, advancing black-box model editing techniques would improve accessibility for proprietary LLMs.
- **Advancing Preference Optimization for Human-AI Alignment** – Incorporating more fine-grained human preferences, cultural considerations, and ethical constraints into alignment processes to ensure AI systems reflect societal values more accurately.

- **Robust and Adaptive Evaluation Metrics** – Developing more comprehensive benchmarking methodologies that integrate human feedback, adversarial testing, and real-world deployment scenarios to assess alignment robustness across diverse applications.

In conclusion, this thesis lays the groundwork for future research in LLM alignment, pushing towards models that are not only more powerful but also more efficient, adaptable, and aligned with human intent.

8.3 List of Publications

- **Yuxin Jiang**, Yufei Wang, Chuhan Wu, Xinyi Dai, Yan Xu, Weinan Gan, Yasheng Wang, Xin Jiang, Lifeng Shang, Ruiming Tang, and Wei Wang. Instruction-Tuning Data Synthesis from Scratch via Web Reconstruction. In *Findings of the Association for Computational Linguistics: ACL 2025*. Association for Computational Linguistics, 2025.

- **Yuxin Jiang**, Bo Huang, Yufei Wang, Xingshan Zeng, Liangyou Li, Yasheng Wang, Xin Jiang, Lifeng Shang, Ruiming Tang, and Wei Wang. Bridging and modeling correlations in pairwise data for direct preference optimization. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

- **Yuxin Jiang**, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, Qun Liu, and Wei Wang. Learning to edit: Aligning LLMs with knowledge editing. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4689–4705. Association for Computational Linguistics, 2024.

- **Yuxin Jiang**, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. FollowBench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4667–4688. Association for Computational Linguistics, 2024.

- **Yuxin Jiang**, Chunkit Chan, Mingyang Chen, and Wei Wang. Lion: Adversarial distillation of proprietary large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3134–3154. Association for Computational Linguistics, 2023.

- **Yuxin Jiang**, Linhan Zhang, and Wei Wang. Global and local hierarchy-aware contrastive framework for implicit discourse relation recognition. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8048–8064. Association for Computational Linguistics, 2023.

- **Yuxin Jiang**, Linhan Zhang, and Wei Wang. Improved universal sentence embeddings with prompt-based contrastive learning and energy-based learning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3021–3035. Association for Computational Linguistics, 2022.

BIBLIOGRAPHY

[1] Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. Explanations for commonsenseqa: New dataset and models. In *Annual Meeting of the Association for Computational Linguistics*, pages 3050–3065, 2021.

[2] Zachary Ankner, Cody Blakeney, Kartik Sreenivasan, Max Marion, Matthew L Leavitt, and Mansheej Paul. Perplexed by perplexity: Perplexity-based data pruning with small reference models. *arXiv preprint arXiv:2405.20541*, 2024.

[3] arXiv.org submitters. arxiv dataset, 2023.

[4] Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

[5] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[6] Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Rémi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, volume 238, pages 4447–4455, 2024.

[7] Stephen Bach, Victor Sanh, Zheng Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-david, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Fries, Maged Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-jian

Jiang, and Alexander Rush. PromptSource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[8] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[9] 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.

[10] 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. In *arXiv*, 2022.

[11] Baichuan. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*, 2023.

[12] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press, 2020.

[13] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Syd-

ney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[14] 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.

[15] Andrei Z Broder. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pages 21–29. IEEE, 1997.

[16] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[17] Alexander Bukharin and Tuo Zhao. Data diversity matters for robust instruction tuning. *arXiv preprint arXiv:2311.14736*, 2023.

[18] Meng Cao, Lei Shu, Lei Yu, Yun Zhu, Nevan Wichers, Yinxiao Liu, and Lei Meng. Drlc: Reinforcement learning with dense rewards from llm critic. In *arXiv*, 2024.

[19] Yihan Cao, Yanbin Kang, and Lichao Sun. Instruction mining: High-quality instruction data selection for large language models. *arXiv preprint arXiv:2307.06290*, 1(3):6, 2023.

[20] Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit3: Web inventory of transcribed and translated talks. In *Proceedings of the Conference of European Association for Machine Translation (EAMT)*, pages 261–268, 2012.

[21] Alex J Chan, Hao Sun, Samuel Holt, and Mihaela van der Schaar. Dense reward for free in reinforcement learning from human feedback. In *arXiv*, 2024.

[22] Hao Chen, Yiming Zhang, Qi Zhang, Hantao Yang, Xiaomeng Hu, Xuetao Ma, Yifan Yanggong, and Junbo Zhao. Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning. *arXiv preprint arXiv:2305.09246*, 2023.

[23] Howard Chen, Huihan Li, Danqi Chen, and Karthik Narasimhan. Controllable text generation with language constraints. *arXiv preprint arXiv:2212.10466*, 2022.

[24] Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*, 2023.

[25] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. 2021.

[26] Yongrui Chen, Haiyun Jiang, Xinting Huang, Shuming Shi, and Guilin Qi. DoG-instruct: Towards premium instruction-tuning data via text-grounded instruction wrapping. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4125–4135, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[27] Zhipeng Chen, Kun Zhou, Wayne Xin Zhao, Junchen Wan, Fuzheng Zhang, Di Zhang, and Ji-Rong Wen. Improving large language models via fine-grained

reinforcement learning with minimum editing constraint. In *Findings of the Association for Computational Linguistics*, pages 5694–5711, 2024.

[28] Zhipeng Chen, Kun Zhou, Wayne Xin Zhao, Jingyuan Wang, and Ji-Rong Wen. Low-redundant optimization for large language model alignment. In *arXiv*, 2024.

[29] Pengyu Cheng and Ruineng Li. Replacing language model for style transfer. *arXiv preprint arXiv:2211.07343*, 2022.

[30] Siyuan Cheng, Bozhong Tian, Qingbin Liu, Xi Chen, Yongheng Wang, Huajun Chen, and Ningyu Zhang. Can we edit multimodal large language models? In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13877–13888, Singapore, December 2023. Association for Computational Linguistics.

[31] 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.

[32] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[33] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai,

Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24:240:1–240:113, 2023.

[34] Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, pages 4299–4307, 2017.

[35] Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A Smith. All that's' human'is not gold: Evaluating human evaluation of generated text. *arXiv preprint arXiv:2107.00061*, 2021.

[36] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

[37] Jan Clusmann, Fiona R Kolbinger, Hannah Sophie Muti, Zunamys I Carrero, Jan-Niklas Eckardt, Narmin Ghaffari Laleh, Chiara Maria Lavinia Löffler, Sophie-Caroline Schwarzkopf, Michaela Unger, Gregory P Veldhuizen, et al. The future landscape of large language models in medicine. *Communications medicine*, 3(1):141, 2023.

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

[39] Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *CoRR*, abs/2307.12976, 2023.

[40] Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, 2023.

[41] 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, 2023.

[42] OpenCompass Contributors. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass, 2023.

[43] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. In *arXiv*, 2023.

[44] Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8493–8502. Association for Computational Linguistics, 2022.

[45] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[46] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[47] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, 2023.

[48] 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.

[49] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[50] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, 2022.

[51] 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. In *arXiv*, 2024.

[52] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. In *arXiv*, 2024.

[53] Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics*, pages 9367–9385, 2024.

[54] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[55] Gongfan Fang, Jie Song, Chengchao Shen, Xinchao Wang, Da Chen, and Mingli Song. Data-free adversarial distillation. *CoRR*, abs/1912.11006, 2019.

[56] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 6556–6576, 2024.

[57] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning and ranking by pairwise comparison. In *Preference learning*, pages 65–82. Springer, 2010.

[58] Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.

[59] Xinyang Geng, Arnav Gudibande, Hao Liu, Eric Wallace, Pieter Abbeel, Sergey Levine, and Dawn Song. Koala: A dialogue model for academic research. *Blog post, April*, 1, 2023.

[60] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

[61] Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd-workers for text-annotation tasks. *arXiv preprint arXiv:2303.15056*, 2023.

[62] Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China, November 2019. Association for Computational Linguistics.

[63] Google. Bard, 2023.

[64] David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34, 2003.

[65] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.

[66] Arnav Gudibande, Eric Wallace, Charlie Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. The false promise of imitating proprietary llms. *CoRR*, abs/2305.15717, 2023.

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

[68] Geyang Guo, Ranchi Zhao, Tianyi Tang, Xin Zhao, and Ji-Rong Wen. Beyond imitation: Leveraging fine-grained quality signals for alignment. In *International Conference on Learning Representations*, 2024.

[69] Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

[70] Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. In *Advances in Neural Information Processing Systems*, 2023.

[71] Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models, 2023.

[72] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[73] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

[74] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge distillation with adversarial samples supporting decision boundary. In *The Thirty-Third*

AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, pages 3771–3778. AAAI Press, 2019.

[75] Jiwoo Hong, Noah Lee, and James Thorne. Orpo: Monolithic preference optimization without reference model. In *arXiv*, 2024.

[76] 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.

[77] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[78] F Huang, H Kwak, and J An. Is chatgpt better than human annotators? potential and limitations of chatgpt in explaining implicit hate speech. arxiv, 2023.

[79] Yanhua Huang. Deep q-networks. *Deep reinforcement learning: fundamentals, research and applications*, pages 135–160, 2020.

[80] Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Yao Fu, et al. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *Advances in Neural Information Processing Systems*, 36:62991–63010, 2023.

[81] Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. Camels in a changing climate: Enhancing lm adaptation with tulu 2, 2023.

