****

**本科毕业论文（设计）**

论文题目：基于微调大语言模型与多Agent协作的数学推理

工作流系统研究

指导老师： 王 曼

学生姓名 陈 柏 政

学 号： 20214001042

学 院： 人工智能学院

专 业： 人工智能

毕业时间： 2 0 2 5 年 6 月

**基于微调大语言模型与多Agent协作的数学推理工作流系统研究**

**摘 要**

本研究聚焦于大语言模型（LLM）推理能力的定向优化和多智能体协同系统的构建。研究的核心目标是从大语言模型的思维链（Chain of Thought, CoT）推理机制入手，探索并优化其推理能力，进而提升模型在复杂任务中的表现。我们采用了递进式的实验框架，分阶段进行系统化优化探索，以确保在多个层面上对模型性能进行全面的提升。

第一阶段，以 Qwen2-1.5B-Instruct 作为基础模型，实施监督微调（Supervised Fine-Tuning, SFT）。在此基础上，我们设计了双分支训练路径，其中一个分支专注于基础推理能力的训练，另一分支则专注于知识增强。通过构建并利用开源高质量的 CoT 数据集，分别对这两条分支进行了微调，并通过 MMLU、GSM8K 和 HUMANEVAL 等标准基准集对产出的模型进行了多维度评估。此外，为探究如何提升模型推理性能，研究引入了混合专家模型（Mixture of Experts, MoE）架构，探索了模型容量的扩展方案，以提升大模型在推理任务中的泛化能力。

第二阶段以 Qwen2.5-1.5B-Instruct为主体，构建 "知识蒸馏 - 强化学习" 二级优化研究体系：首先通过 DeepSeek-R1 蒸馏数据集进行知识蒸馏（Knowledge Distillation, KD），以此数据为基础微调产出了两版 R1-Distlled-1.5B 基础模型；后续再基于Qwen2.5基座模型以及Deepseek-R1官方蒸馏模型尝试复现R1-Zero的训练路线，采用群体相对策略优化算法（GRPO）开展强化学习微调，同时采取不同的上下文长度进行模型的优化提升，最终得到两版的性能表现均有一定提升的轻量级推理模型。

最终通过整合自研模型与主流 API 模型，基于 Dify 开源框架开发一个简单的演示框架进行应用层的封装，实现数学推理能力的场景化验证。

本研究不仅提出了一个基于 "监督训练 - 知识迁移 - 强化学习策略优化" 的三级优化范式，还在技术层面验证了 GRPO 算法在推理任务中的有效性。研究结果表明，该范式不仅能显著提升模型的推理能力，还为领域特定的大语言模型的优化提供了一个具有可复现性的技术路径，具有重要的学术价值和实践意义。

关键词：大语言模型，推理能力，有监督微调，知识蒸馏，强化学习

**Research on the Mathematical Reasoning Workflow System Based on Fine-Tuned Large Language Models and Multi-Agent Collaboration**

**Abstract**

This research focuses on the targeted optimization of reasoning capabilities in large language models (LLMs) and the construction of multi-agent collaborative systems. The core objective is to explore and optimize the reasoning abilities of LLMs starting from their Chain of Thought (CoT) reasoning mechanism, aiming to enhance model performance in complex tasks. A progressive experimental framework is adopted, with a phased optimization approach to systematically improve model performance across multiple levels.

In the **first phase**, Qwen2-1.5B-Instruct is used as the base model for supervised fine-tuning (SFT). A dual-branch training strategy is designed, with one branch dedicated to foundational reasoning and the other to knowledge enhancement. The two branches are fine-tuned using publicly available CoT datasets, and the resulting models are evaluated across multiple dimensions using benchmark sets such as MMLU, GSM8K, and HUMANEVAL. Additionally, a Mixture of Experts (MoE) architecture is introduced to explore model capacity expansion, enhancing the generalization ability of large models in reasoning tasks.

In the **second phase**, a two-stage optimization system combining "knowledge distillation (KD) - reinforcement learning" is constructed using Qwen2.5-1.5B-Instruct as the core model. Knowledge distillation is first applied using the DeepSeek R1 distillation dataset to generate two versions of the R1-Distilled-1.5B base model. Subsequently, training routes from R1-Zero are replicated using the Qwen2.5 base model and the DeepSeek-R1 official distilled model, with reinforcement learning fine-tuning applied using the Group Relative Policy Optimization (GRPO) algorithm. The optimization process includes varying context lengths for model enhancement, resulting in two versions of lightweight reasoning models with improved performance.

Finally, a prototype system is developed by integrating self-developed models with mainstream API models, using the Dify open-source framework to create an application layer for the scene-based validation of mathematical reasoning capabilities.

This study proposes a three-tier optimization paradigm of "supervised training - knowledge transfer - reinforcement learning policy optimization" and validates the effectiveness of the GRPO algorithm in reasoning tasks. The findings demonstrate that this paradigm significantly improves the reasoning capabilities of models and provides a reproducible technical pathway for optimizing domain-specific LLMs, with significant academic and practical implications.

Keywords： Large Language Model，Reasoning Capabilities，Supervised Fine-Tuning， Knowledge Distillation，Reinforcement Learning

目 录

[**摘 要** 2](#_Toc194275951)

[**Abstract** 4](#_Toc194275952)

[第一章 引言 8](#_Toc194275953)

[1.1 研究背景及意义 8](#_Toc194275954)

[1.2 研究内容及创新点 9](#_Toc194275955)

[1.2.1 研究内容 9](#_Toc194275956)

[1.2.2 创新点 10](#_Toc194275957)

[1.3 本文组织结构 11](#_Toc194275958)

[第二章 国内外研究现状 12](#_Toc194275959)

[2.1 大语言模型发展概述 12](#_Toc194275960)

[2.2 推理大模型发展概述 15](#_Toc194275961)

[2.3 大模型微调训练方法的发展 18](#_Toc194275962)

[2.4 大模型推理优化的相关技术发展 20](#_Toc194275963)

[第三章 大模型训练和相关算法技术研究 21](#_Toc194275964)

[3.1 Transformer模型架构分析 21](#_Toc194275965)

[3.2 Qwen系列大语言模型分析 24](#_Toc194275966)

[3.2.1 Qwen系列大模型架构和训练流程分析解读 24](#_Toc194275967)

[3.2.2 Qwen2系列大模型架构和训练流程分析解读 26](#_Toc194275968)

[3.2.3 Qwen2.5系列大模型架构和训练流程分析解读 32](#_Toc194275969)

[3.3 有监督微调算法技术解析 33](#_Toc194275970)

[3.3.1 SFT简介 33](#_Toc194275971)

[3.3.2 SFT数学表示解析 34](#_Toc194275972)

[3.4 大模型强化学习微调算法解析 36](#_Toc194275973)

[3.4.1 ‌近端策略优化算法解析 36](#_Toc194275974)

[3.4.2 直接优化策略算法解析 43](#_Toc194275975)

[3.5 Deepseek-R1 训练策略和关键算法解析 45](#_Toc194275976)

[第四章 数据准备和模型训练实验方案设计 51](#_Toc194275977)

[4.1 数据准备 51](#_Toc194275978)

[4.1.1 知识强化类数据和普通思维链推理数据准备 51](#_Toc194275979)

[4.1.2 带思考过程的知识蒸馏数据准备 51](#_Toc194275980)

[4.1.3 用于GRPO强化学习训练的数据准备 52](#_Toc194275981)

[4.2 实验环境和工具框架介绍 53](#_Toc194275982)

[4.2.1 实验环境 53](#_Toc194275983)

[4.2.2 实验工具框架 54](#_Toc194275984)

[4.3 实验方案和评估指标 56](#_Toc194275985)

[4.3.1 实验方案 56](#_Toc194275986)

[4.3.2 评估指标 57](#_Toc194275987)

[第五章 实验结果分析 57](#_Toc194275988)

[第六章 简单应用模版的搭建演示 57](#_Toc194275989)

[第七章 研究总结与未来展望 57](#_Toc194275990)

[第八章 参考文献 57](#_Toc194275991)

# 第一章 引言

## 1.1 研究背景及意义

近年来，大语言模型（LLM）[1]在自然语言处理（NLP）任务中的表现取得了显著突破，尤其在问答、推理、代码生成等任务中展现出强大的泛化能力。然而，尽管 LLM 具备一定的推理能力，其在复杂逻辑推理、数学计算、多步骤问题求解等方面仍存在明显局限性。推理能力不足不仅影响模型的可解释性和可靠性，也限制了其在专业领域的应用。因此，如何针对性地优化 LLM 的推理能力，提升其在复杂任务中的表现，已成为当前学术界和工业界广泛关注的研究热点。

为提升 LLM 的推理能力，近年来涌现出诸多方法，如思维链推理（Chain of Thought, CoT）、自回归推理增强、检索增强生成（RAG） 以及基于强化学习的策略优化[2-4]等。其中，CoT 通过引导模型进行逐步推理，使其在多步骤推理任务中具备更好的逻辑一致性，已被广泛应用于数学推理、代码生成等任务。然而，当前主流的 CoT 方法多依赖于大规模预训练和少量的指令微调，缺乏对推理能力的定向优化。此外，由于 LLM 训练过程中计算资源需求巨大，研究如何利用知识蒸馏（Knowledge Distillation, KD）和强化学习（Reinforcement Learning, RL）[5, 6]等技术，以更轻量级的方式提升模型推理能力，具有重要的研究价值。

本研究针对 LLM 的推理能力优化问题，结合监督微调（Supervised Fine-Tuning, SFT）[7]、知识蒸馏（KD） 和 强化学习（RL），提出了一种“监督训练 - 知识迁移 - 强化学习策略优化”的三级优化范式。通过构建多智能体协同优化体系，本研究探索了如何在不同参数量级的模型上增强推理能力，并采用混合专家模型（MoE）架构[8]扩展模型容量，以进一步提升其在多任务环境下的泛化能力。此外，研究还基于 Dify [9] 开源框架开发数学推理能力验证系统，探索 LLM 在应用场景中的可行性。

本研究的意义体现在以下几个方面：

优化 LLM 的推理能力，针对 LLM 在复杂逻辑推理任务中的局限性，提出了一种系统化的优化方法，通过指令微调嵌入思维模型和同构强化学习令模型自发的拥有推理能力，提升模型在数学推理、多步骤推理等任务上的表现。

提出可复现的优化范式，结合 SFT、KD 和 RL 等技术，构建了一种可复现的优化框架，为不同参数规模的 LLM 训练提供技术路径参考。

提升轻量级模型的推理性能，通过知识蒸馏与强化学习微调，使小型模型在推理能力上接近甚至超越更大规模的模型，从而降低计算成本，提高部署效率。

验证 GRPO 算法的有效性，在强化学习优化环节，引入群体相对策略优化（GRPO）算法，并在推理型LLM 的微调任务中进行了实验验证，进一步拓展了该算法在大模型训练中的应用价值。

本研究不仅尝试为 LLM 的推理能力优化提供了一条高效、可复现的技术路径，同时也为轻量级 LLM 的推理增强、强化学习在大模型训练中的应用等方向提供了相关思路，在一定程度上具有重要的学术价值和应用前景。

## 1.2 研究内容及创新点

### 1.2.1 研究内容

本研究围绕大语言模型（LLM）推理能力的定向优化和多智能体协同系统的构建展开，具体研究内容可归纳为以下三个阶段：

**第一阶段，基础推理能力的优化与模型扩展**

在此阶段，研究以 Qwen2-1.5B-Instruct [10] 为基础模型，实施监督微调（Supervised Fine-Tuning, SFT）。通过设计双分支训练路径，分别优化基础推理能力和知识增强能力。第一条分支专注于基础推理能力的训练，利用公开的 高质量CoT思维链数据集进行微调；另一条分支则针对知识增强，通过相应的优化策略提高模型的知识整合能力。在此基础上，研究通过 MMLU [11]、GSM8K [12] 和 HUMANEVAL [13]等基准数据集[14-16]对优化后的模型进行多维度评估，确保模型的推理能力在不同任务上的有效性。进一步地，为探索如何提升模型推理性能，研究引入了混合专家模型（Mixture of Experts, MoE）架构[17]，以扩展模型容量，并提升模型在多任务推理中的泛化能力。

**第二阶段，轻量级模型优化与强化学习微调**

第二阶段的研究以 Qwen2.5-1.5B-Instruct [18] 为核心，构建了一个“知识蒸馏 ——强化学习”二级优化体系。首先，研究通过 DeepSeek R1[19] 蒸馏数据集进行知识蒸馏（Knowledge Distillation, KD），产生了两个 R1-Distilled-1.5B 基础模型。随后，基于 Qwen2.5 基座模型和 DeepSeek 官方蒸馏模型，研究复现了 R1-Zero 的训练路线，并采用群体相对策略优化算法（Group Relative Policy Optimization, GRPO）[20]进行强化学习微调。此阶段的目标是进一步优化轻量级推理模型，在保证模型推理性能的同时，通过少量训练资源达到更高效的优化效果。

**第三阶段，自研与主流 API 模型整合与应用验证**

在最终的系统验证阶段，研究将自研的优化模型与主流 API 模型进行整合，并基于 Dify 开源框架开发了一个简单的演示框架。该框架用于封装推理能力模块，并实现数学推理任务的场景化验证。通过该验证系统，评估模型在实际应用中的表现，进一步证明了所提优化方法在真实场景中的可行性和有效性。

### 1.2.2 创新点

在本研究工作中，研究创新方面可以总结为以下五点：

**1. 推理能力定向优化框架**，本研究提出了一种基于“监督训练 - 知识迁移 - 强化学习策略优化”的三级优化范式，通过精细化的模型微调和知识增强策略，显著提升大语言模型（LLM）的推理能力。这一优化框架系统性地结合了监督微调、知识蒸馏与强化学习等多种技术，为模型推理性能的提升提供了新的思路。

**2. 双分支训练路径的设计**，本研究创新性地提出了双分支训练路径，其中一分支专注于基础推理能力的提升，另一分支则专注于知识增强。通过对 CoT 数据集的利用，双分支路径能在不同层次上强化模型的推理能力，为复杂任务提供更加精细的优化手段。

**3. 混合专家模型（MoE）架构的引入**，在推理能力提升的过程中，研究引入了混合专家模型（Mixture of Experts, MoE）架构。这一架构能够在模型容量扩展的同时提高计算效率，进而优化大模型的推理性能，提升其在多任务环境下的泛化能力。

**4. 轻量级推理模型的优化与强化学习微调优化的探索**，本研究通过知识蒸馏（KD）与群体相对策略优化（GRPO）算法，进行了两种高效的轻量级推理模型优化方案的尝试。通过蒸馏技术将思维链能力通过指令微调的方式嵌入模型生成权重参数中，以及进行探索通过强化学习微调让模型自发的拥有长推理思维链能力，在第二种尝试中，采用混合上下文长度的训练方法，在一定程度上可以进一步优化模型针对复杂任务的推理能力，最终两种方法都成功提升了小模型在推理任务中的表现，实现了推理能力和计算效率的平衡。

**5. GRPO算法在推理模型训练中的应用验证**，本研究通过在LLM 训练任务中验证了群体相对策略优化（GRPO）算法的有效性。通过强化学习优化模型推理策略，GRPO 在提升模型表现的同时，还能有效降低训练资源的消耗，拓展了该算法在大规模模型训练中的应用场景。

## 1.3 本文组织结构

本文共有六个章节组成，每个章节的主要内容如下：

第一章，引言。在本章节中，首先介绍了推理模型的研究背景和目的，明确了研究的核心任务和目标，阐述了推理能力优化与多智能体协同系统构建的重要性。紧接着结合当前发展现状，概述了本文的研究内容和创新点，详细描述了研究过程中提出的技术框架及优化方法。