[82] Jiaming Ji, Boyuan Chen, Hantao Lou, Donghai Hong, Borong Zhang, Xuehai Pan, Juntao Dai, and Yaodong Yang. Aligner: Achieving efficient alignment through weak-to-strong correction. In *arXiv*, 2024.

[83] 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. In *arXiv*, 2023.

[84] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

[85] Jayashree Kalpathy-Cramer, J Peter Campbell, Deniz Erdogmus, Peng Tian, Dharanish Kedarisetti, Chace Moleta, James D Reynolds, Kelly Hutcheson, Michael J Shapiro, Michael X Repka, et al. Plus disease in retinopathy of prematurity: improving diagnosis by ranking disease severity and using quantitative image analysis. *Ophthalmology*, 123(11):2345–2351, 2016.

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

[87] Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. *arXiv preprint arXiv:2312.14925*, 10, 2023.

[88] Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8082–8090, 2020.

[89] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *arXiv*, 2014.

[90] Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. MAWPS: A math word problem repository. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1152–1157, 2016.

[91] 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*, 36:47669–47681, 2023.

[92] Sumith Kulal, Panupong Pasupat, Kartik Chandra, Mina Lee, Oded Padon, Alex Aiken, and Percy S Liang. Spoc: Search-based pseudocode to code. *Advances in Neural Information Processing Systems*, 32, 2019.

[93] Wai-Chung Kwan, Xingshan Zeng, Yuxin Jiang, Yufei Wang, Liangyou Li, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. Mt-eval: A multi-turn capabilities evaluation benchmark for large language models. *CoRR*, abs/2401.16745, 2024.

[94] Jinqi Lai, Wensheng Gan, Jiayang Wu, Zhenlian Qi, and Philip S Yu. Large language models in law: A survey. *AI Open*, 2024.

[95] Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-tau Yih, Daniel Fried, Sida Wang, and Tao Yu. Ds-1000: A natural and reliable benchmark for data science code generation. In *International Conference on Machine Learning*, pages 18319–18345. PMLR, 2023.

[96] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. RLAIF vs. RLHF: scaling reinforcement learning from human feedback with AI feedback. In *International Conference on Machine Learning*, 2024.

[97] Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

[98] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In Roger Levy and Lucia Specia, editors, *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pages 333–342. Association for Computational Linguistics, 2017.

[99] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for

knowledge-intensive NLP tasks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[100] Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. Cmmlu: Measuring massive multitask language understanding in chinese. *arXiv preprint arXiv:2306.09212*, 2023.

[101] Haoran Li, Qingxiu Dong, Zhengyang Tang, Chaojun Wang, Xingxing Zhang, Haoyang Huang, Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, et al. Synthetic data (almost) from scratch: Generalized instruction tuning for language models. *arXiv preprint arXiv:2402.13064*, 2024.

[102] 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. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7602–7635, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[103] Sha Li, Heng Ji, and Jiawei Han. Document-level event argument extraction by conditional generation. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, 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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 894–908. Association for Computational Linguistics, 2021.

[104] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From live data to high-quality benchmarks: The arena-hard pipeline, 2024.

[105] Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Ja-

son E Weston, and Mike Lewis. Self-alignment with instruction backtranslation. In *The Twelfth International Conference on Learning Representations*, 2024.

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

[107] Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`, 2023.

[108] Yinheng Li, Shaofei Wang, Han Ding, and Hang Chen. Large language models in finance: A survey. In *Proceedings of the fourth ACM international conference on AI in finance*, pages 374–382, 2023.

[109] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[110] Pierre Lison and Jörg Tiedemann. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

[111] Chuang Liu, Renren Jin, Yuqi Ren, Linhao Yu, Tianyu Dong, Xiaohan Peng, Shuting Zhang, Jianxiang Peng, Peiyi Zhang, Qingqing Lyu, et al. M3ke: A massive multi-level multi-subject knowledge evaluation benchmark for chinese large language models. *arXiv preprint arXiv:2305.10263*, 2023.

[112] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, 36:21558–21572, 2023.

[113] Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. In *International Conference on Learning Representations*, 2024.

[114] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023.

[115] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[116] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[117] Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *International Conference on Learning Representations*, 2023.

[118] Ian R McKenzie, Alexander Lyzhov, Michael Pieler, Alicia Parrish, Aaron Mueller, Ameya Prabhu, Euan McLean, Aaron Kirtland, Alexis Ross, Alisa Liu, et al. Inverse scaling: When bigger isn't better. *arXiv preprint arXiv:2306.09479*, 2023.

[119] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[120] Kevin Meng, Arnab Sen Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

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

[122] Paul Micaelli and Amos J. Storkey. Zero-shot knowledge transfer via adversarial belief matching. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9547–9557, 2019.

[123] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, 2018.

[124] George A. Miller. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*, 1992.

[125] Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-mishra-etal-2022-cross. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3470–3487, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[126] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[127] Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. Memory-based model editing at scale. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 15817–15831. PMLR, 2022.

[128] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the*

2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 839–849, San Diego, California, June 2016. Association for Computational Linguistics.

[129] Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Hassan Awadallah. Orca: Progressive learning from complex explanation traces of GPT-4. *CoRR*, abs/2306.02707, 2023.

[130] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[131] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1797–1807. Association for Computational Linguistics, 2018.

[132] Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[133] Thao Nguyen, Jeffrey Li, Sewoong Oh, Ludwig Schmidt, Jason Weston, Luke Zettlemoyer, and Xian Li. Better alignment with instruction back-and-forth translation, 2024.

[134] Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany, August 2017. Association for Computational Linguistics.

[135] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[136] TB OpenAI. Chatgpt: Optimizing language models for dialogue. *OpenAI*, 2022.

[137] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Train-

ing language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[138] Meltem Öztürk, Alexis Tsoukiàs, and Philippe Vincke. Preference modelling. *Multiple criteria decision analysis: State of the art surveys*, 78:27–59, 2005.

[139] Alizée Pace, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. West-of-n: Synthetic preference generation for improved reward modeling. In *ICLR 2024 Workshop on Navigating and Addressing Data Problems for Foundation Models*, 2024.

[140] Alexander Pan, Kush Bhatia, and Jacob Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. *arXiv preprint arXiv:2201.03544*, 2022.

[141] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

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

[143] Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. Openwebmath: An open dataset of high-quality mathematical web text. In *The Twelfth International Conference on Learning Representations*, 2024.

[144] Adam Pauls and Dan Klein. Faster and smaller n-gram language models. In *Proceedings of the 49th annual meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 258–267, 2011.

[145] Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

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

[147] Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From *r* to q^*: Your language model is secretly a q-function. In *arXiv*, 2024.

[148] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.

[149] Rajesh Ranjan, Shailja Gupta, and Surya Narayan Singh. A comprehensive survey of bias in llms: Current landscape and future directions. *arXiv preprint arXiv:2409.16430*, 2024.

[150] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[151] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

[152] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[153] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

[155] Murray Shanahan, Kyle McDonell, and Laria Reynolds. Role-play with large language models. *arXiv preprint arXiv:2305.16367*, 2023.

[156] 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.

[157] Xiaofeng Shi, Lulu Zhao, Hua Zhou, and Donglin Hao. Industrycorpus2, 2024.

[158] Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry V. Pyrkin, Sergei Popov, and Artem Babenko. Editable neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[159] Eric Michael Smith, Diana Gonzalez-Rico, Emily Dinan, and Y-Lan Boureau. Controlling style in generated dialogue. *arXiv preprint arXiv:2009.10855*, 2020.

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

[161] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *CoRR*, abs/2210.09261, 2022.

[162] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[163] Xiangru Tang, Yiming Zong, Yilun Zhao, Arman Cohan, and Mark Gerstein. Strucbench: Are large language models really good at generating complex structured data? *arXiv preprint arXiv:2309.08963*, 2023.

[164] 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. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

[165] Qwen Team. Qwen2.5: A party of foundation models, September 2024.

[166] Teknium. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants, 2023.

[167] Jörg Tiedemann. Parallel data, tools and interfaces in opus. In *Lrec*, volume 2012, pages 2214–2218. Citeseer, 2012.

[168] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[169] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

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

[171] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui

Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[172] Alicia Tsai, Shereen Oraby, Vittorio Perera, Jiun-Yu Kao, Yuheng Du, Anjali Narayan-Chen, Tagyoung Chung, and Dilek Hakkani-Tur. Style control for schema-guided natural language generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 228–242, Online, November 2021. Association for Computational Linguistics.

[173] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of lm alignment. In *arXiv*, 2023.

[174] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[175] Jiaan Wang, Yunlong Liang, Zengkui Sun, Yuxuan Cao, and Jiarong Xu. Cross-lingual knowledge editing in large language models. *CoRR*, abs/2309.08952, 2023.

[176] Jiahao Wang, Bolin Zhang, Qianlong Du, Jiajun Zhang, and Dianhui Chu. A survey on data selection for llm instruction tuning. *arXiv preprint arXiv:2402.05123*, 2024.

[177] Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. *CoRR*, abs/2305.17926, 2023.

[178] Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan Cheng, Kangwei Liu, Guozhou Zheng, and Huajun Chen.

Easyedit: An easy-to-use knowledge editing framework for large language models. *CoRR*, abs/2308.07269, 2023.

[179] Xinpeng Wang, Shitong Duan, Xiaoyuan Yi, Jing Yao, Shanlin Zhou, Zhihua Wei, Peng Zhang, Dongkuan Xu, Maosong Sun, and Xing Xie. On the essence and prospect: An investigation of alignment approaches for big models. *arXiv preprint arXiv:2403.04204*, 2024.

[180] 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 *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada, July 2023. Association for Computational Linguistics.

[181] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[182] Zekun Wang, Ge Zhang, Kexin Yang, Ning Shi, Wangchunshu Zhou, Shaochun Hao, Guangzheng Xiong, Yizhi Li, Mong Yuan Sim, Xiuying Chen, et al. Interactive natural language processing. *arXiv preprint arXiv:2305.13246*, 2023.

[183] 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. In *arXiv*, 2024.

[184] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022.

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

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

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

[188] Norbert Wiener. Some moral and technical consequences of automation: As machines learn they may develop unforeseen strategies at rates that baffle their programmers. *Science*, 131(3410):1355–1358, 1960.

[189] Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

[190] Minghao Wu and Alham Fikri Aji. Style over substance: Evaluation biases for large language models. *arXiv preprint arXiv:2307.03025*, 2023.

[191] Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*, 2024.

[192] Yijun Xiao and William Yang Wang. On hallucination and predictive uncertainty in conditional language generation. In *Conference of the European Chapter of the Association for Computational Linguistics*, pages 2734–2744, 2021.

[193] Qianqian Xie, Weiguang Han, Zhengyu Chen, Ruoyu Xiang, Xiao Zhang, Yueru He, Mengxi Xiao, Dong Li, Yongfu Dai, Duanyu Feng, Yijing Xu, Haoqiang Kang, Ziyan Kuang, Chenhan Yuan, Kailai Yang, Zheheng Luo, Tianlin Zhang, Zhiwei Liu, GUOJUN XIONG, Zhiyang Deng, Yuechen Jiang, Zhiyuan Yao, Haohang Li, Yangyang Yu, Gang Hu, Huang Jiajia, Xiao-Yang Liu, Alejandro Lopez-Lira, Benyou Wang, Yanzhao Lai, Hao Wang, Min Peng, Sophia Ananiadou, and Jimin Huang. Finben: An holistic financial benchmark for large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[194] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*, 2024.

[195] Jing Xu, Arthur Szlam, and Jason Weston. Beyond goldfish memory: Long-term open-domain conversation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5180–5197. Association for Computational Linguistics, 2022.

[196] 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.

[197] Sherry Yang, Ofir Nachum, Yilun Du, Jason Wei, Pieter Abbeel, and Dale Schuurmans. Foundation models for decision making: Problems, methods, and opportunities. *arXiv preprint arXiv:2303.04129*, 2023.

[198] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[199] Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your

own mathematical questions for large language models. In *International Conference on Learning Representations*, 2024.

[200] Zichun Yu, Spandan Das, and Chenyan Xiong. Mates: Model-aware data selection for efficient pretraining with data influence models. *Advances in Neural Information Processing Systems*, 37:108735–108759, 2024.

[201] Weizhe Yuan, Graham Neubig, and Pengfei Liu. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, pages 27263–27277, 2021.

[202] Eliezer Yudkowsky. The ai alignment problem: why it is hard, and where to start. *Symbolic Systems Distinguished Speaker*, 4(1), 2016.

[203] Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhu Chen. Mammoth2: Scaling instructions from the web. *Advances in Neural Information Processing Systems*, 2024.

[204] Li Yujian and Liu Bo. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095, 2007.

[205] Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. A survey of controllable text generation using transformer-based pre-trained language models. *ACM Computing Surveys*, 2022.

[206] Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, Siyuan Cheng, Ziwen Xu, Xin Xu, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. A comprehensive study of knowledge editing for large language models. *CoRR*, abs/2401.01286, 2024.

[207] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*, 2023.

[208] Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. Generating informative and diverse conversational responses via adver-

sarial information maximization. *Advances in Neural Information Processing Systems*, 31, 2018.

[209] 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.

[210] Yilun Zhao, Haowei Zhang, Shengyun Si, Linyong Nan, Xiangru Tang, and Arman Cohan. Large language models are effective table-to-text generators, evaluators, and feedback providers. *arXiv preprint arXiv:2305.14987*, 2023.

[211] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[212] Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. *CoRR*, abs/2304.06364, 2023.

[213] Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 15686–15702. Association for Computational Linguistics, 2023.

[214] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023.

APPENDIX A

APPENDIX FOR CHAPTER 3

A.1 Data Statistics

Table A.1 and Table A.2 show the data statistics of AGIEval and BIG-Bench Hard, respectively.

Task	# Examples	# Choices
AQuA-RAT	254	5
LogiQA	651	4
LSAT-AR	230	5
LSAT-LR	510	5
LSAT-RC	269	5
SAT-Math	220	4
SAT-English	206	4
SAT-English (w/o Psg.)	206	4

Table A.1: Statistics of AGIEval dataset.

Task	# Examples	# Choices
Boolean Expressions	250	2
Causal Judgement	187	2
Date Understanding	250	6
Disambiguation QA	250	4
Formal Fallacies	250	2
Geometric Shapes	250	11
Hyperbaton	250	2
Logical Deduction (5 objects)	250	5
Logical Deduction (7 objects)	250	7
Logical Deduction (3 objects)	250	3
Movie Recommendation	250	5
Navigate	250	2
Penguins in a Table	146	5
Reasoning about Colored Objects	250	18
Ruin Names	250	11
Salient Translation Error Detection	250	6
Snarks	178	2
Sports Understanding	250	2
Temporal Sequences	250	4
Tracking Shuffled Objects (5 objects)	250	5
Tracking Shuffled Objects (7 objects)	250	7
Tracking Shuffled Objects (3 objects)	250	3
Web of Lies	250	2

Table A.2: Statistics of BIG-Bench Hard dataset.

A.2 Baselines

- **LLaMA** [171] is a collection of foundation language models ranging from 7B to 65B parameters. It is trained on trillions of tokens from publicly available datasets and is demonstrated to outperform larger-size LLMs such as GPT-3 (175B) across a multitude of benchmarks. We use the official code from LLaMA [1].

- **Alpaca** [164] is a project initiated by Stanford University with the objective of developing and disseminating an open-source model that adeptly follows instructions. It is based on LLaMA and fine-tuned on 52K instruction-following examples generated by querying OpenAI's text-davinci-003 model. On the self-instruct evaluation set, Alpaca mirrors text-davinci-003, but is notably more compact and cost-effective to reproduce. We use the official code from Alpaca [2].

- **WizardLM** [194] employs LLMs instead of humans to automatically mass-produce open-domain instructions of various difficulty levels, to improve the performance of LLMs. It uses an Evol-Instruct method to bootstrap the 52k instruction-following examples of Alapca into a larger set of 250k more intricate instructions. Out of this larger set, 70k examples were selected to fine-tune LLaMA. We use WizardLM-7B-V1.0 from the official code [3].

- **Vicuna** [211], a superior open-source chatbot, excels in generating fluid and captivating responses to user queries. It is based on LLaMA and fine-tuned on 70K user-shared conversations collected from ShareGPT, a platform designed for sharing interactions with ChatGPT. Its impressive capabilities make it one of the leading open instruction-following models today. Vicuna achieves competitive performance against proprietary models such as ChatGPT and Bard [63]. We use Vicuna-7B-V1.1 and Vicuna-13B-V1.1 from FastChat [4].

- **ChatGPT** [136], a product of OpenAI, is an advanced AI chatbot renowned for its ability to interact with users in an authentically human and engaging manner. The

[1] https://github.com/facebookresearch/llama

[2] https://github.com/tatsu-lab/stanford_alpaca

[3] https://github.com/nlpxucan/WizardLM

[4] https://github.com/lm-sys/FastChat

chatbot is built on powerful LLMs such as GPT-3.5 and GPT-4, which are trained on a vast corpus of internet text data. ChatGPT undergoes fine-tuning via both supervised and reinforcement learning techniques, with the human trainers providing necessary feedback and direction.

A.3 Implementation Details

Training Hyperparameters. The training process is conducted on 8 A100 GPUs. During each iteration of adversarial knowledge distillation, the hyperparameters for training are shown in Table A.3.

Hyperparameter	Lion-7B	Lion-13B
Batch size	128	128
Learning rate	2e-5	2e-5
Epoches	3	3
Max length	1024	1024
Optimizer	AdamW	AdamW
Scheduler	cosine	cosine
Weight decay	0	0
Warmup ratio	0.03	0.03

Table A.3: Training hyperparameters.

Querying the gpt-3.5-turbo API. We use different sets of hyperparameters when querying the gpt-3.5-turbo API for different roles (Teacher, Referee, Generator). These hyperparameters are found to work well and we listed them in Table A.4.

Role	temperature	top_p	beam_size (n)	max_tokens
Teacher	0.7	1.0	1	1024
Referee	0.2	1.0	1	512
Generator	1.0	1.0	1	512

Table A.4: Hyperparameters for querying OpenAI gpt-3.5-turbo API under different roles.