第二章，国内外研究现状。在此章节中，主要探讨了了在本研究工作中涉及的相关技术研究，包括三个方面：大语言模型的研究，推理模型的相关发展历程以及模型训练技术的研究。与此同时，对相关研究的代表技术和算法进行详细分析，并总结了不同代表算法的技术特点。

第三章，大模型训练和相关算法技术研究。针对本研究涉及的相关技术进行详细阐述，包括但不限于大模型架构分析，有监督微调方法解析以及强化学习算法分析。

第四章，数据处理和模型训练实验方案设计。本章首先介绍了本研究的基础设施和硬件配置，重点介绍了多种开发工具的使用。然后，详细说明了数据的格式清洗与预处理过程，并介绍了在各个阶段所采用的评测指标和模型训练的相关信息。详细介绍本研究的实验设计整体方案，为模型微调以及推理模型训练进一步探索提供了一定的思路。

第五章，实验结果展示与分析。本章详细展示并分析了在第一阶段和第二阶段实验中取得的结果。包括对各阶段模型的性能评估、损失函数变化、KL散度、以及生成文本长度等指标的变化进行分析，重点展示了不同版本模型的性能提升情况，探讨了模型优化过程中的主要挑战与解决方案。

第六章，Dify应用模版的搭建与演示**。**本章介绍了基于Dify开源框架开发大模型的简单应用模版，并进行演示。通过将自研模型与主流API模型整合，展示了如何在实际场景中应用优化后的推理模型，尤其是在数学推理等具体任务中的应用验证。该部分展示了模型的实际应用效果，证明了优化方法在真实环境中的可行性与有效性。

# 第二章 国内外研究现状

## 2.1 大语言模型发展概述

大语言模型（LLM, Large Language Models）是近年来人工智能领域的一个重要研究方向，其核心目标是通过大规模数据和强大的计算能力，构建能够理解和生成自然语言的模型。

大语言模型的崛起源自谷歌Vaswani等人于2017提出的Transformer架构的提出及其注意力机制（Attention Mechanism）[21]的成功应用，Transformer的核心创新在于引入了自注意力机制（Self-Attention），该机制能够在处理序列数据时捕捉输入序列中各个位置之间的依赖关系，而不需要传统的循环神经网络（RNN）或长短期记忆网络（LSTM）[22]中的逐步迭代。这种机制通过计算输入序列中每个元素与其他元素的相关性，从而实现全局信息的捕捉，相比于传统方法具有显著的并行计算优势。Transformer的关键组件包括**多头注意力**（Multi-Head Attention）、**位置编码**（Positional Encoding）和**前馈神经网络**（Feedforward Neural Networks），这些都为模型的深度学习和并行计算奠定了基础。自注意力机制不仅提升了模型的训练效率，还增强了模型处理长程依赖的能力，使其能够在自然语言处理（NLP）任务中表现出色。

在Transformer架构的基础上，BERT（Bidirectional Encoder Representations from Transformers）[23]实现了一个重大的突破，提出了一种新的预训练和微调方法。BERT采用了Transformer的Encoder部分，最显著的特点是其双向性。与传统的单向语言模型（例如左到右或右到左的语言模型）不同，BERT通过在预训练阶段进行双向上下文的学习，能够在理解文本时同时利用左右两侧的信息，从而提高了对句子整体语义的理解能力。BERT采用了两种主要的预训练任务：Masked Language Model（MLM）和Next Sentence Prediction（NSP）[24]。MLM任务通过遮蔽输入序列中的一些词汇，让模型预测这些被遮蔽的词，而NSP任务则帮助模型学习句子之间的关系。BERT的成功标志着预训练模型的时代来临，预训练加微调（Pretrain-Finetune）[25]的方法成为了大规模NLP模型的标准训练流程。

与BERT不同，GPT（由OpenAI提出）更侧重于生成任务。GPT系列模型采用了Transformer的Decoder部分，并且利用了自回归的方式生成文本。自回归模型的特点是，模型通过逐步生成下一个词汇，并利用前面的词汇生成后续的内容，这种方式特别适合生成任务，如文本生成、机器翻译等。GPT的第一个版本（GPT-1）是一个相对较小的模型，但随着后续版本的发布（GPT-2、GPT-3、GPT-4）[26-29]，模型的规模大幅度提升，参数数量也从1.1亿增加到1750亿最后到到现在的闭源商业模型，并在多个NLP任务上展示出了卓越的性能。GPT的成功关键在于其庞大的参数量和超大规模的训练数据，使得模型具备了强大的语言理解和生成能力。GPT-3的发布进一步推动了大语言模型的研究热潮，其生成的文本不仅在语言流畅性和内容连贯性上表现出色，且能够应对许多复杂的推理和生成任务。GPT系列强调的是无监督预训练和零-shot学习能力，即模型在没有专门任务微调的情况下，能够处理各种NLP任务。随着后续性能表现更加强大的GPT-4O以及各种新型AI产品的到来，大语言模型逐渐在各个领域上展露具有无限可能的发展潜力。

LLaMA是Meta（前Facebook）提出的一个新型大语言模型系列[30]，旨在通过较小的计算资源实现较大的性能突破。LLaMA的核心创新之一是其对不同规模模型的训练进行了优化同时在模型架构上进行了优化，例如使用RMS Norm 归一化函数，使用SwiGLU替代了ReLU作为激活函数，利用ROPE通过绝对位置编码的方式实现相对位置编码[31]，采用了高效的参数量与计算资源的平衡，使得较小的LLaMA模型能够在处理复杂任务时仍保持较高的性能。LLaMA系列在数据处理、模型训练和推理效率上做出了创新，其主要目标是通过在更少的计算资源上训练出更强的语言能力，进而为广泛的AI应用提供更高效的解决方案。LLaMA在多个标准基准上展示了优异的性能，特别是在低资源设备上，能够有效进行推理任务。近年来也在研究领域不断开源分享相关模型和技术，为大语言模型发展奠定了一定的基础。

而国内的研究团队也是紧跟热点，随着阿里通义千问团队推出Qwen系列模型，结合了最新的强化学习和优化技术，通过采用BPE分词器扩展词表，同时在QKV的注意力机制中添加了偏差增强模型的外推能力[32]，同时使用了Flash Attention技术[33]加快整体的训练速度。Qwen的核心优势在于其针对特定任务的定向优化策略，包括在基础推理能力和知识增强方面的双分支训练路径。通过这种优化策略，Qwen能够在多个NLP和推理任务中展现出强大的性能，并且具有更好的泛化能力和计算效率。Qwen系列的推出，标志着国内大语言模型在推理能力和技术创新方面的快速发展，进一步推动了大语言模型技术的多元化应用。与此同时，国内其他研究团队也相继分享开源了自行研发训练的国产化大模型，如Baichuan、InternLM、Minicpm[34-36]等，这些模型都针对中文语言做了相关的优化和数据处理，同时尝试利用更小的参数量训练出更加强大的模型，更适合国内用户开发使用。

与此同时，Scaling Laws也成为了业界公认的大模型提升性能的标准。Scaling Laws是指：随着模型大小、数据集大小和用于训练的计算浮点数的增加，模型的性能会提高。并且为了获得最佳性能，所有三个因素必须同时放大。当不受其他两个因素的制约时，模型性能与每个单独的因素都有幂律关系[37]。因此随着模型规模和技术的不断进步，未来的大语言模型将在多个领域展示出更强的表现，推动NLP和AI应用的广泛落地。

## 2.2 推理大模型发展概述

推理大模型（Reasoning Large Language Models，RL-LLMs）作为大语言模型（LLM）的一个重要分支，专注于模型在复杂推理任务中的表现。推理任务通常涉及到抽象思维、逻辑推理、数学问题求解、常识推理等，这些任务不仅需要模型具备强大的语言理解能力，还要求模型在推理过程中能够做出合理的判断和推导。推理大模型的崛起源于对基础大语言模型（如BERT、GPT等）能力的拓展和优化，特别是在深层推理和推断能力上的提升。

在增强大语言模型推理能力的研究中，提示词工程、链式思维推理、少样本推理(Few-shot prompting) [38, 39]等技术日益成为提升模型表现的重要方法。这些技术通过细化问题的理解过程，再通过部分示例样本促使模型能够更系统性地分步推理，从而显著提升其在复杂任务中的表现。CoT推理通过内嵌推理链条，引导模型逐步分解任务，以实现更连贯的推理路径，尤其适用于多步骤逻辑推理与数学问题。CoT的核心思想是，在推理过程中，模型通过模拟思考链的方式逐步展开推理，而不是直接给出答案。这种方法可以显著提升模型在复杂推理任务中的表现。CoT机制通过将推理过程显式化，允许模型逐步推导出结论。这种逐步推理的方法不仅提高了模型在复杂推理任务中的准确性，也增强了模型的解释性和可控性。例如，在数学问题求解时，CoT使得模型能够分步骤推理，展示出推理过程而不仅仅是输出最终答案，从而大大提升了推理的准确性和可靠性在此基础上，少样本推理策略(如TOT , Tree of Thought) [40]通过有限的样本展示推理路径，有效减少了对大规模标注数据的依赖，同时使模型能够更灵活地应对多样化问题。

随着推理任务的日益复杂，强化学习（Reinforcement Learning, RL）被引入推理大模型的优化过程，以进一步提升推理能力。强化学习通过奖励机制引导模型逐步改进推理策略，尤其适用于需要在多轮交互或连续决策中进行优化的任务。强化学习微调（RL Fine-Tuning）[41]已经成为推理大模型中一种重要的技术，尤其是在解决诸如对话生成、复杂数学问题和推理题目时，它能够提升模型的推理准确性。DeepMind提出的AlphaCode模型[42]，通过在编程和推理任务中的强化学习微调，取得了显著的成功。AlphaCode能够通过学习大量编程题目中的输入输出模式，逐步推理并生成符合题意的代码，标志着推理大模型在代码理解和生成上的一大突破。

Numina 团队是一个专注于研究大语言模型数学能力的团队，于2024年7月参加并获得AI 数学奥林匹克 (AI Math Olympiad，AIMO) 的首届进步奖，研发的Numina-Math的数学解题模型采用两阶段训练方法[43]，第1阶段通过在自然语言“数学题 + 解答”的大规模、多样化数据集上微调基础模型，其中每个解答都需套用思维链模板以促使 LLM 进行推理。第 2 阶段则是在Python代码工具整合推理的合成数据集上微调第 1 阶段得到的模型，其中每个数学题都分解为一系列推理、Python程序及其输出。其遵循微软的ToRA [44]的做法，提示模型以ToRA格式生成带有代码执行反馈的解答。对这些数据进行微调会产生一个推理代理，它可以通过将自然语言推理和使用外部Python REPL来计算中间结果结合起来以解决数学问题。后续，阿里云通义千问团队也用类似的方法即CoT+TIR的两阶段方式强化模型的数学推理能力，训练并开源Qwen2.5-Math系列模型[45]，其中还涉及到强化学习的人类反馈微调[46]。

在2024年9月份，OpenAI正式推出了其闭源推理模型GPT-O1[47]，在处理复杂推理问题和理解长文本推理方面具备了更强的能力, GPT-O1强化了思维链推理（Chain of Thought, CoT）能力，优化了模型在多步骤推理中的表现。其通过强化学习算法引导模型逐步推导解决方案，在训练过程中引入了任务自适应的机制，针对不同类型的推理任务（如常识推理、数学推理、程序推理等），通过微调和数据集增强，O1能够在复杂推理任务中显著减少逻辑错误，并提升推理的准确性和可解释性, 为了更好地处理长文本推理任务，GPT-O1对模型的上下文理解能力进行了增强。在推理过程中，GPT-O1能够处理更多的上下文信息，优化了模型在长文推理中逐步推理和保持一致性的能力，成为了业界的SOTA模型。自此，研究具有深度推理思维链能力的推理模型逐渐开始成为各界的研究热点。

国内Deepseek-AI团队也一直在推理模型的领域发光发热，从Deepseek-Math到Deepseek-Coder [20, 48, 49]最后到利用更低的训练推理成本却能达到业界SOTA O1效果的Deepseek-R1 [19]，在研究过程中提出了群体相对策略优化（GRPO）算法，能够直接剔除常规近端强化学习算法PPO流程中的Value Model[3]以及直接在基座模型上进行强化学习的训练思路使模型具备推理反思的能力，而不是像以往的常规训练流程一样，先通过大量高质量思维链数据加有监督微调的方式让模型扩充能力后再利用强化学习增强模型对思考过程和最后答案的优化，并在探索完推理能力的提升后，为了完善优化模型输出的可读性和可解释性，进行了高质量的普通知识数据冷启动的方式，在推理和日常交流能力中寻找一定的平衡，同时为了节省训练推理的GPU显存，利用MoE机制和MLA（Multi-Head Latent Attention）。Deepseek-R1的爆火在一定程度上也象征着我国在推理模型领域能够比肩国外的顶尖水平，甚至能够做到利用更低的时间和金钱成本进行训练推理。其开源相关训练方法更是给诸多大模型研究团队提供了十分宝贵的训练思路，于2025年3月，通义千问团队也正式开源了其高性能推理模型QwQ-32B[50]，而R1的参数量为671B，基本少了20倍的参数量却能比肩R1，代表国内推理模型的研发水平更进一步。

## 2.3 大模型微调训练方法的发展

大模型的微调（Fine-Tuning）是深度学习研究中的一个核心课题，尤其是在自然语言处理（NLP），微调已经成为提升模型性能的主要手段。随着大模型（如GPT、BERT、T5、Qwen、LLaMA等）的不断发展和复杂度以及参数两的提升，微调训练方法也经历了显著的演进发展。大模型的微调最早来源于“预训练-微调”（Pre-training and Fine-tuning）范式[25]，这一方法的核心是首先在大规模通用数据集上进行无监督预训练，然后在特定任务上进行有监督微调。BERT作为预训练和微调范式的标志性模型，BERT采用了双向Transformer结构，首先在大规模的文本语料上进行无监督预训练（Masked Language Model），然后针对具体任务进行微调。BERT的创新之处在于它的预训练阶段使用了大规模的语料库进行自监督学习，极大地提升了其在下游任务中的表现。BERT的微调方法通过将预训练得到的参数作为初始值，然后在特定任务上进行适当的微调，从而实现了从通用模型到任务特定模型的转换。与BERT不同，GPT采用了基于自回归的生成模型，模型首先进行预训练（自回归语言建模任务），然后通过微调来实现特定任务的优化。GPT的微调方法同样依赖于将预训练的通用模型应用到特定任务中，针对下游任务的特定数据进行训练，进一步优化性能。

随着大规模预训练模型的逐步成熟，研究者们开始探索如何在多任务、跨领域的情境下进行有效的微调。不同于传统的单一任务微调，任务特定微调（Task-specific Fine-tuning）[51]方法通过考虑不同任务的特性，调整微调策略以达到更好的效果，多任务学习[52]是微调方法中的一种创新策略，它通过共享参数来同时优化多个任务，通常通过共享某些层的参数来利用多个任务之间的关联性。这样的多任务微调方法不仅能够提高模型在各个任务上的表现，还能降低过拟合的风险。

由于大模型的不断扩展和优化，微调方法开始融合更多先进技术，尤其是有监督微调和知识蒸馏，也是目前学术界和工业界为了满足模型应用需求最常用的训练方法。监督微调（Supervised Fine-Tuning，SFT）[53]是一种传统且广泛应用的微调方法，它主要依赖于大量的人工标注数据进行训练。在大模型的背景下，SFT通常指在经过预训练的基础上，针对特定任务或数据集通过有监督的学习来调整模型参数。该过程是根据训练数据中的标签来最小化损失函数，以便模型能够在给定的任务中获得最佳表现。