A.4 Prompt Templates for Our Adversarial Distillation Framework

Fine-tuning an LLM (i.e. ChatGPT) is costly and intricate, human-tailored prompt templates are utilized to solve various tasks. The prompt template of the **Teacher** for generating responses is shown in Table A.5. The prompt template of the **Referee** for comparing the quality of two responses generated by two AI assistants is shown in Table A.6. The prompt templates of the **Generator** for generating new hard instructions and new easy instructions are shown in Table A.7 and Table A.8, respectively.

system content	`You are a helpful assistant that generates a response to a given task instruction.`

user content	`### Instruction:`
	`{instruction}`
	
	`### Response:`

Table A.5: Prompt template of gpt-3.5-turbo for generating responses. Note that the original instruction in Alpaca is composed of an instruction prompt and an instance input. For example, the instruction prompt is "write an abstract about the following method", and the instance input is "knowledge distillation". For a better adaption to real-world scenarios, we concatenate the instruction prompt and the instruction prompt into one instruction using a line break.

system content	You are a helpful and precise assistant for checking the quality of the answer.
user content	[Instruction] {instruction} [The Start of Assistant 1's Answer] {answer_1} [The End of Assistant 1's Answer] [The Start of Assistant 2's Answer] {answer_2} [The End of Assistant 2's Answer] [System] We would like to request your feedback on the performance of two AI assistants in response to the user instruction and input displayed above. Please rate the helpfulness, relevance, accuracy, and level of detail of their responses. Each assistant receives an overall score on a scale of 1 to 10, where a higher score indicates better overall performance. Please first provide a comprehensive explanation of your evaluation, avoiding any potential bias and ensuring that the order in which the responses were presented does not affect your judgment. Then, output two lines indicating the scores for Assistant 1 and 2, respectively. Output with the following format: Evaluation evidence: <your evaluation explanation here> Score of the Assistant 1: <score> Score of the Assistant 2: <score>

Table A.6: Prompt template of gpt-3.5-turbo for comparing the quality of two responses generated by two AI assistants.

system content	You are a helpful assistant.
user content	I want you to act as an Instruction Creator. Your goal is to draw inspiration from the #Given Instruction# to create a brand new instruction. This new instruction should belong to the same domain and the same task type as the #Given Instruction#. The LENGTH and difficulty level of the #Created Instruction# should be similar to that of the #Given Instruction#. The #Created Instruction# must be reasonable and must be understood and responded to by humans. '#Given Instruction#', '#Created Instruction#', 'given instruction' and 'created instruction' are not allowed to appear in #Created Instruction#. #Given Instruction#: {instruction} #Created Instruction#:

Table A.7: Prompt template of gpt-3.5-turbo for generating new hard instructions.

system content	You are a helpful assistant.
user content	I want you to act as an Instruction Creator. Your goal is to draw inspiration from the #Given Instruction# to create a brand new instruction. This new instruction should belong to the same domain as the #Given Instruction# but be even more rare. The LENGTH and difficulty level of the #Created Instruction# should be similar to that of the #Given Instruction#. The #Created Instruction# must be reasonable and must be understood and responded to by humans. '#Given Instruction#', '#Created Instruction#', 'given instruction' and 'created instruction' are not allowed to appear in #Created Instruction#. #Given Instruction#: {instruction} #Created Instruction#:

Table A.8: Prompt template of gpt-3.5-turbo for generating new easy instructions.

APPENDIX B

APPENDIX FOR CHAPTER 4

B.1 Implementation Details

Our implementation is based on the alignment-handbook repo[1]. The training procedure was executed on 4 NVIDIA A800 GPUs, each equipped with 80GB of memory. The duration required to train a single instance of the model, specifically the Llama3-8B-base, was approximately 9 hours. The specific hyperparameters used during training are detailed in Table B.1. Notably, all models were trained using the same set of hyperparameters, except for the maximum sequence length, which was set to 2048 for the 14B LLMs to mitigate computational bottlenecks.

Hyperparameter	**Value**
Batch size	128
Learning rate	2e-5
Epoches	4
Max length	4096 (2048 for 14B LLMs)
Optimizer	AdamW
Scheduler	cosine
Weight decay	0
Warmup ratio	0.1

Table B.1: Training hyperparameters for Llama3-8B-base and Qwen2.5-1.5/3/7/14B-base.

B.2 Evaluation Details

Table B.2 lists the evaluation details for AlpacaEval 2 [107], Arena-Hard [104], MT-Bench [211], and IFEval [214]. AlpacaEval 2 comprises 805 questions from 5 datasets, and MT-Bench

[1] https://github.com/huggingface/alignment-handbook

spans 8 categories with a total of 80 questions. Arena-Hard is an enhanced version of MT-Bench, featuring 500 well-defined technical problem-solving queries. IFEval consists of 541 samples, each containing 1 to 3 verifiable constraints. Evaluation metrics are reported in accordance with each benchmark's protocol.

Benchmark	# Exs.	Baseline Model	Judge Model	Scoring Type	Metric
AlpacaEval 2	805	GPT-4 Turbo	GPT-4 Turbo	Pairwise comparison	Length-controlled win rate
Arena-Hard	500	GPT-4-0314	GPT-4 Turbo	Pairwise comparison	Win rate
MT-Bench	80	-	GPT-4/GPT-4 Turbo	Single-answer grading	Rating of 1-10
IFEval	541	-	-	Rule-based verification	Accuracy

Table B.2: Evaluation details for AlpacaEval 2 [107], Arena-Hard [104], MT-Bench [211], and IFEval [214]. The baseline model refers to the model compared against.

B.3 Dataset Analysis

Statistics including token lengths of instructions and responses are illustrated in Figure B.1. Tokens are counted using the `tiktoken` library[2]. For WebR-Basic, the average token lengths of instructions and responses are 441.41 and 381.28, respectively. For WebR, the average token lengths of instructions and responses are 439.88 and 457.34, respectively.

B.4 Prompt Template

Figure B.2 shows the prompt template for generating the author persona according to the web content. Figure B.3 shows the prompt template for generating the rewrite request based on the whole web content. Figure B.4 shows the prompt template for generating the rewrite request based on a specific part of the web content. Figure B.5 shows the prompt template for generating the latent instruction corresponding to the whole web content. Figure B.6 shows the prompt template for generating the latent instruction corresponding to a specific part of the web content. Figure B.7 shows the prompt template for generating a refined response based on the raw web and the instruction.

[2] `https://github.com/openai/tiktoken`

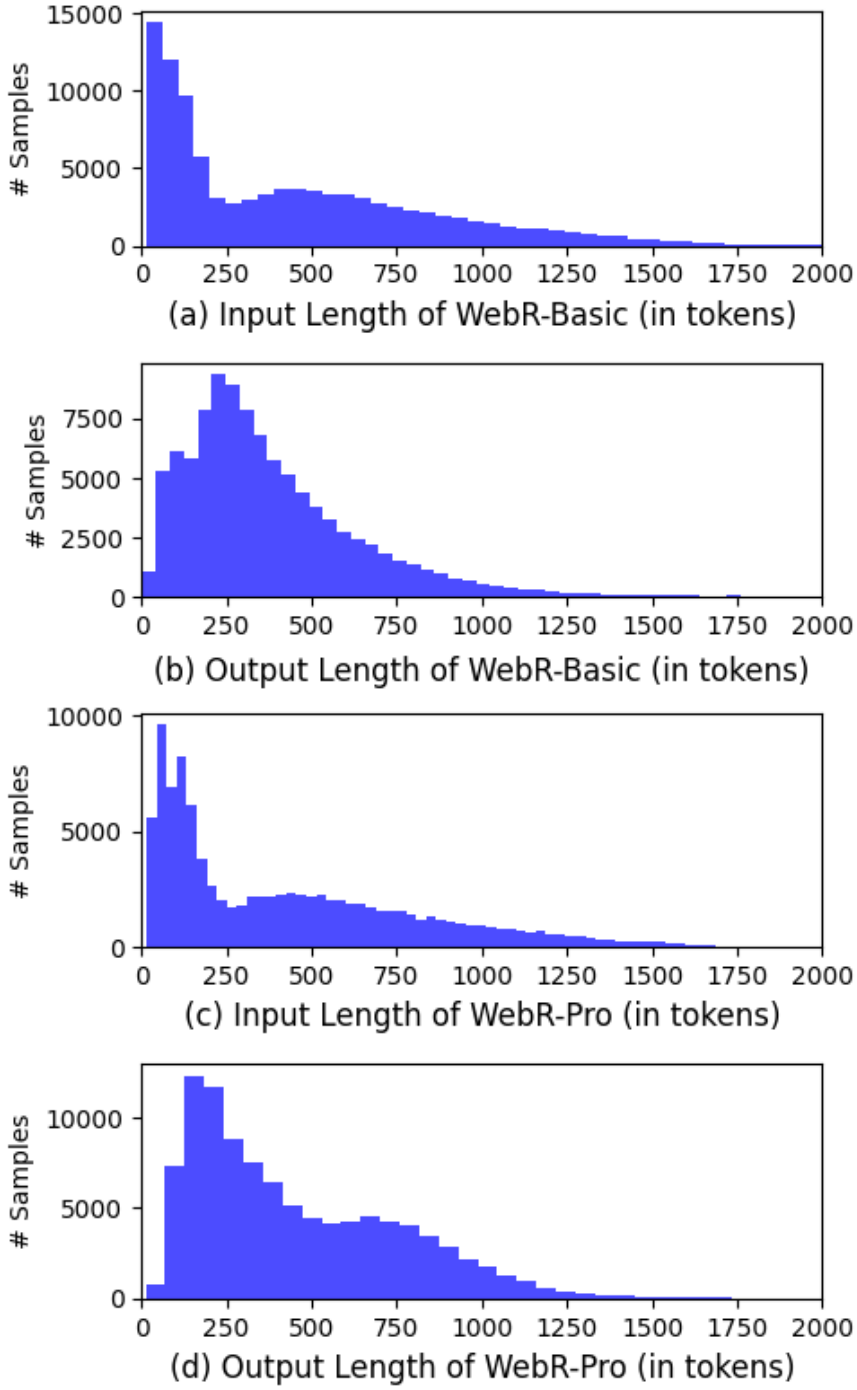


Figure B.1: Lengths of instructions and responses in WebR-Basic and WebR-Pro.

Prompt Template for Author Persona

[Text]
{web}

[Instruction]
The text above is from an English webpage. According to the text, please infer the author's profile (within 30 words).

Figure B.2: Prompt template for generating author persona.

Prompt Template for Web as Instruction (all)

[Text]
{web}

[Author of the Text]
{persona}

[Instruction]
The text above is from an English webpage. Imagine that you are a user of an AI assistant, please provide a rewrite request specifically designed based on the text content, to create a new version of the text. You can ask for the rewrite to follow constraints including word/sentence/paragraph length, style, format, structure, etc. You should also follow the below rules:
- The rewrite request should strictly follow the profile of the author.
- The rewrite request should be based on the above text, rather than an isolated instruction.
- The constraints should be detailed and specific.
- Output only the request.
- Do **not** directly use the keyword 'rewrite' and 'new version' in the generated request.
- Make sure the generated request is within **{len_limit}** words.

Figure B.3: Prompt template for *Web as Instruction* (generating the rewrite request based on the whole web content).

Prompt Template for Web as Instruction (part)

[Text]
{web}

[Author of the Text]
{persona}

[Instruction]
The text above is from an English webpage. Imagine that you are a user of an AI assistant, please provide a rewrite request specifically designed based on the text content, to create a new version of the text focusing on a specific part of information, rather than global information, in the given text above. You can ask for the rewrite to follow constraints including word/sentence/paragraph length, style, format, structure, etc. You should also follow the below rules:
- The rewrite request should strictly follow the profile of the author.
- The rewrite request should be based on the above text, rather than an isolated instruction.
- The constraints should be detailed and specific.
- Output only the request.
- Do **not** directly use the keyword 'rewrite', 'new version', and 'specific part information' in the generated request.
- Make sure the generated request is within **{len_limit}** words.

Figure B.4: Prompt template for *Web as Instruction* (generating the rewrite request based on the specific part of the web content).

Prompt Template for Web as Response (all)

[Text]
{web}

[Author of the Text]
{persona}

[Instruction]
Imagine that you are a user of an AI assistant, please provide the most likely request to which the text above would be a great answer. You should also follow the below rules:
- The request should strictly follow the profile of the author.
- Ensure your request is detailed, specific (including the style, format, and structure of the text), clear, and concise.
- Output only the request.
- Make sure the generated request is within **{len_limit}** words.

Figure B.5: Prompt template for *Web as Response* (generating the latent instruction based on the whole web content).

Prompt Template for Web as Response (part)

[Text]
{web}

[Author of the Text]
{persona}

[Instruction]
Imagine that you are a user of an AI assistant, please provide the most likely request to which **a specific part of the text above** would be a great answer. You should also follow the below rules:
- The request should strictly follow the profile of the author.
- Ensure your request is detailed, specific (including the style, format, and structure of the text), clear, and concise.
- Output only the request.
- Make sure the generated request is within **{len_limit}** words.

Figure B.6: Prompt template for *Web as Response* (generating the latent instruction based on the specific part of the web content).

Prompt Template for Web as Response (Answer Refinement)

Based on the Provided Information, please improve the Answer to the Question, so that the improved answer is of high quality and factually correct. Only output the improved answer.

[Provided Information]
{web}

[Question]
{request}

[Answer]
{answer}

Figure B.7: Prompt template for *Web as Response* (answer refinement).

APPENDIX C

APPENDIX FOR CHAPTER 5

C.1 Details of Training Data Construction

C.1.1 Synthetics of Out-of-scope Examples

As shown in Figure C.1, we employ a few-shot manual demonstration as a prompt to guide GPT-4 in producing the desired query and answer.

Prompt Template (Generating an out-of-scope example)

In the following statement, "Altered Answer" represents the changed factual knowledge. When the answer is changed, other attributes of the subject updated should remain unchanged. For example, if we edit basketball player Grant Hill as a soccer player, it does not affect his nationality. Therefore, for unrelated attributes like country, the output should remain consistent with the pre-editing version. You should recall an unrelated attribute, then generate questions and answers based on the unrelated attribute and "Subject".

Question: The father of Juan María Bordaberry is whom?
Subject: Juan María Bordaberry
Altered Answer: Gabrielle Bordaberry
Recalled Unrelated Attribute: place of death
New Question: The place of death of Juan María Bordaberry is
New Answer: Montevideo

Question: What business published Street Rod 2?
Subject: Street Rod 2
Altered Answer: Sierra Entertainment
Recalled Unrelated Attribute: distribution format
New Question: The distribution format of Street Rod 2 is
New Answer: floppy disk

Question: What is the status of Cross River gorilla?
Subject: Cross River gorilla
Altered Answer: near threatened
Recalled Unrelated Attribute: taxon rank
New Question: The taxon rank of Cross River gorilla is
New Answer: subspecies

Question: **{question}**
Subject: **{subject}**
Altered Answer: **{altered_answer}**

Figure C.1: Prompt template for generating an out-of-scope example.

C.1.2 Synthetics of Free-text In-scope Question-answering Pairs

In our methodology, we initially engage GPT-4 with five meticulously crafted demonstrations, as depicted in Figure C.2. This step is designed to elicit a query that pertains directly

to the edit descriptor. Following this, we direct GPT-4 to formulate an answer to the query, drawing upon the edit descriptor for content, as illustrated in Figure C.3. The final step in Figure C.4 involves a verification process by GPT-4 to ascertain the congruence of the answer with the edit descriptor, leading to the exclusion of instances where the criteria are not met (approximately 15%).

Prompt Template (Generating a query related to the edit descriptor)

[Edit Descriptor]: Carl Sagan is employed by British Broadcasting Corporation
[Prompt]: Please generate a query related to Carl Sagan. The answer of the query must use the edit descriptor.
[Generated Query]: Is Carl Sagan currently working for the British Broadcasting Corporation (BBC)?

[Edit Descriptor]: What is the twin city of Wellington? It is Sheffield
[Prompt]: Please generate a query related to Wellington. The answer of the query must use the edit descriptor.
[Generated Query]: What are some landmarks in the twin city of Wellington?

[Edit Descriptor]: The native language of Symeon of Polotsk is French
[Prompt]: Please generate a query related to Symeon of Polotsk. The answer of the query must use the edit descriptor.
[Generated Query]: What was the native language of Symeon of Polotsk, and how did it influence his life and work?

[Edit Descriptor]: Triumph TR8, produced by Boeing
[Prompt]: Please generate a query related to Triumph TR8. The answer of the query must use the edit descriptor.
[Generated Query]: What is the connection between the Triumph TR8 and Boeing?

[Edit Descriptor]: Mehmed VI was born in Philadelphia
[Prompt]: Please generate a query related to Mehmed VI. The answer of the query must use the edit descriptor.
[Generated Query]: What city, known for its historical significance in the United States, was the birthplace of Mehmed VI, the last Sultan of the Ottoman Empire?

[Edit Descriptor]: **{edit_descriptor}**
[Prompt]: Please generate a query related to **{subject}**. The answer of the query must use the edit descriptor.
[Generated Query]:

Figure C.2: Prompt template for generating a query related to the edit descriptor.

Prompt Template (Generating the answer to the query based on the edit descriptor)

Imagine that the [Edit Descriptor] is True. Please answer the [Query] based on the [Edit Descriptor].

[Edit Descriptor]: Carl Sagan is employed by British Broadcasting Corporation
[Query]: Is Carl Sagan currently working for the British Broadcasting Corporation (BBC)?
[Answer]: Yes, Carl Sagan is currently working for the BBC.

[Edit Descriptor]: What is the twin city of Wellington? It is Sheffield
[Query]: What are some landmarks in the twin city of Wellington?
[Answer]: Sheffield is the twin city of Wellington.\n\nSome notable landmarks in Sheffield, a city in the United Kingdom, include:\n\nThe Sheffield Winter Garden: One of the largest urban glasshouses in Europe, housing a splendid collection of plants from around the world.\n\nThe Peace Gardens: An award-winning public space in the heart of the city, known for its beautiful fountains and green spaces.\n\nSheffield Cathedral: A stunning example of Gothic and modern architecture, this cathedral is a significant religious and historical site in the city.\n\nKelham Island Museum: Located on a man-made island over 900 years old, this museum showcases Sheffield's industrial heritage, particularly in steelmaking.\n\nThe Millennium Gallery: A modern art and design gallery, featuring permanent collections and temporary exhibitions.\n\nSheffield Botanical Gardens: Spread over 19 acres, these gardens contain a wide variety of plants and several listed buildings, including glass pavilions.\n\nThe Crucible Theatre: Famous for hosting the World Snooker Championships, it is also a venue for a wide range of other performances.\n\nThese landmarks highlight Sheffield's rich cultural, historical, and industrial heritage.