知识蒸馏（Knowledge Distillation, KD）[5]是一种模型压缩技术，它将一个大型、复杂的教师模型（Teacher Model）中的知识传递到一个较小的学生模型（Student Model）中。通过蒸馏，大模型的知识可以转移到更小的模型中，从而提升小模型的性能。尤其在大规模模型无法直接应用的场景中，知识蒸馏成为了一种有效的微调方法。许多现代的推理大模型（如Deepseek、Qwen等）都在微调过程中使用了蒸馏方法，以提高模型的推理性能。RFT（Rejection sampling Fine-Tuning）拒绝采样来生成和收集正确的推理路径，以此作为增强的微调数据集。RFT能够生成包含更多独特推理路径的增强样本，更大幅度地提升LLMs的数学推理性能。对于性能较差的LLMs，RFT带来的改进更为显著。[54]

强化学习（Reinforcement Learning, RL）在大模型微调中的应用已经取得了显著进展，尤其是在推理任务和决策任务中的效果越来越突出。与传统的监督微调（SFT）不同，强化学习通过奖励信号来引导模型的学习过程，模型并不依赖于人工标注数据，而是通过与环境的交互来探索最优策略。在大语言模型（LLM）中，强化学习的应用可以提高模型在推理、对话、推理能力等方面的表现。近端策略优化算法PPO（Proximal Policy Optimization）是强化学习中的一种策略优化算法，它的目标是通过最大化策略的期望回报来优化模型。在大模型的微调过程中，PPO通过优化模型的策略，使得模型能够在环境中更有效地做出决策。直接偏好优化算法DPO（Direct Preference Optimization）[55]是一种简化版的强化学习算法，旨在直接优化模型生成的输出的“偏好”或“质量”。与PPO等基于策略梯度的算法不同，DPO通过直接对模型的生成结果进行评价和优化，推动模型在任务中更好地优化目标。DPO特别适用于文本生成类任务，例如机器翻译、对话系统和文本生成中的推理任务，能够直接对生成文本的质量进行优化，很适合优化SFT后的大模型，从而提升大模型的任务适应性和响应能力。群体相对策略优化GRPO（Group Relative Policy Optimization）是近期提出的一种强化学习算法，旨在优化模型多个输出群组之间的协作策略，尤其适用于多智能体系统和集体决策的场景。在大模型训练中，GRPO的优势在于它可以在多个模型或多个策略之间进行相对优化，从而提升整体系统的推理性能

## 2.4 大模型推理优化的相关技术发展

推理优化（Inference Optimization）[56]在大模型的发展过程中逐渐成为一个重要的研究方向。随着模型规模的扩大，推理性能成为了一个瓶颈，尤其是在硬件资源和计算时间上。推理优化的目标是减少大语言模型在推理阶段的计算资源消耗，提高响应速度，并降低部署成本。随着计算能力的提升和模型规模的不断增大，推理优化逐渐从初期的硬件加速发展到更加智能化的优化技术，具体包括，硬件加速优化：通过利用专用硬件（如TPU、GPU、FPGA等）来加速模型推理，以减轻计算压力。算法层面的优化：例如，通过压缩模型参数、优化计算图等方式，减少推理时的计算复杂度和内存占用。资源高效性优化：在确保推理质量的前提下，通过量化、剪枝、蒸馏等技术减少计算资源需求，提高模型的推理效率。

量化和剪枝是常见的推理优化技术[56, 57]，能够有效减少模型的参数量、存储需求和计算复杂度，进而提升推理速度。量化 (Quantization)是将浮点数模型参数转换为低位表示（例如8位整数或16位整数）的过程。量化能够减少模型存储和计算的开销，从而加速推理过程。常见的量化方法包括：权重量化、激活量化、训练时量化例如QLORA[58]。剪枝通过去除神经网络中的不重要连接（权重较小或影响较小的部分），来减少网络的复杂性。剪枝后的模型可以显著减少计算量和内存消耗，尤其是在推理时，能够更快地执行。但需要注意，剪枝也可能带来模型性能的下降，因此需要通过精细的控制和再训练来确保性能的保持。

混合专家模型MoE (Mixture of Experts)，是一种基于模型容量和计算资源有效分配的推理优化方法。MoE模型通过将不同的专家（子模型）分配给不同的任务或输入，而不是使用单一的模型进行所有任务。它的主要特点在于：专家动态选择，在每次推理时，根据输入数据动态选择若干个专家进行推理，而不是全模型参与计算。这有效减少了计算量，并提高了推理速度。容量与计算的平衡，MoE使得模型可以在不同的任务中充分利用计算资源，同时避免在不需要全部专家时浪费计算能力。MoE的应用让大规模模型能够在推理过程中进行更高效的计算资源调度，尤其在大规模任务中，MoE能够显著提高计算效率。

与此同时，众多大模型推理优化工具框架也层出不穷，VLLM是一个开源的高效推理框架，专为加速大语言模型的推理过程而设计。VLLM利用自行研发的Page-Attentin机制[59]通过优化模型的计算图和内存访问，降低了推理过程中的延迟，并且能够有效提升模型在不同硬件平台上的推理性能。除此之外，SGLang、LMDeploy[60, 61]等大模型推理优化工具也各有各的特点，适配不同的设备的同时能为用户带来吞吐量的提升和并发度的控制。

未来由于大语言模型规模和能力的持续增长，推理优化将成为未来研究和应用的核心方向，推动大模型技术在实际场景中的广泛部署和应用

# 第三章 大模型训练和相关算法技术研究

在本章中，会介绍本研究涉及到的一些基本算法原理，包括但不限于Transformer模型架构分析，本研究实验所使用底座模型Qwen系列模型架构和训练过程分析，大模型微调算法分析包括强化学习算法，Deepseek-R1关键训练算法和流程解析。

## 3.1 Transformer模型架构分析

Transformer[21]由谷歌Vaswani等人于2017提出，作为近年来在自然语言处理（NLP）领域取得突破性进展的架构，其核心思想在于摒弃了传统循环神经网络（RNN）和卷积神经网络（CNN）的局部计算方式，转而采用全局自注意力机制（Self-Attention）来进行序列建模，解决序列到序列(Sequence-to-Sequence)任务。Transformer的基本架构分为两部分：编码器(Encoder)和解码器(Decoder)。如图3-1所示，左边为编码器部分，右边为解码器部分。编码器部分主要由个相同的层堆叠而成，而每一层都包含两个子层，分别为多头注意力(Multi-Head Attention, MHA)层和前馈网络(Feed-Forward Network, FNN)层，并采用相加和层归一化(Layer Normalization, LayerNorm)操作连接两个子层。解码器同样是由个相同的层堆叠而成，除了编码器的两层之外，又插入了一个掩码多头注意力层，用于将编码器的输出与解码器的输入相融合。Transformer模型在解码器部分的注意力机制上增加了上三角掩码矩阵，防止在模型训练过程中出现信息泄露的情况，保证模型在计算当前位置信息时不受后面位置信息的影响。

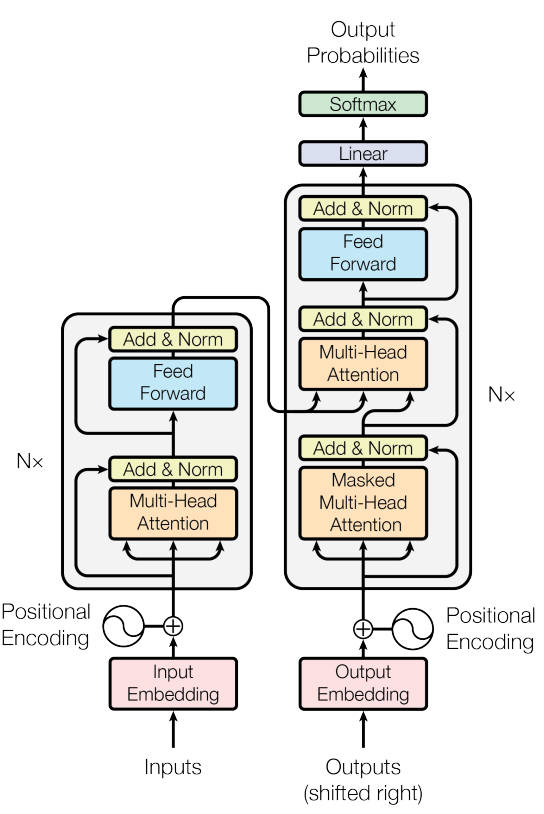


图3-1：Transformer模型架构示意图

自注意力机制是Transformers架构的核心，其目的是在输入序列中捕捉单词之间的依赖关系。在自注意力机制中，输入的每个词都会计算出一个加权和，这个加权和依赖于其他所有词的表示。一般可以看作将查询（Query）和一组键值对（Key Value Pair）映射到高维空间即对Value进行加权求和计算。自注意力的计算可以用以下公式表示：

(3-1)

在公式(3-1)中，是查询（Query）矩阵，是键（Key）矩阵是值（Value）矩阵，是键向量的维度。通过计算查询与键的点积并进行缩放，能够得到一个权重分布，再通过该分布对值进行加权求和，从而生成最终的输出表示。输入的每个单词首先通过三个不同的线性变换生成对应的查询、键和值向量。这些变换使得模型能够灵活地学习不同的表示。自注意力机制通过计算查询和键值对之间的相似度来为每个词分配一个权重，进而对值进行加权求和，得出最终的输出。通过这种方式，模型能够在一个序列中聚焦于与当前词最相关的其他词。

为了进一步增强模型的表达能力，Transformer引入了多头注意力机制。该机制将自注意力的计算分成多个注意力头，每个头独立地学习不同的表示，然后将所有头的输出拼接起来并进行线性变换。多头注意力的计算表示如公式(3-2)所示：

(3-2)

通过多个不同的线性变换矩阵  分别对输入的查询、键 、值  进行变换，得到不同头的注意力结果，然后将这些结果拼接起来(操作)，最后再通过输出线性变换矩阵进行变换得到最终的多头注意力输出 。 这样模型可以从不同子空间捕捉信息，更好处理序列复杂关系。

而由于Transformer模型没有使用递归或卷积结构[62]，它无法自动捕捉输入序列的顺序信息。因此，位置编码被引入以补充位置信息。位置编码是通过对输入序列的每个元素添加一个独特的向量来实现的，这个向量与元素在序列中的位置相关。常见的做法是使用正弦和余弦函数来生成位置编码，计算同时如下：

(3-3)

表示位置，是维度索引，是模型的维度。位置编码通过正弦和余弦函数为序列中的每个位置生成固定维度的向量，为模型引入位置信息 。

为了优化训练过程，Transformer采用了位置-Wise前馈网络来对每个位置的表示进行独立变换，这一过程的公式为:

(3-4)

在公式(3-4)中，x是输入向量，是权重矩阵， 是偏置向量。运算过程是先将输入与权重矩阵相乘再加上偏置，然后通过 激活函数即 ，最后再与权重矩阵相乘并加上偏置得到输出。

通过这些设计，Transformer成功地在NLP任务中提供了更高效、更强大的性能，尤其在处理长序列任务时表现突出。不过在后续的生成式大语言模型结构中，考虑到提高训练效率以及通过词表无需对输入进行编码操作，基本都只需要用到Transformer的解码器即Decoder-Only结构，生成输出序列也会更快。

## 3.2 Qwen系列大语言模型分析

Qwen系列模型作为本研究实验过程中的基座模型，在本节中会基于Qwen系列的大模型的技术细节进行分析和解读，在实验的设计中也会涉及到与Qwen类似的相关设计。

### 3.2.1 Qwen系列大模型架构和训练流程分析解读

自从2022年底ChatGPT爆火之后，国内各个互联网厂商都紧追不舍，纷纷开始自研大语言模型，并开放接口邀请用户进行测试和体验。2023年8月阿里巴巴通义千问团队开源70亿参数Qwen-7B，而后又分别在9月和11月开源了Qwen-14B以及Qwen-1.8B和Qwen-72B [32]。

在该系列中，Qwen模型的预训练数据主要涉及公共网络文档、百科全书、书籍、代码等，数据涉及多种语言但主要以中文和英文为主。为了保证预训练数据的质量，其制定了一套全面的预处理流程，最后仅仅保留了3万亿个Token的训练语料，具体流程总结如下：

* Web数据需要从HTML中提取文本内容，并采取语言识别工具确定语种。
* 通过重复数据删除技术增加数据的多样性，包括规范化后的精确匹配重复数据删除方法以及使用MinHash和LSH算法的模糊重复数据删除算法[63, 64]。
* 结合规则和机器学习的方法过滤低质量数据，即通过多个模型对内容进行评分，包括语言模型、文本质量评分模型及用于识别潜在冒犯性的模型。
* 在各种来源数据中手动采样并进行审查，以确保数据质量。
* 有选择性地对某些来源的数据进行采样，确保模型在各种高质量内容上进行训练。

Qwen模型在构建词表中，采用了BPE分词器，以cl100k为基础词库，增加了常用的中文字词和其它语言的词汇，同时把数字字符串拆成单个数字，最终将词表大小定位15.2万，对多语言非常友好，一般不需要拓展词表适配新语言。而在模型结构上，是大部分生成式语言模型会采用的Transformer的Decoder-Only结构，但其做出了以下的修改优化：

* 对于模型嵌入层和lm\_head层不进行权重共享，而是两个单独的权重。
* 采用RoPE位置编码，并选择使用FP32精确度的逆频率矩阵。
* 在QKV注意力曾中添加了偏差来增强模型的外推能力。
* 采用预归一化来提高训练稳定性，并将传统的归一化方法替换为RMSNorm。
* 采用SwiGLU激活函数来降低隐藏层的维度，提高训练效率。

在模型预训练过程中，Qwen模型采用Flash-Attention技术[33]提高训练速度，Flash-Attention旨在加速注意力计算并减少内存占用。其利用底层硬件的内存层次知识，例如GPU的内存层次结构，来提高计算速度和减少内存访问开销。 FlashAttention的核心原理是通过将输入分块并在每个块上执行注意力操作，从而减少对高带宽内存（HBM）的读写操作。具体而言，FlashAttention使用平铺和重计算等经典技术，将输入块从HBM加载到SRAM（快速缓存），在SRAM上执行注意力操作，并将结果更新回HBM。FlashAttention减少了内存读写量，从而实现了2-4倍的时钟时间加速。

由于Transformer模型的注意力机制在上下文长度上有很大的限制，随着上下文长度的增加，模型的计算成本和所需内存会成倍增加。而Qwen利用动态NTK感知插值，随着序列长度的增加动态缩放位置信息，LogN-Scaling 将查询和值的点积重新缩放，确保注意力值的熵随上下文长度增加而保持稳定以及通过窗口注意力机制将注意力限制在一个上下文窗口内，防止模型关注到太远的内容，在实际推理过程中可以将上下文长度扩展到1.6万Tokens。

在模型后训练阶段，应用了经典的SFT+PPO即有监督微调和人类偏好对齐的训练方法，先通过SFT有监督微调的方式，增强模型对复杂对话的处理和分析能力。在人类偏好对齐阶段，构造偏好数据集进行偏好模型训练(Preference Model Pretraining, PMP)得到奖励模型，再利用高质量偏好数据进行奖励模型微调。奖励模型是由同等参数量的Qwen模型+池化层构建，用特殊的句子结束标记映射作为模型奖励值。

### 3.2.2 Qwen2系列大模型架构和训练流程分析解读

2024年6月，通义千问团队正式开源发布Qwen2系列大模型，包括5个尺寸的预训练和指令微调模型, 包括 Qwen2-0.5B、Qwen2-1.5B、Qwen2-7B、Qwen2-57B-A14B 以及Qwen2-72B，在中文英语的基础上，训练数据中增加了 **27**种语言相关的高质量数据，同时其代码和数学能力显著提升，增大了上下文长度支持，最高达到 128K tokens[10]，其模型架构图如图3-2所示。

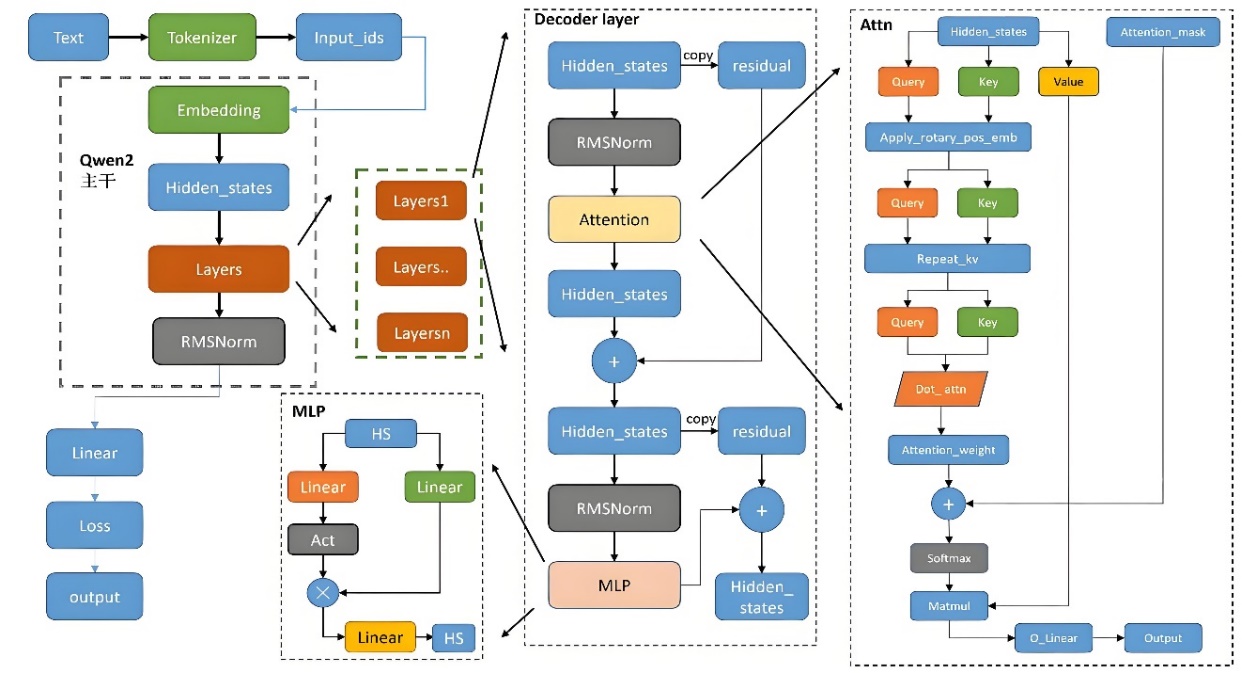


图3-2：Qwen2模型结构图

Qwen2 Dense 本质是基于 Transformer 架构开发训练的模型， 与前一代 Qwen 模型相比，Qwen2 Dense 在多个方面进行了改进，极大提高了模型的性能和效率。在Qwen2的预训练过程中，核心关注点之一是数据集的质量提升与有效处理扩展上下文长度的方法。首先，数据质量的优化是通过改进过滤算法实现的，结合了额外的启发式规则和基于模型的筛选策略。具体来说，Qwen模型被用来筛选和去除低质量数据，并且通过生成高质量预训练数据进一步强化了模型的训练质量。此外，Qwen2还增加了更大规模的高质量代码、数学及多语言数据集，从而提升了模型在这些特定领域的表现。新收集的数据集支持约30种语言，包括英语、中文、西班牙语、法语、德语、阿拉伯语、俄语、韩语、日语、泰语和越南语等，显著扩展了模型的多语言能力。

为了优化模型的学习过程并使其更接近人类的学习模式，Qwen2在较小规模的模型上进行了实验，探索不同领域和来源数据的混合策略。这一过程促使预训练数据的规模从Qwen1.5版本的3万亿tokens扩展至7万亿tokens。尽管进一步放宽质量阈值后生成了一个包含12万亿tokens的数据集，但在该数据集上训练的模型未能展示出优于7万亿tokens数据集的效果。除了Qwen2-0.5B模型外，所有其他Qwen2 Dense模型都是基于超过7万亿tokens的大规模数据集进行预训练的，其中Qwen2-0.5B则使用了12万亿tokens的数据集进行训练。同时，其MoE模型也遵循了增益再利用原则，额外使用了4.5万亿tokens进行预训练。

Qwen2 采用了分组查询注意力(Grouped Query Attention, GQA) [65]，替代了传统的 MHA（Multi-Head Attention, MHA），是对MHA和 Multi-Query Attention (MQA) 的扩展。GQA 通过优化 KV Cache 使用，在推理过程中显著提高了吞吐量。使用G个组可以减少存储每个头的键和值所需的内存开销，特别是在具有大的上下文窗口或批次大小的情况下，GQA提供了对模型质量和效率的细致控制，图3-3形象地展示了GQA、GHA、MQA的特点。

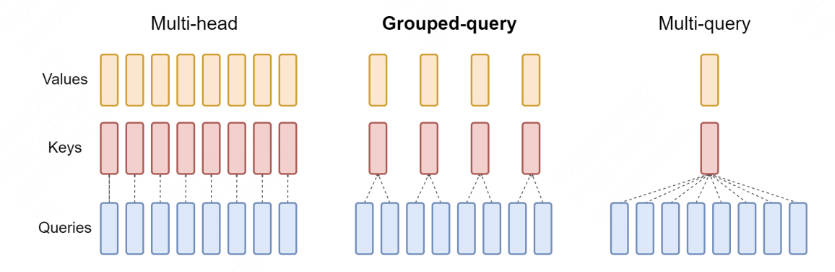


图3-3 MHA,MQA,MGA对比图

GQA的计算公式表示如下：

(3-4)

GQA 提供了 MHA 到 MQA 的自然过渡。当  时即为MHA；当时就是MQA；当时，它只将 KV Cache 压缩到 ，压缩率不如MQA，但同时也提供了更大的自由度，效果上更有保证。

在预训练的最后阶段，Qwen2 将上下文长度从 4,096 个 token 增加到了32,768 个token。这一扩展通过引入大量高质量的长文本数据得到了补充。为了配合这些增强，Qwen2 将 RoPE的频率从 10,000 调整到了 1,000,000，以优化长上下文场景下的性能。为了扩大模型的上下文窗口即充分利用模型的外推潜力，Qwen2引入了分块注意力机制（Dual Chunk Attention，DCA）[66]，将长序列划分为可管理的较短块，从而有效地提升了模型处理长序列的能力。此方法可以优化计算资源的分配，允许模型在不牺牲上下文信息的前提下，更加高效地处理大规模输入数据。DCA 通常分为两个阶段：局部注意力和全局注意力。在局部注意力阶段，输入序列被分成若干个块（chunks），每个块内的元素之间进行注意力计算。这种方式可以有效减少计算复杂度，因为注意力计算只在较小的块内进行。在全局注意力阶段，每个块的表示被用来计算全局注意力。具体来说，每个块的表示可以是该块内所有元素的平均值或最大值。然后，这些块表示之间进行注意力计算，从而捕捉全局信息。

此外，Qwen2还采用了YARN（Yet Another Rescaling Method）方法[67]，以此来重新调整注意力权重，以改善模型对不同长度序列的处理能力。使得模型能够更好地处理长序列和短序列，避免在处理长序列时注意力权重过于分散，从而提升模型的整体表现。YARN 的核心思想是对注意力权重进行重新缩放，以便在处理不同长度的序列时，注意力机制能够更有效地聚焦于重要的信息。具体来说，YARN 通过以下步骤实现这一目标：

* 计算原始注意力权重：首先，计算原始的注意力权重矩阵
* 计算缩放因子：根据序列的长度，计算一个缩放因子。这个因子通常是一个与序列长度相关的函数，用于调整注意力权重的分布。
* 应用缩放因子：将计算得到的缩放因子应用到原始的注意力权重矩阵上，重新调整权重的分布。
* 归一化：对调整后的注意力权重进行归一化处理，确保权重的总和为1。

在Qwen2的后训练阶段，采用了与传统方法显著不同的策略，重点在于实现可扩展的对齐方法，并尽可能减少对大量人类监督的依赖。传统方法通常严重依赖大量人类注释，而Qwen2则通过优化数据获取流程，旨在在最大化数据质量和可靠性的同时，最小化人类标注的需求。具体而言，通义千问团队深入研究了如何有效地获取高质量的指令和偏好数据，以自监督式微调（SFT）和基于人类反馈的强化学习策略（RLHF）[68]。

训练数据的构建流程采用了两步策略：首先，通过协作数据注释，从大规模指令语料库中提取出广泛且多样化的高质量指令集合。这些指令经过精心筛选，确保能够覆盖多个领域和任务类型，从而为模型提供了多样性和深度的训练素材。随后，为了提升模型对复杂任务的处理能力，这些指令数据被系统性地增强，使其具备更高的复杂性和挑战性

在Qwen2的训练过程中，为了提高数据质量和任务执行效果，采用了几种创新的数据处理和优化策略。

首先，对于数学类任务，通义千问团队应用了拒绝采样（Rejection Sampling）[69]方法。这一策略通过为每个指令生成多个推理路径，从中筛选出既能给出正确答案，又被模型认为合理的路径作为示范数据。通过对比正确和错误的推理路径，不仅可以提高数据的质量，还能够帮助生成更加精确的偏好数据，从而进一步优化模型在数学任务中的表现。

对于编程类任务，采取了执行反馈（Execution Feedback）策略。通过利用大模型（LLM）生成解决方案及其相关的测试用例，并对这些解决方案进行编译和执行，Qwen2可以有效地验证解决方案的正确性。测试用例的通过与否，成为评估解决方案有效性的重要依据。

在数据再利用方面，Qwen2特别针对文学写作任务提出了一种有效解决方案。由于文学创作任务的复杂性，传统的注释者难以给出高质量的回答，Qwen2通过收集来自公共领域的高质量文学作品，并结合大模型生成不同细节程度的指令，极大地提升了训练数据的质量。这些指令与原始作品配对，作为示范数据，确保了生成的文学内容既富有创意又符合文本要求。例如，在生成角色扮演数据时，Qwen2通过从维基百科等知识库提取详细的人物档案，并指引大模型生成相应的指令与响应，确保了人物形象的完整性与一致性。

Qwen2在SFT训练过程中收集了一个庞大的、包含超过50万个示例的指令数据集，涵盖了广泛的技能领域，包括指令遵循、编程、数学、逻辑推理、角色扮演、多语言能力以及安全性等。这一数据集为模型的多任务学习提供了丰富的素材，确保了Qwen2在多个领域具备强大的处理能力。在模型的微调阶段，Qwen2进行了两轮监督式微调（SFT），每轮的序列长度设置为32,768，这样可以有效提升模型对长序列的处理能力，增强其上下文理解和信息保持能力。为了优化学习过程，采取了动态调整的学习率策略，初始学习率为7 × 10⁻⁶，并在训练过程中逐渐降低至7 × 10⁻⁷，旨在训练的后期精细调整模型的参数，以便更好地捕捉数据中的复杂模式。为了应对过拟合问题，Qwen2引入了0.1的权重衰减，帮助正则化模型，避免其在训练数据上过度拟合。此外，还采用了梯度裁剪技术，将梯度的最大值限制在1.0，从而有效防止了梯度爆炸现象，保证了训练过程的稳定性。

Qwen2在RLHF（基于强化学习的反馈优化）训练方案中，采用了两阶段的训练流程：离线训练和在线训练，分别针对模型的初步优化和实时性能提升进行精细化调整。在离线训练阶段，Qwen2首先使用预处理好的偏好数据集，通过DPO（Differentiable Preference Optimization）[70]方法，最大化正向响应 和负向响应 之间的差异。DPO作为一种优化方法，通过对比不同响应的质量，能够更好地学习到用户偏好和任务目标，从而提高模型生成高质量响应的能力。进入在线训练阶段后，Qwen2进一步引入了基于奖励模型（Reward Model）的实时反馈机制。此阶段，模型在每一步生成的响应中，从当前策略模型中采样多个候选响应，奖励模型（reward model）会根据给定的标准对这些响应进行评估，选择出最优和最差的响应形成偏好对（preference pair）。然后，通过这些偏好对进一步优化模型的输出，以提高响应质量和任务适应性。为了提高训练效率并减轻对齐成本，Qwen2采用了在线合并优化器（Online Merging Optimizer）。该优化器通过合并和优化在线学习中的多个响应反馈，减少了在训练过程中对齐偏好数据的复杂度和计算负担，从而加速了训练过程并提升了模型的响应生成效率。

### 3.2.3 Qwen2.5系列大模型架构和训练流程分析解读

2024年9月通义千问团队重磅发布了 Qwen2.5系列，包括Qwen2.5-Math和Qwen2.5-Coder以及多模态模型Qwen2.5-VL [18, 45, 71, 72]，在模型结构层面，Qwen2.5与Qwen2系列模型基本保持一致，主要优化点还是在于预训练和后处理两个阶段的改进，在高质量预训练数据集的基础上，通过多阶段强化学习等技术进行后处理，提高了人类偏好、长文本生成、结构数据分析和指令遵循等方面的能力。

在Qwen2.5的预训练流程中，为了进行高质量数据筛选，引入Qwen2-72B-Instruct作为数据质量评估工具，对训练样本进行了全面的打分与筛选。该方法基于多维度的分析框架，有效评估并优化了训练数据的质量。特别是在多语言数据处理的场景下，Qwen2-72B-Instruct展现出了优于传统方法的显著优势。在Qwen2的基础上，Qwen2.5版本进一步引入大量高质量的数学和编程数据，加强了数学推理和代码生成的能力。在合成的数据方面，基于Qwen2-72B-Instruct和Qwen2-Math-72B-Instruct生成高质量合成数据，并使用专有奖励模型和Qwen2-Math-RM-72B模型进行严格过滤，增强了模型在数学、代码和知识领域的能力。Qwen2.5的预训练也参考了Scaling Law[37]，基于得到的最优超参数预测结果，将最终损失建模为模型架构和训练数据规模的函数。通过这个函数来确定不同模型架构的最优超参数。

而在后训练阶段，在监督微调SFT过程，构建了一个超过100万个样本的大规模数据集，主要聚焦于弥补Qwen2在多个领域的局限性。该数据集涵盖了多个关键任务，包括长序列生成、数学推理、编程能力、指令遵循、结构化数据理解、逻辑推理、跨语言转移以及系统指令的稳健性等。

在离线强化学习（Offline RL）阶段，基于执行反馈与答案匹配等策略确保了响应质量。利用SFT微调后的模型对新查询进行重新采样，并构建了一个包含约150,000个训练对的数据集。为了确保训练信号的可靠性与准确性，采用了人类和自动化审查的多重审核过程，对模型输出进行精细化校准。