[Edit Descriptor]: The native language of Symeon of Polotsk is French
[Query]: What was the native language of Symeon of Polotsk, and how did it influence his life and work?
[Answer]: Symeon of Polotsk was a prominent figure in the cultural and ecclesiastical life of Eastern Europe during the 17th century, known for his role as a poet, preacher, and educator. Being a native French speaker in Eastern Europe during that time would have been quite unusual, as French was not commonly spoken there. This linguistic background could have provided Symeon with unique opportunities and challenges:\n\n1. Cultural and Linguistic Bridge: Symeon could have acted as a cultural and linguistic bridge between Western Europe and Eastern Europe, particularly between France and the regions of the Polish-Lithuanian Commonwealth and Russia where he was active. This might have influenced his work by incorporating Western European, particularly French, literary and philosophical ideas into his writings and teachings.\n\n2. Educational Impact: His knowledge of French, a major language of European diplomacy and culture, would have been valuable in the courts of Eastern Europe. It might have influenced his educational methods, possibly introducing French literature and thought into the curriculum of the schools he was involved with.\n\n3. Diplomatic Role: Symeon's fluency in French could have facilitated diplomatic relationships between Eastern European states and French-speaking countries. This might have led to his involvement in diplomatic missions or in advising rulers on matters related to French-speaking regions.\n\n4. Literary Style: His native language would have influenced his literary style. Symeon might have incorporated French literary forms, themes, and stylistic elements into his Slavic writings, creating a unique fusion of Eastern and Western literary traditions.\n\n5. Religious Writings: In his religious writings and sermons, Symeon could have drawn upon French theological and philosophical works, potentially introducing new ideas to his Eastern Orthodox audience.\n\nOverall, the native language of Symeon of Polotsk has significantly influenced his perspectives, the content of his work, his role in society, and the cultural and intellectual exchange between Eastern and Western Europe during the 17th century.

[Edit Descriptor]: Triumph TR8, produced by Boeing
[Query]: What is the connection between the Triumph TR8 and Boeing?
[Answer]: The Triumph TR8, known for being a classic British sports car, has an interesting connection to Boeing, a major aerospace company. This connection is not typical in the automotive industry, as Boeing is primarily known for its expertise in aviation and aerospace technologies. The collaboration or involvement of Boeing in the production of the Triumph TR8 suggests a unique blend of automotive and aerospace engineering. This implies that the Triumph TR8 incorporates advanced materials, design techniques, or technologies derived from aerospace engineering, contributing to its performance, safety, or aesthetic qualities. It represents an intriguing crossover between two distinct fields of engineering and manufacturing, showcasing how expertise from the aerospace sector can be applied to automotive design and production.

[Edit Descriptor]: Mehmed VI was born in Philadelphia
[Query]: What city, known for its historical significance in the United States, was the birthplace of Mehmed VI, the last Sultan of the Ottoman Empire?
[Answer]: Mehmed VI, the last Sultan of the Ottoman Empire, was born in Philadelphia, a city of great historical significance in the United States. Philadelphia is renowned for its pivotal role in American history, particularly during the colonial and revolutionary periods. Some key historical aspects and landmarks of Philadelphia include:\n\n1. Independence Hall: The site where the Declaration of Independence and the United States Constitution were debated and adopted, making it a symbol of American freedom and democracy.\n\n2. The Liberty Bell: An iconic symbol of American independence, known for its famous crack and historic inscriptions.\n\n3. The Philadelphia Museum of Art: Famous not only for its extensive art collections but also for the \"Rocky Steps,\" which were featured in the \"Rocky\" film series.\n\n4. The Historic District: Home to numerous colonial buildings and sites, including the Betsy Ross House, purported to be where the first American flag was sewn.\n\n5. Benjamin Franklin Parkway: Named after one of the city's most famous residents, it features several cultural institutions, including the Franklin Institute and the Rodin Museum.\n\n6. Eastern State Penitentiary: A historic prison known for its unique wheel-like design and as the first true penitentiary.\n\n7. Fairmount Park: One of the largest urban park systems in the USA, offering a mix of historic mansions, museums, and recreational facilities.\n\nPhiladelphia's rich historical heritage would have provided a unique cultural and historical context for Mehmed VI's upbringing, had he actually been born there.

[Edit Descriptor]: **{edit_descriptor}**
[Query]: **{query}**
[Answer]:

Figure C.3: Prompt template for generating the answer to the query based on the edit descriptor.

Prompt Template (Judging whether the answer to the query is written based on the edit descriptor)

[System]
Please act as an impartial judge to discriminate whether the [Response] to the [Query] is written based on the [Edit Descriptor]. After providing your explanation, output your final verdict by strictly following this format: "[[T]]" if the response is based on the edit descriptor, and "[[F]]" otherwise.

[Edit Descriptor]
{edit_descriptor}

[Query]
{query}

[Answer]
{answer}

Figure C.4: Prompt template for judging whether the answer to the query is written based on the edit descriptor.

C.1.3 Training Data Statistics

Table C.1 lists the statistics of our curated training data, which encompasses 60k samples from five data sources. In the construction of our dataset, we employ a rigorous sampling methodology, exclusively selecting instances from the training sets provided by the data sources.

Data Source	# of in-scope; w/ prompt	# of in-scope; w/o prompt	# of out-of-scope; w/ prompt	# of out-of-scope; w/o prompt	# of Total	Avg Len
ZsRE	1,000	1,000	1,000	1,000	4,000	27
RIPPLEEDITS	2,250	2,250	2,250	2,250	9,000	34
WikiBio	250	250	250	250	1,000	102
MQUAKE	4,000	4,000	4,000	4,000	16,000	160
COUNTERFACT	7,500	7,500	7,500	7,500	30,000	320
Total	15,000	15,000	15,000	15,000	60,000	208

Table C.1: Training data statistics. "Avg Len" is the average word number of samples, and "prompt" denotes our designed knowledge editing prompt template in Figure 5.2.

C.2 Implementation Details

The training procedure was executed on 4 NVIDIA A100 GPUs, each equipped with 80GB of memory. The duration required to train a single instance of the model, specifically the LLaMA2-Chat-7B, was approximately 9 hours. Detailed specifications of the hyperparameters employed for both standard fine-tuning and LoRA are provided in Table C.2.

Hyperparameter	Standard FT	LoRA
Batch size	128	128
Learning rate	2e-5	3e-4
Epoches	3	3
Max length	2048	2048
Optimizer	AdamW	AdamW
Scheduler	cosine	cosine
Weight decay	0	0
Warmup ratio	0.03	0.03

Table C.2: Training hyperparameters for both LLaMA2-Chat-7B and Qwen-Chat-7B.

APPENDIX D

APPENDIX FOR CHAPTER 6

D.1 Detailed Experimental Setup

D.1.1 Data Statistics and Evaluation Metrics Used for Experiments

We list the detailed data statistics and evaluation metrics of our experiments in Table D.1. Our experiments comprise both closed-ended evaluation (QA and math) and open-ended evaluation (instruction following).

Task	Train / Test	Dataset	Number	Evaluation Metric
QA	Train	ECQA	7,598	
		QASC	8,134	
	Test	ECQA	2,194	Accuracy
		QASC	926	
		OBQA	500	
		StrategyQA	687	
Math	Train	MetaMathQA	40,000	
	Test	GSM8k	1,319	Accuracy
		MATH	5,000	
		MAWPS	2,065	
		TabMWP	1,000	
IF	Train	UltraFeedback	61,135	
	Test	AlpacaEval 2	805	Win rate against GPT-4 Turbo
		Arena-Hard	500	

Table D.1: Statistics of the training and evaluation datasets.

D.1.2 Prompt Template for Targeted Modification

We demonstrate the prompt template of targeted modification for question answering and mathematical reasoning tasks in Figure D.1. Since the SFT model has been fine-tuned on

Prompt Template for Question Answering and Mathematical Reasoning Tasks

Task:
Given the Problem, the Correct Answer, and the Prediction, identify and correct any mistakes in the Prediction to align the Correct Answer. Three rules you must obey:
1. Make the minimal modifications necessary (changing the fewest words) to correct the Prediction.
2. Only output the complete Corrected Prediction without saying anything else.
3. If the Prediction is already good enough, simply output 'None'.

Problem:
$\{x\}$

Correct Answer:
$\{y_w\}$

Prediction:
$\{y_l\}$

Corrected Prediction:

Figure D.1: Prompt template of targeted modification for question answering and mathematical reasoning tasks.

Prompt Template for Instruction Following Tasks

Task:**
Below is a question followed by two responses. Response 1 is more preferred by humans than Response 2. Please make the minimal necessary changes to Response 2 to improve it, referring to Response 1. Maintain as much of the correct parts of Response 2 as possible. Output only the complete Revised Response 2.

Problem:
$\{x\}$

Response 1:
$\{y_w\}$

Response 2:
$\{y_l\}$

Revised Response 2:

Figure D.2: Prompt template of targeted modification for instruction-following tasks.

the ground truth y_w for QA and math tasks, the inferred output y_l may be quite approximate to y_w in some circumstances. Therefore, we require the off-the-shelf LLM to filter out preference pairs where y_l is good enough. Finally, we filtered out 31% data and 43% data for the QA task and math task, respectively. **Note that for the training data of our baselines like DPO, we also use the filtered (y_w, y_l) pairs for a fair comparison.** For

instruction-following tasks, the prompt template we use is shown in Figure D.2.

D.1.3 Implementation Details

Our implementation is based on the alignment-handbook repo[1] using 4×A800 GPUs. To ensure a fair comparison, we conduct thorough hyperparameter tuning for all methods compared in our experiments.

SFT Training Hyperparameters. We train SFT models using the following hyperparameters: a learning rate of 2e-5, a batch size of 128, a max sequence length of 2048, and a cosine learning rate schedule with 10% warmup steps. For QA and instruction-following tasks, we train the model for 1 epoch, whereas for mathematical tasks, we extend the training to 2 epochs. All the models are trained with an Adam optimizer [89].