在在线强化学习（Online RL）过程中，为了进一步提高模型在实际应用中的响应质量，开发了一种严格遵循标注标准的奖励模型，确保生成的响应符合伦理要求并满足用户需求。这些标准涵盖了真实性、有用性、简洁性、相关性、无害性和去偏性等多个维度。通过Group Relative Policy Optimization（GRPO）方法进行训练，根据奖励模型评估的响应分数方差确定查询处理顺序，提高了模型生成高质量响应的能力，在Qwen2.5-Math和Qwen2.5-Coder系列这一需要深度逻辑思考推理能力的模型中表现尤为突出，这也是本研究实验过程中强化学习的重要过程。Qwen-Math系列的训练流程如图3-4[45]所示：

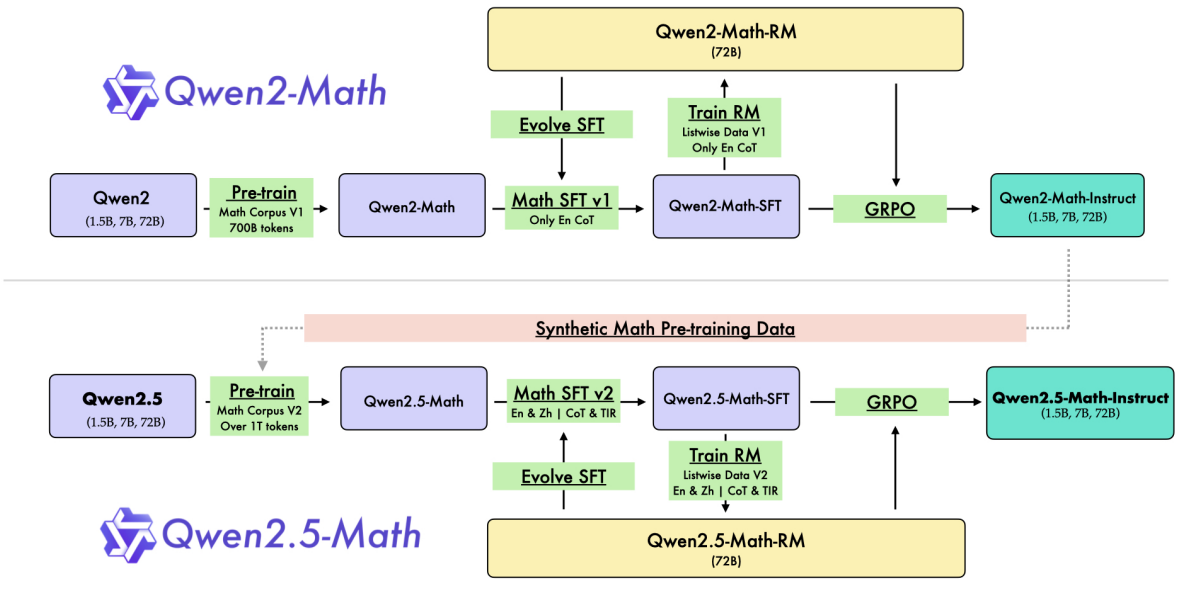


图3-4 Qwen2/Qwen2.5-Math模型训练流程

由图3-4可见，Qwen2-Math到Qwen2.5-Math系列最关键的步骤在于预训练语料规模以及质量的提升，然后是SFT过程中CoT和TIR两阶段的通过奖励模型筛选数据后进行高质量数据训练，最后再利用相同的强化学习算法GRPO进一步提高模型的性能。

## 3.3 有监督微调算法技术解析

### 3.3.1 SFT简介

有监督微调（Supervised Fine-Tuning，SFT）[73]，指采用预先训练好的网络模型，并针对专门任务在少量的监督数据上对其进行重新训练的技术。简单来说，SFT就是在特定任务的数据集上对预训练模型进行微调，以增强模型在特定领域或任务上的表现。在当前大模型应用领域，SFT仍是主流的训练方式，与传统的从头开始训练不同，SFT依赖于先前的预训练知识，通过数据的标注样本来优化模型参数，使模型更适应实际应用。可以理解为老师针对学生的薄弱环节进行专项辅导。它利用少量高质量的专家推理数据，让模型学习如何模仿专家的解题思路和方法，像指令遵循、问答以及思维链等能力都是在这个阶段培养的。

同时SFT在训练效率和资源消耗以及性能表现上达到了一定的平衡，相比于预训练所需要的大量语料数据清洗以及强化学习过程中所需的高额算力，SFT更适合普通企业和个人用户对大模型进行训练微调，达到目标效果。

进行SFT的基本流程大致如下：

1. 预训练模型初始化，在进行有监督微调前，首先需要一个经过大规模数据预训练的模型（如Qwen系列、LLaMA系列等）或者是经过大规模数据SFT过后得到的指令大模型（Instruct系列）。这些模型已经从大量的无标注数据中学习到了一般性知识，或者已经从大量多领域数据学习到了多个关键任务的解决思路，但它并没有针对用户所需要的特定任务进行优化。
2. 任务特定数据集准备，有监督微调要求对模型进行特定任务或特定领域的数据训练，如特定格式输出、医疗领域、法律领域等。训练集通常包含输入数据（如文本）和对应的目标输出（标签或答案）组成的问答对。
3. 微调目标函数的设计，有监督微调通过最小化损失函数来调整预训练模型的参数。常见的损失函数包括交叉熵损失（用于分类问题）或均方误差（用于回归任务）。微调过程的目标是通过标注样本来优化模型在该任务上的表现。

### 3.3.2 SFT数学表示解析

SFT的数学表示可以理解为：给定一个任务和其对应的训练数据集公式(3-5)所示， 表示输入数据，是目标输出，为样本数量，预训练模型的参数为，通过微调得到的最终参数为。

(3-5)

SFT的目标是通过最小化训练数据上的损失函数来优化模型参数 ，如公式(3-6)所示：

(3-6)

在公式(3-6)中，是模型在参数下对输入的预测，是损失函数，通常采用交叉熵定义为：

(3-7)

在公式(3-7)中，表示类别总数， 是样本真实标签中类别 的取值（通常为 0 或 1 ，表示样本是否属于该类别）, 是模型  对输入 预测为类别 的概率。

在SFT中，还会涉及到一些优化策略，从而在模型微调的过程中保证模型训练的稳定性，首先是学习率调度策略(Learning Rate Scheduling)，在微调过程中，学习率的选择至关重要。通常采用逐步衰减策略，从一个较高的学习率开始，随着训练的进行逐渐减小学习率。这有助于稳定训练过程，避免过早地跳出最优解。目前大模型领域常用的优化策略还是以混合精度训练 (Mixed Precision Training) [74]结合学习率预热(Learning Rate Warmup) [75]为主。

为了防止模型训练出现过拟合的情况，会使用权重衰减（Weight Decay）优化策略，通常会在损失函数中加入正则化项。最常见的正则化方式是L2正则化，或者称为权重衰减。其形式为：

(3-8)

其中，是正则化系数，控制着模型复杂度与拟合能力之间的平衡。

在训练过程中，特别是在使用非常深的网络时，可能会遇到梯度爆炸的问题。为了解决这一问题，梯度裁剪（Gradient Clipping）技术被引入，即在每次反向传播时，若梯度的范数超过某个阈值，则将梯度进行缩放，数学公式表示如公式(3-9)所示：

(3-9)

其中，是裁剪的阈值，通常设定为一个较小的常数值。

## 3.4 大模型强化学习微调算法解析

### 3.4.1 ‌近端策略优化算法解析

在大语言模型（LLM）领域，强化学习（Reinforcement Learning, RL）[6]被广泛应用于提升模型在具体任务中的表现，尤其是在生成任务、对话系统和任务驱动的生成任务中，可以显著提升模型生成的文本质量。近端策略优化（Proximal Policy Optimization, PPO）[3] [76]是当前强化学习中常用的算法之一，其稳定性和高效性使得它成为了强化学习在大语言模型中的主流方法之一。但是由于PPO过程过于消耗GPU资源，本研究所进行的实验并没有涉及PPO相关的实验组而是设置了GRPO [20]实验组，但在本小节中，仍旧需要阐述解读PPO的训练流程，目的是为了理解后续GPRO算法的优化点，因为本质上GRPO也是PPO的某种变体算法。

针对大模型领域的RLHF，实际上可以理解为模型先试错再学习的过程，用通俗的话来形容，我们会设置一个老师的角色，给出有趣的问题，而模型则会像小学生一样，不断尝试给出答案。模型会对着黑板写下它的答案，有时候是正确的，有时候会有错误。老师会仔细检查每一个答案，如果它表现得好，就会给予它高声赞扬；如果它表现不佳，我们则会给予它耐心的指导和反馈，帮助它不断改进，直到达到令人满意的水平。

PPO训练过程涉及到四个模型，分别是：

1. Actor Model：演员模型，是需要训练的目标语言模型
2. Critic Model：评论家模型，它的作用是计算预期收益
3. Reward Model：奖励模型，它的作用是计算实际收益
4. Reference Model：参考模型，它的作用是给语言模型增加一些“约束”，防止语言模型训练偏离预期或者训歪，使模型的回答结果与预期结果的分布相近

一般来说，为了节约训练过程的资源消耗，Actor与Reference的初始化模型就是需要训练的底座模型，Reward与Critic的初始化模型就是Reward模型，其中Actor与Critic在后续训练中需要更新参数，而Reward与Reference Model的参数是冻结的，这样可以减少计算量。

在大模型领域中，PPO的训练过程如图3-5所示，图中的a值是一个控制阈值，通常设定为一个较小的常数值。

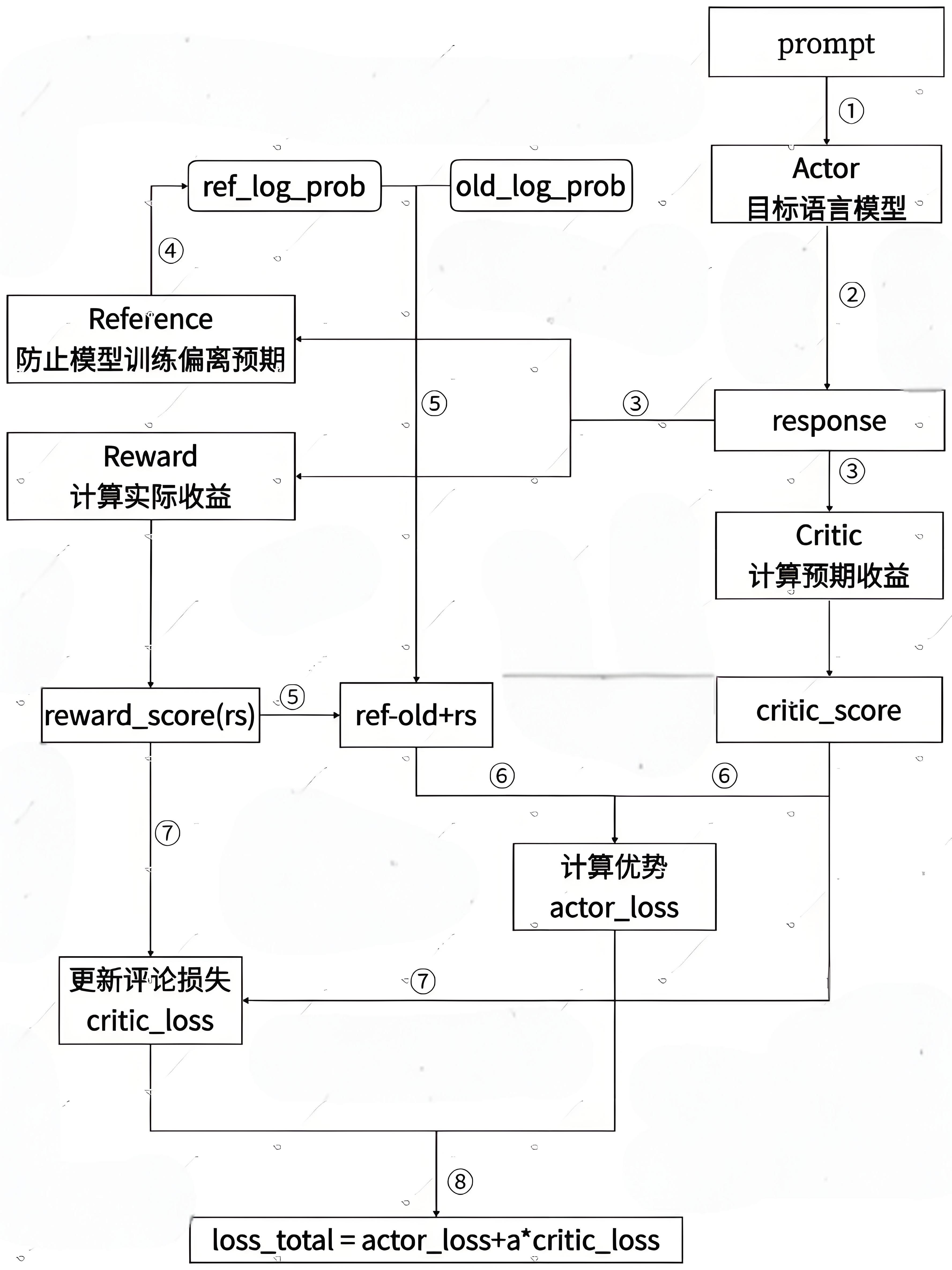


图3-4 大模型PPO训练流程简化图

**第一步**，给演员模型(Actor Model)输入问题Prompt，Actor Model会有两个输出：生成的回答response；还有给出上一部分的文字，每生成下一个字的对数概率，记为old\_log\_prob，是一个张量，长度就是response的长度。这一步本质上是**采样**，采样就是学生回答问题的过程，是模型根据提示（prompt）输出回答（response）的过程，或者说是模型自行生产训练数据的过程，即旧的Actor Model对prompt进行token采样的过程：

* 计算response的第1个token的概率分布，然后从概率分布中采样出第1个token的值
* 根据第1个token，计算response的第2个token的概率分布，然后从概率分布中采样出第2个token值
* ……
* 根据前个token，计算response的第个token的概率分布，然后从概率分布中采样出第个token

最后结合评论家模型(Critic Model)，一共会得到三个输出即模型的回答response，response中token的对数概率old\_log\_prob，Critic Model对每次生成token时预估的收益critic\_score，如图3-6所示：