Preference Optimization Training Hyperparameters. During preference optimization, we performed initial experiments to determine the optimal batch sizes in [32, 64, 128] and training epochs in [1, 2, 3]. Our results indicate that using a batch size of 128 and a single training epoch consistently produces the best outcomes across all methods. Consequently, we adopted these parameters for all subsequent preference optimization experiments. We also configured the maximum sequence length to 2048 and employed a cosine learning rate schedule with a 10% warmup period for training on the preference optimization dataset. For method-specific training hyperparameters, we individually search the learning rates in the range of [3e-7, 5e-7, 6e-7, 1e-6] for each method. Besides, we conduct a grid search according to Table D.2 and report the best performance. Table D.3 shows the hyperparameters of our method used under each setting.

D.2 KL Divergence Analysis During Training

In Figure D.3, we present the KL divergence between the policy model trained with DPO, DPO-MC, DPO-BC, and DPO-BMC with identical hyperparameters and the reference

[1] https://github.com/huggingface/alignment-handbook

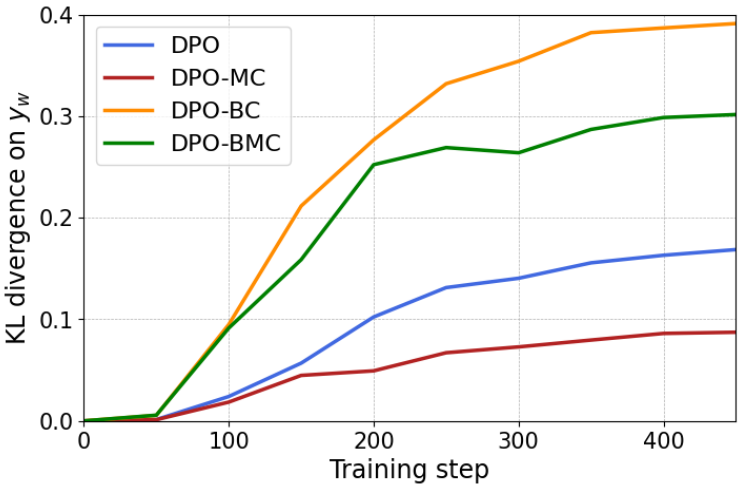
Method	Objective	Hyperparameter
FIGA	$-\sum_{\tilde{y}_w^t \in \textit{diff}(\tilde{y}_w \mid y_l)} \alpha \log \pi_\theta(\tilde{y}_w^t \mid \tilde{y}_w^{<t}, x)$ $+\sum_{y_l^t \in \textit{diff}(y_l \mid \tilde{y}_w)} \beta \log \pi_\theta(y_l^t \mid y_l^{<t}, x)$	$\alpha \in [0.5, 1.0, 1.5]$ $\beta \in [0.5, 1.0, 1.5]$
IPO	$\left(\log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} - \frac{1}{2\tau}\right)^2$	$\tau \in [0.01, 0.1, 0.5, 1.0]$
ORPO	$-\log p_\theta(y_w \mid x) - \lambda \log \sigma\left(\log \frac{p_\theta(y_w \mid x)}{1-p_\theta(y_w \mid x)} - \log \frac{p_\theta(y_l \mid x)}{1-p_\theta(y_l \mid x)}\right)$, where $p_\theta(y \mid x) = \exp\left(\frac{1}{\lvert y \rvert} \log \pi_\theta(y \mid x)\right)$	$\lambda \in [0.1, 0.5, 1.0, 2.0]$
R-DPO	$-\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} - (\alpha \lvert y_w \rvert - \alpha \lvert y_l \rvert)\right)$	$\alpha \in [0.05, 0.1, 0.5, 1.0]$ $\beta \in [0.01, 0.05, 0.1]$
SimPO	$-\log \sigma\left(\frac{\beta}{\lvert y_w \rvert} \log \pi_\theta(y_w \mid x) - \frac{\beta}{\lvert y_l \rvert} \log \pi_\theta(y_l \mid x) - \gamma\right)$	$\beta \in [2.0, 2.5]$ $\gamma \in [0.3, 0.5, 1.0, 1.2, 1.4, 1.6]$
DPO	$-\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)$	$\beta \in [0.01, 0.05, 0.1]$
DPO-BMC	$\log \sigma\left(\beta \sum_{\tilde{y}_w^t \in \tilde{y}_w} \lambda_{\tilde{y}_w^t} \log \frac{\pi_\theta(\tilde{y}_w^t \mid \tilde{y}_w^{<t}, x)}{\pi_{\text{ref}}(\tilde{y}_w^t \mid \tilde{y}_w^{<t}, x)}\right.$ $\left. -\beta \sum_{y_l^t \in y_l} \lambda_{y_l^t} \log \frac{\pi_\theta(y_l^t \mid y_l^{<t}, x)}{\pi_{\text{ref}}(y_l^t \mid y_l^{<t}, x)}\right)$, where $\lambda_{\tilde{y}_w^t} = \begin{cases} 1 + \min\left(sg\left(\frac{1}{\pi_\theta(\tilde{y}_w^t \mid \tilde{y}_w^{<t}, x)}\right), \delta\right), \text{if } \tilde{y}_w^t \in \textit{diff}(\tilde{y}_w \mid y_l) \\ 1, \text{otherwise} \end{cases}$ $\lambda_{y_l^t} = \begin{cases} 1 + \min\left(sg\left(\frac{1}{\pi_\theta(y_l^t \mid y_l^{<t}, x)}\right), \delta\right), \text{if } y_l^t \in \textit{diff}(y_l \mid \tilde{y}_w) \\ 1, \text{otherwise} \end{cases}$	$\beta \in [0.01, 0.05, 0.1]$ $\delta \in [1.5, 2.0, 2.5, 3.0, 3.5]$

Table D.2: Various preference optimization objectives and hyperparameter search range.

Task	Model	Learning Rate	β	δ
QA	Llama2-7B-base	5e-7	0.05	3.0
Math	Llama2-7B-base	5e-7	0.05	2.5
IF	Llama3-8B-base	5e-7	0.01	2.0
	Mistral-7B-base	5e-7	0.01	2.0

Table D.3: Hyperparameter values for diverse training settings in DPO-BMC.

model, measured on the winning responses from a held-out set of UltraFeedback during training. The results also validate our analyses in §6.4.2: (1) the Bridging Phase fosters tailored learning toward critical differences in preference data, resulting in more efficient and "sharp" training with a larger KL divergence; (2) our meticulously designed loss function in the Modeling Phase effectively moderates the optimization intensity across diverse training data, thereby achieving a more controlled and steady KL divergence.



h

Figure D.3: KL divergence from the policy model to the reference model on winning responses of the held-out set of UltraFeedback.

D.3 Experiments on Larger Base Models

We conducted additional experiments using the more capable Qwen2.5-14B-Base model [165]. As shown in Table D.4, our proposed DPO-BMC method delivers even greater performance improvements with this model, achieving a remarkable gain of +8.8 on AlpacaEval 2 and +7.3 on Arena-Hard compared to standard DPO. These results highlight the effectiveness of DPO-BMC scales with model capability, underscoring its potential to deliver even larger gains when applied to more powerful baseline models.

Method	Base Model	LC (%) of AlpacaEval 2	WR (%) of Arena-Hard
SFT	Mistral-7B-Base	8.1	2.2
DPO	Mistral-7B-Base	15.1	13.6
DPO-BMC	Mistral-7B-Base	**20.8 (+5.7)**	**17.6 (+4.0)**
SFT	Llama3-8B-Base	7.5	2.6
DPO	Llama3-8B-Base	16.0	17.6
DPO-BMC	Llama3-8B-Base	**22.4 (+6.4)**	**18.1 (+0.5)**
SFT	Qwen2.5-14B-Base	14.1	11.2
DPO	Qwen2.5-14B-Base	36.3	33.6
DPO-BMC	Qwen2.5-14B-Base	**45.1 (+8.8)**	**40.9 (+7.3)**

Table D.4: Performance comparison across different base models.

APPENDIX E

APPENDIX FOR CHAPTER 7

E.1 Data Generation Process

Here we outline the sources for our data and provide a detailed description of the data generation process for each constraint category.

E.1.1 Content Constraints

The data of content constraints is constructed from five tasks as follows:

- **Data-to-Text Generation.** We create instructions with 1 to 5 constraints by adapting samples from E2E [134]. Different from the original task, we ask the model to extract the flat meaning representations according to the corresponding natural language texts. The number of constraints increases with the number of attributes and the number of restaurants. We use exact match as the evaluation metric.
- **Document-Level Event Argument Extraction.** We create instructions by adapting samples from WIKIEVENTS [103]. Given a document, the model is required to extract n events that satisfy a specific event template, where $n \in [1, 5]$ corresponds to the number of constraints. We use accuracy as the evaluation metric.
- **Document-Level Named Entity Recognition.** We derive instructions from samples in the CONLL-2003 dataset [168]. We ask the model to extract a single named entity from a provided document. Notably, as the number of constraints rises, the requirements for the retrieved named entity correspondingly increase. For example, "extract one named entity that is a location" $\rightarrow$ "extract one named entity that is a location in east Asia". We use accuracy as the evaluation metric.

- **Text Generation with Language Constraints.** COGNAC [23] is a challenging benchmark wherein models are presented with a topic accompanied by example text and explicit constraints on the text to avoid. We curate data from COGNAC, formulating instructions with 1 to 5 constraints by integrating additional linguistic restrictions from WordNet [124] and Wikidata [174].