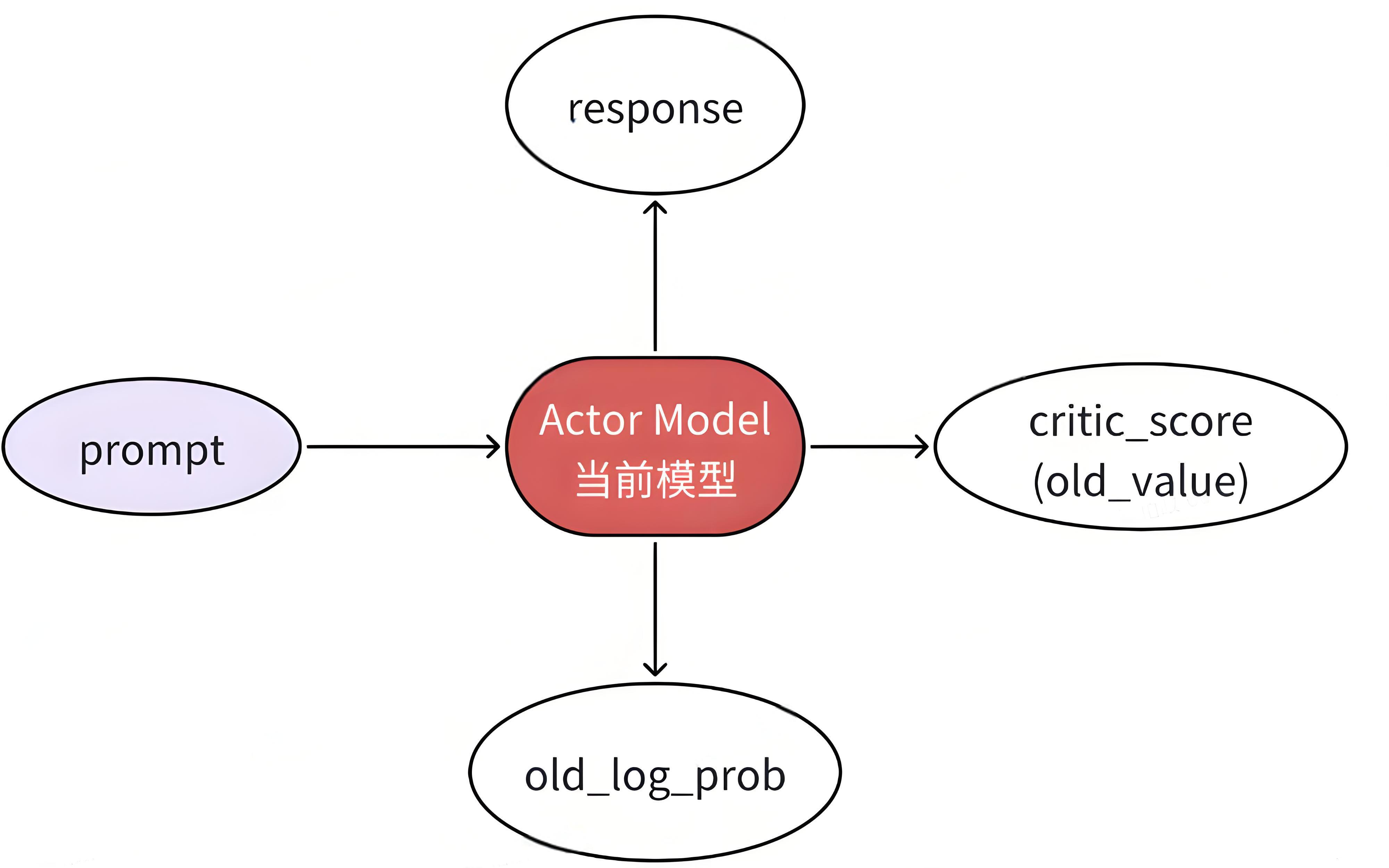


图3-6 PPO过程中的采样流程简化图

**第二步**，就是强化学习中关键的反馈步骤也就是**优势奖励**，反馈就是老师检查答案的过程，是奖励模型（Reward Model）给response打分的过程，或者说是奖励模型给训练数据标上值即奖励值的过程。在这一个过程中，我们会将上一步中的prompt和模型生成的response输入给参考模型(Reference Model)，会得出在原先模型下输出该句话的张量分布，记为ref\_log\_prob，这也是一个张量，长度为response的长度。同时我们还会把相同的输入给到奖励模型(Reward Model)，Reward Model会输出一个匹配分数，作为模型的实际收益，记作，但是，需要注意的是，并不是最终对于模型的奖励，它和最终的奖励rewards之间还隔着一个根据用户需求设置的奖励函数(Reward Function)，PPO的奖励过程的具体流程如图3-7所示：

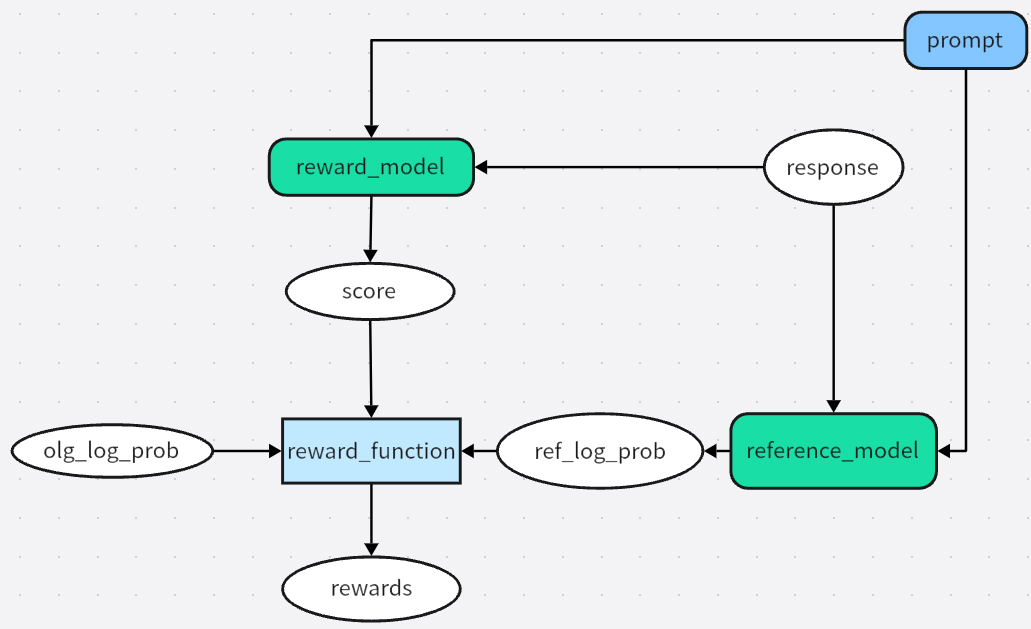


图3-7 PPO过程中的奖励流程简化图

一般来说，我们在大模型PPO训练过程中，会采取过程合理性奖励如特殊格式、逻辑链的正确性等和结果正确性奖励即最终输出的答案是否符合预期。

过程合理性奖励的计算方式是这样的。Reference Model拿到prompt，然后给old Actor Model生成的response的每个token计算对数概率，得到一个二维张量ref\_log\_prob作为概率分布。由此来看old\_Actor\_Model为第个response生成的第个token，生成这个token应该获得的奖励可以简单写为公式(3-9)：

(3-9)

公式(3-9)可以这样理解ref\_log\_prob[i, j]越高，Reference Model越认可old\_Actor\_Model的输出，说明old Actor Model更符合输出的预期，因此该获得更高的奖励值，old\_log\_prob[i, j]越高，old\_policy获得的奖励反而越低。old\_log\_prob[i, j]又可以理解为一个正则项，保证了概率分布的多样性。

公式(3-9)本质上可以理解为 KL散度（ KL Penalty）的简化版本。实际上完全可以改成计算两个token的概率分布的KL散度。在PPO过程中，KL散度用于衡量当前策略与旧策略之间的差异。KL散度的计算有助于避免策略发生过大的变化，并且可以作为一个附加的约束，通过监控KL散度，可以确保当前策略不会与旧策略相差过大，从而避免过度更新，进一步稳定训练过程。KL散度的标准数学计算公式如下：

(3-10)

其中，表示旧的策略分布，表示新的策略分布，是 t 时刻的状态，是在状态下采取的动作。

最终，我们将过程合理性奖励和结果正确性奖励合并起来，就得到了最终奖励的计算方式。注意，我们只在最后一个token上，本质上是期望的答案上应用结果正确性奖励（Reward Model的输出）。由此可以得到一个简化版的奖励计算公式，即第个response生成的第个token的奖励公式如公式(3-11)所示：

(3-11)

在这里需要注意，训练过程只对模型的response计算奖励。另外在整个反馈阶段，Reward Model和Reference Model是被冻结的，不更新参数的。一旦给出了response对应的reward，就完成了反馈阶段。可以简单理解为学生的答案得到了老师的反馈，此时学生就可以通过相关的反馈了解其具体表现如何，但获得反馈并不是结束，而是新的开始。正如学生需要用这些反馈来进行复习和改进一样，模型也需要通过学习阶段来优化其性能和预测能力。

由此PPO过程进入到了**第三步——强化优势动作**，这一步是大模型PPO过程的重点，因为模型基于第二步得到的反馈进行自我学习和优化的强化优势动作的过程，即开始进行权重的更新迭代，从而让自己的输出更加符合预期。在这里，优势本质上可以被定义为实际获得的收益超出预期的程度。强化优势动作可以形象地理解为：

* 当优势大于0时，概率越大，loss越小；因此优化器会通过增大概率（即强化优势动作）来减小loss。
* 当优势小于0时，概率越小，loss越小；因此优化器会通过减小概率（即弱化劣势动作）来减小loss。

如图3-8所示:

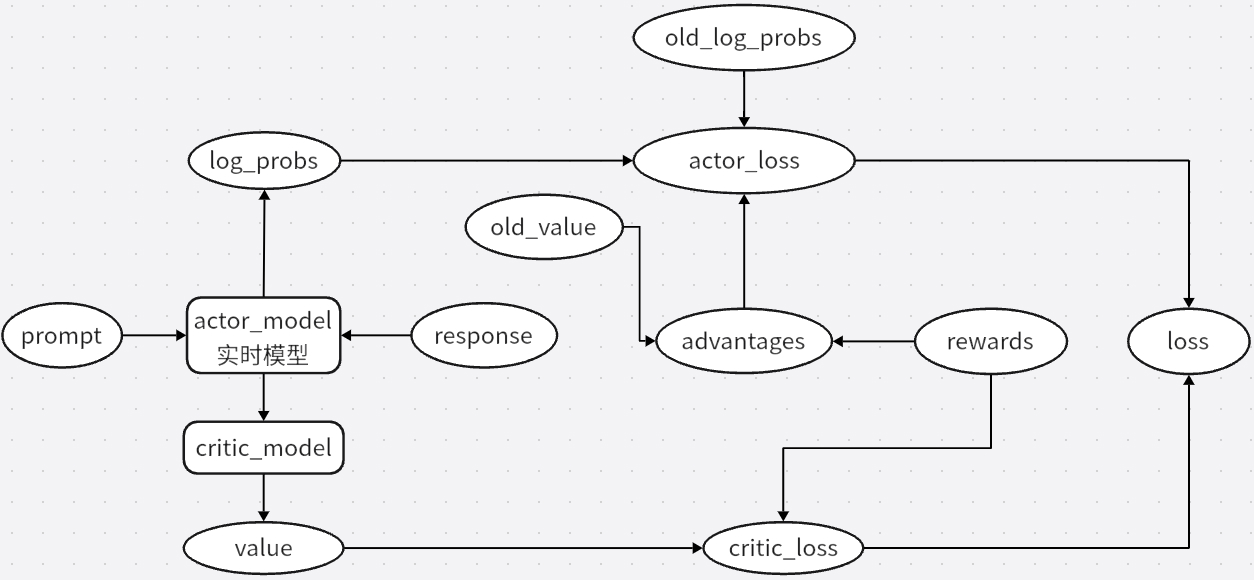


图3-8 PPO过程中的学习(强化优势动作)简化流程图

而对于语言模型而言，为第个response生成第个token的实际收益就是：从生成第个token开始到生成第个token为止，所能获得的所有奖励的总和。假设用return来表示实际收益，它的计算方式可以形象地简化为公式(3-12)，如下所示：

(3-12)

本质上，公式(3-12)是PPO过程中的优势函数，来衡量在某个状态下采取某个动作的优越性。如果用标准数学语言表达，优势函数可以通过广义优势估计（Generalized Advantage Estimation，GAE）来计算。GAE有助于减小方差并提高策略更新的稳定性。其数学公式如公式(3-13) 所示：

(3-13)

其中， 表示时间步 的优势估计值；是时间步   的时间差分误差；是折扣因子，用于衡量未来奖励的现值； 是控制偏差 - 方差权衡的参数。

基于上述推导，根据old Actor Model的概率分布输出和当前的Actor Model的概率分布输出，第个response生成第个token时，可以将Actor Model优化时使用的损失函数形象地表达为：

(3-14)

由于Critic Model也需要更新，就设计一个损失函数来衡量评论家预期收益和真实收益之间的差距，可以形象地表示（暂时不考虑GAE）为：

(3-15)

最终优化Actor Model时用的损失函数是和的加权和。

至此，PPO整体的训练流程的形象化推导结束。

然而，在真正落地的大模型PPO训练方案中，往往会使用PPO算法有的两个主要的变种：近端策略优化惩罚（PPO-penalty）和近端策略优化裁剪（PPO-clip），使用这两种方法的原因在于当策略模型Actor Model是深度模型时，沿着策略梯度更新参数，很有可能由于步长太长，策略突然显著变差，进而影响训练效果，大模型显然是参数量极大的深度模型，因此为了避免新旧两个策略分布相差大即意味着避免步长过大，利用信任区域策略优化(Trust Region Policy Opimization，TRPO) [77]的思想，会考虑在更新时找到一块信任区域(trust region)，在这个区域上更新策略时能够得到安全性保证。故而在大模型PPO训练过程中，引入惩罚约束。

近端策略优化惩罚PPO-penalty的流程如下：

1. 首先明确PPO需要优化的目标函数：

(3-16)

目标是最大化公式 (3-16)

1. 接下来，先初始化一个策略的参数, 在每一个迭代里面，我们用前一个训练的迭代得到的Actor Model的参数，与环境交互，采样到大量状态-动作对， 根据 交互的结果估测
2. 由于目标函数牵涉到重要性采样，而在做重要性采样的时候， 与即两个策略分布不能相差太多，所以需要在训练的时候加个约束，类似于正则化项，本质上是和 输出动作的 KL散度，可用于衡量和 的相似程度，在预期中，我们希望和 越相似越好 所以需要最后使用 PPO 的优化公式：

(3-17)

其中：是 PPO 算法的目标函数，用于指导策略参数的更新。 是在策略 基础上关于参数的一个原始目标函数，通常衡量策略在环境中获得的期望回报等收益。是一个超参数，用于权衡原始目标函数和 KL 散度项的重要性，该参数是可以动态调整的，故又称之为自适应KL惩罚(adaptive KL penalty)。是当前策略参数和旧策略参数 之间的 KL 散度，它衡量了两个策略分布之间的差异程度。

1. 由此我们得到PPO-penalty的计算公式：

- (3-18)

近端策略优化裁剪（PPO-clip）的计算公式本质上类似，但使用了一个裁剪函数(可视为在的控制下对模型的策略学习步长进行一定的裁剪)，在此由于篇幅问题，不做更多赘述，：

其中，(3-19)

### 3.4.2 直接优化策略算法解析

从大模型训练领域的发展来看，RLHF (Reinforcement Learning from Human Feedback) 即以强化学习方式依据人类反馈优化语言模型是大模型训练的标准范式，本质上可以理解为SFT+Reward Model Training+PPO，但从上文中的推导不难发现，其缺点也很明显：训练流程繁琐、算法复杂、超参数多和计算量大，所以大部分企业都是以SFT微调为主要的模型训练方式用于节省成本，不过，RLHF后续替代方案这几年来层出不穷，直接优化策略算法（Direct Preference Optimization，DPO）是一种非常高效的强化学习算法[70]。它巧妙地绕过了构建奖励模型和强化学习这两个的繁琐过程，直接通过偏好数据进行微调，效果简单粗暴，在使模型输出更符合人类偏好的同时，极大地缩短了训练时间和难度。

DPO节约资源的实质在于需要的数据与SFT本质上是一致的，只需要增加一列rejected数据，但其实都是经过人工排序后的QA语料对，可以理解为在数据准备阶段就已经提前告知模型哪个回复是更好的，哪个回复是更坏的。其巧妙地将reward model和强化学习两个步骤合并，使得训练更加的快速高效，在它的训练过程中Reference Model参数固定，只对目标语言模型Actor Model进行参数更新，调试更加简单。而与常规PPO主要不同就是损失函数的设计：

(3-20)

在公式(3-20)中， 是偏好数据对中好的回答 (chosen)， 则是偏好数据对中坏的回答(rejected)， 是当给定输入为x时，当前策略生成好的答案的概率， 是当给定输入为x时，原始（reference）策略生成好的答案的概率，当 越大时，整体的Loss就越小，由此可以看出， DPO 期望最大化的就是奖励模型对chosen数据和rejected数据的差值，从而来达到使模型的回答更偏向于人类排序靠前回答的目标。

上述DPO推导结果中看似非常完美，但实际使用过程中与PPO优化算法仍有差距，主要是因为DPO的训练目标会导致过拟合，当优化策略为零，那么就可以使得偏好概率变的很大，整体的损失很小。简单的理解就是模型可以通过随机乱回复的方法下就能使得Loss降低，没有像PPO损失函数考虑了结果整体的分值，因为PPO的要求是如果除非采样能拿到好的结果，不然不会更新参数，DPO本质上会强制性的更新参数，从而导致如果数据质量不佳很容易导致模型训练扭曲的结果，虽然训练指标看似正常，但其实达不到相关预期。在本研究中，本人也做了相关的简单实验探讨，结果就是确实针对部分需要模型核心能力的任务，DPO的效果往往不尽人意，但是如果只是需要输出特定格式或者特殊回复，DPO会有不错的效果。

## 3.5 Deepseek-R1 训练策略和关键算法解析

2025年01月20日，Deepseek-AI 团队正式发布推理大模型（RLLM）DeepSeek-R1，并同步开源模型权重[19]，其主要研究成果可以列举如下：

* 开源 DeepSeek-R1 推理大模型，通过高质量数据SFT冷启动以及GRPO的RL训练后，在低成本的情况下，与OpenAI公司发布的闭源SOTA推理大模型 O1 性能相近。‍‍
* 开源 DeepSeek-R1-Zero，对于拥有一定能力的DeepSeek-V3 [78]模型直接 RL，而不是使用 SFT+PPO的范式。
* 开源用 R1 数据蒸馏的 Qwen、Llama 系列小模型，蒸馏模型超过 O1-mini 和 QWQ [79]。

自从O1系列模型推出后，各大厂商均在研发自己的推理模型，一些研究尝试使用基于过程的奖励模型（Process Reward Model，PRM）、强化学习和搜索算法（MCTS）来解决推理问题，但没有达到 O1 系列模型的通用推理性能水平。Deepseek-AI团队在众多厂商中脱颖而出，探索并证明可以通过纯强化学习来让 LLM 自主发展推理能力，而无需依赖 SFT 数据提前做微调提高模型的推理能力和知识范围。

DeepSeek-R1-Zero，就是使用以Deepseek-V3-671B模型进行纯强化学习得到的推理模型，也是证明Deepseek-AI 团队选择的这条道路是正确的，即无需准备相关的 SFT高质量数据，仅通过 RL 即可激励模型自主学会长链推理和反思等能力。此模型的训练过程和结果正是本研究第二阶段实验希望复现的结果。

为了训练 DeepSeek-R1-Zero，其团队采用了一种基于规则的奖励系统，该系统主要由两种奖励组成：

* 准确率奖励：准确率奖励模型评估响应是否正确。例如，在具有确定性结果的数学问题中，模型需要以指定的格式提供最终答案，从而能够通过基于规则的验证来可靠地确认正确性。同样，对于 LeetCode 问题，可以使用编译器根据预定义的测试用例生成反馈。
* 格式奖励: 除了准确性奖励模型，还采用了一种格式奖励模型，要求模型将其思考过程放在<think> 和<\think>标签之间。

上述的两个规则其实就是Reward Model，训练 DeepSeek-R1-Zero 时没有使用结果奖励模型或者过程奖励奖励模型，这意味着减少了一个外部Reward Model的显存占用，只需要利用限定的规则构建Reward Function针对不同类型的数据设置不同的奖励分数即可。不过，如何针对难度不同、语言不同、题目类型设置合理的加权奖励得分，也是一个值得探究的问题。

通过结果正确性判定的方式：对于数学题、编程题等有客观正确答案的任务，可以把最终答案与标准结果对比给出奖励。虽没有逐步的过程标注，但最终答案是否正确足以在强化学习流程中当作回报（Reward）来引导模型学会更好的推理，这样模型就知道自己的推理过程是正确的。

对于该训练过程中涉及的强化学习算法，群体相对策略优化（Group Relative Policy Optimization, GRPO）算法，本质上也是PPO算法的变体。在常规的PPO算法之中，通常会引入一个Critic Model（或者称为Value Model）来判断每个当前动作的优劣，从而改进策略，但它的存在也同时带来了两个问题：价值函数估计可能不准确，在LLM的语义环境里里，通常只有一个完整的句子会容易给出一个好坏的判断，而对中间过程生成的token，我们很难给一个准确的奖励值，而价值函数估计不准确则会导致学习策略变差；其次就是模型显存占用高，消耗计算资源大，Critic Model通常和Actor Model的参数量差不多。GPRO的特别之处在于，无需设置Critic Model即Value Model。

GRPO的算法流程如下：

* 从数据集中随机采样一个batch的数据，对于中的每个问题，从旧的策略模型中采样个输出。
* 针对于每个问题，使用也就是Reward Model即Reward Function对每个采样的输出计算奖励分数。
* 对于G个输出，每个输出内部进行token分组，在组内进行相对优势估计（Group Relative Advantage Estimation）来计算每个输出中第个token的优势。
* 最后最大化目标函数来更新策略模型Actor Model，目标函数如下：

(3-21)

为了更加鲜明的对比PPO和GRPO的区别，在此引入PPO的目标函数：

(3-22)

通过对比公式(3-21)和(3-22)，不难发现GRPO的目标优化函数和PPO本质上是大相径庭的，GRPO运用组内进行相对优势估计和优化KL散度惩罚。需要注意的是，GRPO算法存在两种类型的奖励：过程奖励和结果奖励。过程奖励会对模型输出的每一个token进行奖励，而结果奖励则只会对一个response中的最后一个token输出奖励。在计算优势函数时，GRPO算法与之前的PPO算法也有所不同，具体而言：它将奖励减去组平均奖励并除以组标准差进行归一化，将归一化后的奖励作为优势，以结果奖励为例，优势函数可以写为：

(3-23)

过程奖励也是类似，用Reward Model对输出的每一步打分，同样进行归一化，再计算每个 token 之后步骤的归一化奖励之和作为优势，最后总结GRPO和PPO算法的区别：

* PPO算法属于Actor-Critic类型的强化学习算法，需要训练一个与策略模型规模相当的价值函数作为基线，而GRPO算法丢弃了Critic Model，放弃了价值函数近似，通过对同一问题采样多个输出，用这些输出的平均奖励作为基线。
* 优势函数的计算方式不同，PPO算法采用的广义优势估计（GAE）基于奖励和学习到的价值函数计算优势，而GRPO算法根据组内输出的相对奖励计算优势。

算法流程对比图大致如图3-9所示：

图示

描述已自动生成

图3-9 PPO与GRPO算法流程对比图

DeepSeek-R1-Zero就是使用GRPO算法直接得到的推理大模型，在训练历程中甚至自发出现了顿悟时刻(aha moment)，在这个阶段，DeepSeek-R1-Zero 通过重新审视其初始策略，学会了为问题分配更多思考时间。这一行为不仅彰显了模型推理能力的显著提升，也是强化学习如何催生意外且复杂成果的一个生动例证。在大规模强化学习中，模型的思考过程会不断与最终的正确率奖励相互作用。当模型最初得出的答案并未得到较高奖励时，它会在后续的推理中反思自己的思考链路，尝试补充或修正先前的思路，从而获得更高的奖励。随着强化学习的迭代，这种主动回溯、推翻先前想法并重新推理的行为逐渐巩固，便在输出中表现为所谓的“aha moment”。本质上，这是强化学习算法为模型提供了足够的思考和试错空间，当模型自行发现更优思路时，就会出现类似人类恍然大悟的瞬间。这也展示了强化学习这条道路的强大潜力，它可以让模型在没有明确指导的情况下，自主学习并改进自己的思考过程。

但是这样训练出来的模型还是有一定的缺点，一是会出现语言混乱的现象，例如中文回答的过程中会夹杂其他语言，二是输出的内容可读性较差也有格式混乱的问题。所以在训练Deepseek-R1的过程中，还是需要一定的SFT数据来规范模型的输出的。故Deepseek-AI团队使用了高质量思维链数据SFT冷启动+多阶段训练的方式来提高模型的整体表现而不单单是只考虑推理能力，后续也有利用知识蒸馏的方式训练小模型，相关训练流程图可以参考图3-10：

图示

描述已自动生成

图3-10 Deepseek-R1训练流程图

图3-10的内容可以大致分为五个阶段：

* 阶段1：使用少量高质量的 CoT 数据进行冷启动，预热模型。
* 阶段2：进行面向推理的强化学习，提升模型在推理任务上的性能。
* 阶段3：使用拒绝采样和监督微调，提升模型的综合能力，规范输出。
* 阶段4：再次进行强化学习，使模型在所有场景下都表现良好。
* 阶段5：利用Deepseek-R1的蒸馏数据在小模型上进行知识蒸馏。

相比于直接在基础模型上进行纯强化学习，使用少量的高质量CoT数据进行冷启动，可以让模型更快地进入稳定训练阶段，避免在强化学习阶段模型训练的不稳定问题。而且通过精心设计不同场景内高质量CoT冷启动数据的模式，可以引导模型产生更好的推理能力。

后续Deepseek-AI团队进行了一个重要实验来验证了一个关键理论，通过蒸馏 DeepSeek-R1 得到的 DeepSeek-R1-Distill-Qwen-32B 在所有基准测试中的表现均明显优于DeepSeek-R1-Zero-Qwen-32B。这表明，在 Qwen-32B-Base 上，直接进行强化学习训练的效果不如通过蒸馏DeepSeek-R1。这意味着将更强大的模型蒸馏到较小模型中能产生优异的结果，而依赖大规模强化学习的小模型则需消耗巨大的计算资源，且可能仍无法达到蒸馏所能达到的性能水平。同时这也说明了，尽管蒸馏策略既节省成本又具有较高的效率，但若要突破能力的界限，可能仍需依赖参数量更大或者更加强大的基础模型底座以及大规模的精细化控制的强化学习训练策略。

基于Deepseek的训练流程以及其结果的成果，可以印证一个观点，大模型的训练本身就是一个强化学习步骤，SFT也只是一种特殊的强化学习模式，就是只保留策略模型，直接让策略模型学习参考模型的示例输出，或者是RFT让模型学习不符合预期的输出来避免不好的输出。至于DPO、GRPO等，本质上都是PPO的变体，所以本质上能把大模型的训练归类于模型进行强化学习的过程，这也就意味着，针对部分情景，是能够抛弃SFT+RL的训练范式直接通过大规模强化学习就可以得到预期的效果。

# 第四章 数据准备和模型训练实验方案设计

在本章中，会介绍实验进行前所做的数据准备工作，本次实验使用的数据包括直接下载的开源数据以及基于开源数据经过Deepseek-R1蒸馏过后得到的数据，本章会简单阐述这一流程，但不会过多赘述，因为不是所有数据都用于后续的实验。其次会介绍实验使用到的开源工具框架以及自己开发的GRPO框架模版，最后会详细展开本次模型训练的实验方案和相关评估指标。

## 4.1 数据准备

### 4.1.1 知识强化类数据和普通思维链推理数据准备

在第一章中提到要训练一个相对来说知识强化型的模型，因此本人选择使用UW-Madison-Lee-Lab制作的开源数据集UW-Madison-Lee-Lab/MMLU-Pro-CoT-Train-Labeled共84000条问答对SFT数据，通过GPT4o，Gemini-1.5以及Qwen2.5-72B-Instruct，利用CoT的System Prompt结合few shot进行部分问答对的再生成，最终得到了MMLU-Pro-CoT-Train-84K，作为知识强化型的模型SFT微调数据，其中20K数据被随机抽样出来用于MoE架构训练，保证模型的稳定性。

对于普通思维链推理数据，这里的普通是指不像Deepseek-R1或者O1那样，会有冗长的思维过程展示，而是放在最后的回复中，因为本人制作这组数据时O1还尚未发布。在经过筛选后，我选择微软发布的Orca-Math-200K [80]和Numina团队制作的Numina-Math-COT [81]约860000条数据，通过1:2的方式，随机抽取，并对于Orca-Math-200K中抽取得到的数据同样通过GPT4o，Gemini-1.5利用CoT的System Prompt结合few shot进行了思维链数据的强化，Numina的数据则不做改变只是处理了相关的数据字段进行匹配，得到了Reasoning-150K的150000条问答对的推理数据，作为推理强化型模型的SFT数据

### 4.1.2 带思考过程的知识蒸馏数据准备

本节介绍用于知识蒸馏的带思考过程的数据准备，主要的蒸馏方式是基于Deepseek-R1的API接口对数学推理数据进行强化再生成，由于是为了构建SFT微调数据，在此直接将思考过程和最后的答案输出耦合起来作为一个字段表示模型的生成内容。

首先是基于Numina-Math-CoT，在其840K条问答对数据中，抽样了1/5的数据约172K，利用R1进行了再生成，得到了Numina-Math-R1-COT-172K。其次本人在GSM8K-CN中抽取了6K数据用于R1再生成，将Meta发布的MetaMathQA-395K数据也做了带思考过程的强化，然后从AMC（美国数学竞赛，American Mathematics Competition）和AIME（美国数学邀请赛，American Invitational Mathematics Examination）1983-2023的所有考题数据约4K数据进行答案生成，最后进行答案正确性的筛选，得到约2.5K的数据R1-Annotated-AMC-AIME-True-2K。后续根据一些开源数据重新生成后得到了R1-Distill-MultiMath-76K、R1-Preview-Data-40K。

最后是在Hugging Face社区中搜寻得来的开源数据，包括多场景的R1蒸馏中文数据Chinese-DeepSeek-R1-Distill-data-110K、OpenR1-Math-220（只取93K的default版本）、GeneralThought-123K。

### 4.1.3 用于GRPO强化学习训练的数据准备

这一小节涉及到的数据准备相对来说不会特别复杂，因为这组数据的要求只需要问题和答案，完全不需要思考过程的描述。而且数据量不宜过多，因为GRPO强化学习虽然说减少了资源的消耗，但由于采样过程和Reference Model的存在，仍然需要大量的算力资源。对此，本人基于上一小节准备的数据，筛选出了7500条数据，将问题的答案提取出来构成新字段，最后再通过随机抽样划分出了三个数据集： R1-Zero-GRPO-750、R1-Zero-GRPO-1500、R1-Zero-GRPO-7500，这三个数据集将用于本研究对GRPO强化学习的初步探索实验。但为了能够得到有一定可用性的模型，后续会使用开源的工具进行GRPO的训练，所以将上一节R1-Preview-Data-40K进行答案提取后，作为本次GRPO训练的第一组数据选择，其次从开源社区搜寻后决定使用FastCuRL共80K数据作为第二组数据选择，分别来源于AIME problems (1984-2023)、AMC problems (before 2023)、Omni-MATH Dataset、Still Dataset。至此，数据准备工作完成。

## 4.2 实验环境和工具框架介绍

### 4.2.1 实验环境

本次实验是在不同的两台服务器环境下进行的，分别为单卡A100-SXM-80GB和八卡A800-SXM-40GB服务器。本小节会介绍服务器相关环境配置。

单卡服务器环境配置如下：

* CPU：Intel(R) Xeon(R) Platinum 8336C CPU @ 2.30GHz\*1
* 内存：232GB
* 硬盘：12TB
* GPU：A100-SXM-80GB
* CUDA：12.4
* 系统内核：GNU/Linux