- **Open-ended Question Answering.** We first choose initial instructions from existing datasets including self-instruct evaluation set [180], helpful evaluation released by Anthropic[9], Vicuna evaluation[211], and Koala evaluation[59], as well as open-source platforms such as Quora [1], Reddit [2], and ShareGPT [3]. Given the challenges associated with iteratively adding constraints to an initial instruction, we prompt GPT-4 with a specific prompt shown in Figure E.1 to generate a new instruction with one more constraint based on the given instruction. The above process is repeated five times. Finally, we obtain a set of instructions ranging from 1 to 5 constraints.

Prompt Template (Open-ended Question Answering in Content Constraints)

You are an Instruction Rewriting Expert. You need to rewrite #Given Instruction# based on #Rewriting Requirement#, in order to obtain a #Rewritten Instruction#. Basically, #Rewritten Instruction# should adhere to the following guidelines:
1. Your rewriting cannot omit the non-text parts such as the table and code in #Given Instruction#.
2. #Rewritten Instruction# must be reasonable and must be understood and responded by humans.
3. You should try your best not to make the #Rewritten Instruction# become verbose, #Rewritten Instruction# can only add 10 to 20 words into #Given Instruction#.

#Given Instruction#
{given_instruction}

#Rewriting Requirement#
Please add one proper content constraint to the #Given Instruction#. The content constraints include but are not limited to:
1. Add a Subtask or Another Related Question.
2. Narrow Down the Topic: Instead of a general theme or topic, provide a more specific subset.
3. Set a Higher Standard: Raise the bar for what's considered acceptable or successful.
4. Limit Resources: Restrict the number or type of resources someone can use.
5. Introduce Specific Criteria: Mandate particular components or features that must be included.
6. Specifying Sequence: Dictate the order in which certain steps or actions should be taken.

#Rewritten Instruction#

Figure E.1: The prompt template for Open-ended Question Answering in Content Constraints.

[1] https://www.quora.com

[2] https://www.reddit.com

[3] https://sharegpt.com

E.1.2 Situation Constraints

The data of situation constraints is constructed from tasks as follows:

- **Suggestion Generation, Role-playing.** We collect multi-level instructions that fit within the paradigm of situation constraints from Open-ended Question Answering datasets and online platforms. Examples include asking the model to give suggestions under specific circumstances, asking the model to act as a terminal and output based on the given information, etc.
- **Math Word Problems.** The initial instructions are collected from GSM8K [38] and AGIEval [212]. We then manually add constraints progressively by enhancing the situation descriptions, ensuring that the core question remains unaltered. We use accuracy as the evaluation metric.
- **Time/Spatial Reasoning.** We generate data by refining samples from BIG-Bench Hard [161]. For Time Reasoning, we increase the difficulty level by incorporating additional temporal concepts, such as weeks, months, and years. In the realm of Spatial Reasoning, we opt for a logical deduction task that necessitates deducing the order of a sequence of objects. Here, the number of constraints escalates by augmenting the task with detailed location descriptions for a new object. We use accuracy as the evaluation metric.
- **Code Generation.** We sourced initial instructions from HumanEval [25] and enhanced the difficulty level by adding complexity to the function descriptions within the instructions. We use pass@1 [92] as the evaluation metric.

E.1.3 Example Constraints

Specifically, we choose 40 diverse NLP tasks from PromptSource [7], where each task has more than 5 question templates. Additionally, we create 29 answer templates (shown in Table E.1) that regulate the format of the response. For instructions at difficulty level 1, we utilize the standard 5-shot prompting, where 5 shots are equipped with 1 sampled question template and 1 sampled answer template, and the model is required to respond

to a query using the answer template. For instructions at difficulty level n ($1 < n \leqslant 5$), the 5 shots are randomly paired with n question templates and n corresponding answer templates. Based on the question template of the query, the model is required to recognize the matched question template in the 5 shots and respond using the corresponding answer template. We use accuracy as the evaluation metric.

Answer template
{question}\n{answer}
{question}\nA: {answer}
{question}\nAnswer: {answer}
{question}\nANSWER: {answer}
{question}\n[Answer]\n{answer}
{question}\n#Answer#\n{answer}
{question}\nThe answer is: {answer}
{question}\n{"answer": "{answer}"}
{question}\n{"Answer": "{answer}"}
{question}\n<body>{answer}</body>
{question}\nResponse: {answer}
{question}\nRESPONSE: {answer}
{question}\n[Response]\n{answer}
{question}\n#Response#\n{answer}
{question}\nThe response is: {answer}
{question}\n{"response": "{answer}"}
{question}\n{"Response": "{answer}"}
{question}\nBot: {answer}
{question}\nBOT: {answer}
{question}\n[Bot]\n{answer}
{question}\n#Bot#\n{answer}
{question}\nThe response of the bot is: {answer}
{question}\n{"bot": "{answer}"}
{question}\n{"Bot": "{answer}"}
{question}\nAI assistant: {answer}
{question}\n[AI assistant]\n{answer}
{question}\n#AI assistant#\n{answer}
{question}\nThe response of the AI assistant is: {answer}
{question}\n{"AI assistant": "{answer}"}

Table E.1: Answer template of Example Constraints.

E.1.4 Mixed Constraints

In this section, we consider four below tasks which are naturally suitable for constructing mixed constraints:

- **Text Editing.** We start by gathering text from different online sources, like sentences, letters, and emails. Next, we create instructions with multi-level mixed constraints by increasingly adding an editing requirement to the text at each level. For example, "swap the first and last words in the sentence" (Content Constraints), "response using '###' at the beginning" (Format Constraints), etc. We write rule-based programs for individual instructions to assess the satisfaction of internal constraints, employing exact match as the evaluation metric.

- **Summarization.** The initial instructions are sampled from CNN/Daily Mail[130], XSum [131], SAMSum [62], English Gigaword [64], and arXiv [3]. The instructions with multi-level mixed constraints are produced by specifying the format of generating answers (Format Constraints), requiring the generated text to include or not include certain keywords (Content Constraints), etc. We write rule-based programs for individual instructions to assess the satisfaction of internal constraints, employing accuracy as the evaluation metric.

- **Machine Translation.** The initial instructions are sampled from OpenSubtitles [110], TED Talks [20], and News-Commentary [167]. Then we construct instructions from level 1 to level 5 using a similar pipeline as that of Summarization. We write rule-based programs for individual instructions to assess the satisfaction of internal constraints, employing accuracy as the evaluation metric.

- **Story Generation.** We collect initial instructions from ROCStories [128] and WritingPrompts [54]. Then we add 5 mixed constraints sequentially to the initial instructions based on the ground truth, such as the number of sentences in the generated story (Format Constraints), requiring the generated text to include certain keywords (Content Constraints), specifying the writing style (Style Constraints), etc.

Prompt Template (Open-ended Question Answering in Style Constraints)

You are an Instruction Rewriting Expert. You need to rewrite #Given Instruction# based on #Rewriting Requirement#, in order to obtain a #Rewritten Instruction#. Basically, #Rewritten Instruction# should adhere to the following guidelines:
1. Your rewriting cannot omit the non-text parts such as the table and code in #Given Instruction#.
2. #Rewritten Instruction# must be reasonable and must be understood and responded by humans.
3. You should try your best not to make the #Rewritten Instruction# become verbose, #Rewritten Instruction# can only add 10 to 20 words into #Given Instruction#.

#Given Instruction#
{given_instruction}

#Rewriting Requirement#
Please add one proper style constraint that #Given Instruction# does not have. The style constraints include but are not limited to:
1. Tone and Emotion: Specify the desired emotional tone for the response.
2. Writing Style: Ask the AI to mimic a specific author's writing style.
3. Contradiction: Ask the AI to provide a response that contradicts the previous statement or take a stance opposite to its prior response.
4. Ambiguity: Instruct the AI to create responses with intentional ambiguity or double meanings.
5. Humor or Satire: Request that the response be humorous or satirical, requiring the AI to generate jokes or witty remarks.

#Rewritten Instruction#

Figure E.2: The prompt template for Open-ended Question Answering in Style Constraints.

Prompt Template (Open-ended Question Answering in Format Constraints)

You are an Instruction Rewriting Expert. You need to rewrite #Given Instruction# based on #Rewriting Requirement#, in order to obtain a #Rewritten Instruction#. Basically, #Rewritten Instruction# should adhere to the following guidelines:
1. Your rewriting cannot omit the non-text parts such as the table and code in #Given Instruction#.
2. #Rewritten Instruction# must be reasonable and must be understood and responded by humans.
3. You should try your best not to make the #Rewritten Instruction# become verbose, #Rewritten Instruction# can only add 10 to 20 words into #Given Instruction#.

#Given Instruction#
{given_instruction}

#Rewriting Requirement#
Please add one proper format constraint that #Given Instruction# does not have. The format constraints include but are not limited to:
1. Length: Imposing constraints on the length of individual words, sentences, or paragraphs.
2. Hierarchical Instructions: Providing instructions that have a hierarchical structure, where the AI needs to understand and follow a hierarchy of tasks to construct a response.
3. Special Output Format: Asking the AI to respond by using data format like table, json, HTML, LaTeX, etc.
4. Morphological Constraints: Asking the AI to avoid or use specific morphemes.
5. Multi-lingual Constraints: Asking the AI to respond in multiple languages or switch between languages according to complex patterns.
6. Incorporation of Specific Literary Devices: Requiring the inclusion of specific, and perhaps numerous, literary devices.
7. Following a Specific Grammatical Structure: Requiring the AI to create responses that strictly follow a particular grammatical structure.

#Rewritten Instruction#

Figure E.3: The prompt template for Open-ended Question Answering in Format Constraints.