表格

描述已自动生成

图4-1 单卡服务器GPU信息图

八卡服务器环境配置如下：

* CPU：Intel(R) Xeon(R) Platinum 8336C CPU @ 2.30GHz\*2
* 内存：2TB
* 硬盘：24TB
* GPU：A100-SXM-40GB\*8
* CUDA：12.4
* 系统内核：GNU/Linux

表格

描述已自动生成

图4-2 八卡服务器GPU信息图

上述两台服务器都会以Python3.10作为进行实验的Python版本。

### 4.2.2 实验工具框架

本小节中，会介绍本次实验使用的工具框架，用于模型的下载、存储、训练、推理。

首先是基座模型的下载和训练后模型的云存储框架HuggingFace [82]，HuggingFace是一个自然语言处理。工具集和平台，致力于为研究人员和工程师提供高效、易用且可重复的NLP解决方案。它成立于2016年，总部位于美国纽约市，并在加拿大蒙特利尔、巴黎和旧金山等地设有办事处。其提供各种NLP解决方案，如提供预训练模型、提供开源NLP库Transformers库用于任务化的微调测试优化等。同时其支持多种流行的深度学习框架如PyTorch和TensorFlow进行实现的。此外，HuggingFace还提供了各种预训练模型的API和文档，使得用户可以方便地使用它们来完成自己的NLP任务。同时用户也可以自由地在自己的账号空间内上传自己的模型和数据集进行存储，NLP 开发人员或者爱好人士可以分享和发现代码、模型和数据集。HuggingFace 的 Transformer 库已经成为了自然语言处理领域的标杆，被应用于各种文本生成和理解任务，其平台和工具也成为了每一个NLP开发人员的必备工具和框架。

其次就是实验所需的训练框架，这里主要分为四个部分：SFT训练框架、MoE的模型合并框架、自行开发的GRPO初步探索简单工具、成熟的GRPO工程化训练框架。

在SFT和知识蒸馏阶段，本人采用了业内广受好评的LLaMA-Factory[83]工具框架，LLaMA-Factory 是一个易于使用的大规模语言模型（Large Language Model, LLM）微调框架，它支持多种模型，包括 LLaMA、BLOOM、Mistral、Baichuan、Qwen 和 ChatGLM 等，而且会基于开源模型架构不断进行更新。该框架旨在简化大型语言模型的微调过程，封装并提供了一套完整的工具和接口，使得用户能够轻松地对预训练的模型进行定制化的训练和调整，以适应特定的应用场景。同时可以使用多种微调算法，包括SFT、RFT、PPO、DPO等，基本上只需要做好基本的数据处理即可使用工具微调自己的大模型。此外，本人为了探索MoE的能力和过程，采用了MergeKit[84]工具框架，该工具专门设计用于无缝整合大语言模型。通过模型合并技术，我们可以有效整合多个模型的参数，从而在不重新训练的情况下创建出更强有力的多任务模型。该工具框架通过YAML配置文件来定义合并操作。这种方式使得用户可以灵活地指定合并方法、模型切片、参数设置等细节，而且通过采用懒加载张量的技术，可以在内存受限的情况下处理大型模型的合并。

然后是用于GRPO初步探索由本人自己开发的简单框架，更应该称为简单的调用GRPO进行训练的相关代码，在这个过程中，使用较为简单的方法进行GRPO的算法调用执行，以及适配性一般的数据集处理和读取，主要是为了探索GRPO的一个具体流程是如何创建的。其中的奖励函数设置也较为简单，分为格式奖励和答案奖励，格式奖励是让模型将思考过程放到<think><\think>标签中根据模型格式是否符合要求给予奖励，答案奖励是将放置于特殊标记的答案提取出来和正确答案进行对比后，根据正确与否进行奖励。不过这样训练出来的模型还是略显稚嫩和粗糙。

之后为了得到较好效果和性能表现的模型，采取了工程化的较为成熟的强化学习大模型训练工具框架VERL[85]，该框架不仅兼容多种训练和推理框架，还支持模型的灵活部署以及多种RL算法的实现，同时其采用了混合编程模型，该模型融合了单控制器的灵活性和多控制器的高效性。这种设计能够更好地实现和执行多种RL算法，从而显著提升训练吞吐量，并降低开发和维护的复杂度，其相关实验结果也显示，在各种模型规模和RL算法下的训练吞吐量提升了1.5倍至20倍。在Deepseek-R1发布后，该工具也是很快地进行了更新，适配了GRPO算法的训练。

最后在强化学习过程中以及后续测试的过程中，需要用到的推理框架vLLM[59]，是伯克利大学LMSYS组织开源的大语言模型高速推理框架，旨在极大地提升实时场景下的语言模型服务的吞吐与内存使用效率。vLLM是一个快速且易于使用的库，用于 LLM 推理和服务，可以和HuggingFace 无缝集成。vLLM利用了全新的注意力算法PagedAttention，有效地管理注意力键和值节省内存空间。同时其吞吐量最多可以达到 HuggingFace 实现的24倍，文本生成推理（TGI）高出3.5倍，并且不需要对模型结构进行任何的改变。为了提高强化学习训练过程中采样过程以及后续模型测试时的回答效率和资源消耗，故选择了该工具作为本次实验的核心推理工具框架。

## 4.3 实验方案和评估指标

### 4.3.1 实验方案

### 4.3.2 评估指标

# 第五章 实验结果分析

# 第六章 简单应用模版的搭建演示

# 第七章 研究总结与未来展望

# 第八章 参考文献

[1] 文森, 钱力, 胡懋地, et al. 基于大语言模型的问答技术研究进展综述 [J]. 2024, 1.

[2] LEWIS P, PEREZ E, PIKTUS A, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks [J]. 2020, 33(9459-74.

[3] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization algorithms [J]. 2017,

[4] WEI J, WANG X, SCHUURMANS D, et al. Chain-of-thought prompting elicits reasoning in large language models [J]. 2022, 35(24824-37.

[5] GOU J, YU B, MAYBANK S J, et al. Knowledge distillation: A survey [J]. 2021, 129(6): 1789-819.

[6] KAELBLING L P, LITTMAN M L, MOORE A W J J O A I R. Reinforcement learning: A survey [J]. 1996, 4(237-85.

[7] CHEN T, KORNBLITH S, SWERSKY K, et al. Big self-supervised models are strong semi-supervised learners [J]. 2020, 33(22243-55.

[8] LI J, WANG X, ZHU S, et al. Cumo: Scaling multimodal llm with co-upcycled mixture-of-experts [J]. 2024, 37(131224-46.

[9] ARAI K J I J O A C S, APPLICATIONS. Design of On-Premises Version of RAG with AI Agent for Framework Selection Together with Dify and DSL as Well as Ollama for LLM [J]. 2024, 15(12):

[10] YANG A, YANG B, HUI B, et al. Qwen2 technical report [J]. 2024,

[11] HENDRYCKS D, BURNS C, BASART S, et al. Measuring massive multitask language understanding [J]. arXiv preprint arXiv:200903300, 2020,

[12] YUAN Z, YUAN H, LI C, et al. Scaling relationship on learning mathematical reasoning with large language models [J]. arXiv preprint arXiv:230801825, 2023,

[13] GU A, ROZIÈRE B, LEATHER H, et al. Cruxeval: A benchmark for code reasoning, understanding and execution [J]. arXiv preprint arXiv:240103065, 2024,

[14] LIU B, BUBECK S, ELDAN R, et al. Tinygsm: achieving> 80% on gsm8k with small language models [J]. 2023,

[15] SINGH S, ROMANOU A, FOURRIER C, et al. Global mmlu: Understanding and addressing cultural and linguistic biases in multilingual evaluation [J]. 2024,

[16] ZHENG Q, XIA X, ZOU X, et al. Codegeex: A pre-trained model for code generation with multilingual benchmarking on humaneval-x; proceedings of the Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, F, 2023 [C].

[17] RAJBHANDARI S, LI C, YAO Z, et al. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale; proceedings of the International conference on machine learning, F, 2022 [C]. PMLR.

[18] YANG A, YANG B, ZHANG B, et al. Qwen2. 5 technical report [J]. 2024,

[19] GUO D, YANG D, ZHANG H, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning [J]. 2025,

[20] SHAO Z, WANG P, ZHU Q, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models [J]. 2024,

[21] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need [J]. 2017, 30(

[22] SHERSTINSKY A J P D N P. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network [J]. 2020, 404(132306.

[23] JAWAHAR G, SAGOT B, SEDDAH D. What does BERT learn about the structure of language?; proceedings of the ACL 2019-57th Annual Meeting of the Association for Computational Linguistics, F, 2019 [C].

[24] GREZES F, BLANCO-CUARESMA S, ACCOMAZZI A, et al. Building astroBERT, a language model for astronomy & astrophysics [J]. 2021,

[25] TAY Y, DEHGHANI M, RAO J, et al. Scale efficiently: Insights from pre-training and fine-tuning transformers [J]. 2021,

[26] ACHIAM J, ADLER S, AGARWAL S, et al. Gpt-4 technical report [J]. 2023,

[27] ZHANG M, LI J J F R. A commentary of GPT-3 in MIT Technology Review 2021 [J]. 2021, 1(6): 831-3.

[28] FLORIDI L, CHIRIATTI M J M, MACHINES. GPT-3: Its nature, scope, limits, and consequences [J]. 2020, 30(681-94.

[29] LEE J-S, HSIANG J J W P I. Patent claim generation by fine-tuning OpenAI GPT-2 [J]. 2020, 62(101983.

[30] LI Y, LI Z, ZHANG K, et al. Chatdoctor: A medical chat model fine-tuned on a large language model meta-ai (llama) using medical domain knowledge [J]. 2023, 15(6):

[31] HOFFMANN J, BORGEAUD S, MENSCH A, et al. Training compute-optimal large language models [J]. 2022,

[32] BAI J, BAI S, CHU Y, et al. Qwen technical report [J]. 2023,

[33] DAO T, FU D, ERMON S, et al. Flashattention: Fast and memory-efficient exact attention with io-awareness [J]. 2022, 35(16344-59.

[34] YANG A, XIAO B, WANG B, et al. Baichuan 2: Open large-scale language models [J]. 2023,

[35] CHEN Z, WU J, WANG W, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks; proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, F, 2024 [C].

[36] HU S, TU Y, HAN X, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies [J]. 2024,

[37] KAPLAN J, MCCANDLISH S, HENIGHAN T, et al. Scaling laws for neural language models [J]. 2020,

[38] WHITE J, FU Q, HAYS S, et al. A prompt pattern catalog to enhance prompt engineering with chatgpt [J]. 2023,

[39] WANG Y, YAO Q, KWOK J T, et al. Generalizing from a few examples: A survey on few-shot learning [J]. 2020, 53(3): 1-34.

[40] YAO S, YU D, ZHAO J, et al. Tree of thoughts: Deliberate problem solving with large language models [J]. 2024, 36(

[41] ZIEGLER D M, STIENNON N, WU J, et al. Fine-tuning language models from human preferences [J]. 2019,

[42] LI Y, CHOI D, CHUNG J, et al. Competition-level code generation with alphacode [J]. 2022, 378(6624): 1092-7.

[43] YIN S, YOU W, JI Z, et al. MuMath-Code: Combining Tool-Use Large Language Models with Multi-perspective Data Augmentation for Mathematical Reasoning [J]. 2024,

[44] GOU Z, SHAO Z, GONG Y, et al. Tora: A tool-integrated reasoning agent for mathematical problem solving [J]. 2023,

[45] YANG A, ZHANG B, HUI B, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement [J]. 2024,

[46] DAI J, PAN X, SUN R, et al. Safe rlhf: Safe reinforcement learning from human feedback [J]. 2023,

[47] JAECH A, KALAI A, LERER A, et al. Openai o1 system card [J]. 2024,

[48] LIU A, FENG B, WANG B, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model [J]. 2024,

[49] ZHU Q, GUO D, SHAO Z, et al. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence [J]. 2024,

[50] ZHENG C, ZHANG Z, ZHANG B, et al. Processbench: Identifying process errors in mathematical reasoning [J]. 2024,

[51] LI H, ZHU C, ZHANG Y, et al. Task-specific fine-tuning via variational information bottleneck for weakly-supervised pathology whole slide image classification; proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, F, 2023 [C].

[52] ZHANG Y, YANG Q J I T O K, ENGINEERING D. A survey on multi-task learning [J]. 2021, 34(12): 5586-609.

[53] CHEN T, LIU S, CHANG S, et al. Adversarial robustness: From self-supervised pre-training to fine-tuning; proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, F, 2020 [C].

[54] LIU T, ZHAO Y, JOSHI R, et al. Statistical rejection sampling improves preference optimization [J]. 2023,

[55] RAFAILOV R, SHARMA A, MITCHELL E, et al. Direct preference optimization: Your language model is secretly a reward model [J]. 2024, 36(

[56] HISAHARO S, NISHIMURA Y, TAKAHASHI A J A P. Optimizing llm inference clusters for enhanced performance and energy efficiency [J]. 2024,

[57] LIN J, TANG J, TANG H, et al. Awq: Activation-aware weight quantization for on-device llm compression and acceleration [J]. 2024, 6(87-100.

[58] DETTMERS T, PAGNONI A, HOLTZMAN A, et al. Qlora: Efficient finetuning of quantized llms [J]. 2023, 36(10088-115.

[59] KWON W, LI Z, ZHUANG S, et al. Efficient memory management for large language model serving with pagedattention; proceedings of the Proceedings of the 29th Symposium on Operating Systems Principles, F, 2023 [C].

[60] ZHENG L, YIN L, XIE Z, et al. Sglang: Efficient execution of structured language model programs [J]. 2024, 37(62557-83.

[61] YIN W, XU M, LI Y, et al. Llm as a system service on mobile devices [J]. 2024,

[62] KATTENBORN T, LEITLOFF J, SCHIEFER F, et al. Review on Convolutional Neural Networks (CNN) in vegetation remote sensing [J]. 2021, 173(24-49.

[63] ANDONI A, INDYK P, LAARHOVEN T, et al. Practical and optimal LSH for angular distance [J]. 2015, 28(

[64] WU W, LI B, CHEN L, et al. A review for weighted minhash algorithms [J]. 2020, 34(6): 2553-73.

[65] AINSLIE J, LEE-THORP J, DE JONG M, et al. Gqa: Training generalized multi-query transformer models from multi-head checkpoints [J]. 2023,

[66] AN C, HUANG F, ZHANG J, et al. Training-free long-context scaling of large language models [J]. 2024,

[67] PENG B, QUESNELLE J, FAN H, et al. Yarn: Efficient context window extension of large language models [J]. 2023,

[68] BAI Y, JONES A, NDOUSSE K, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback [J]. 2022,

[69] GILKS W R, WILD P J J O T R S S S C. Adaptive rejection sampling for Gibbs sampling [J]. 1992, 41(2): 337-48.

[70] RAFAILOV R, SHARMA A, MITCHELL E, et al. Direct preference optimization: Your language model is secretly a reward model [J]. 2023, 36(53728-41.

[71] HUI B, YANG J, CUI Z, et al. Qwen2. 5-coder technical report [J]. 2024,

[72] BAI S, CHEN K, LIU X, et al. Qwen2. 5-vl technical report [J]. 2025,

[73] LU K, YUAN H, YUAN Z, et al. # instag: Instruction tagging for analyzing supervised fine-tuning of large language models [J]. 2023,

[74] MICIKEVICIUS P, NARANG S, ALBEN J, et al. Mixed precision training [J]. 2017,

[75] TARG S, ALMEIDA D, LYMAN K J A P A. Resnet in resnet: Generalizing residual architectures [J]. 2016,

[76] OUYANG L, WU J, JIANG X, et al. Training language models to follow instructions with human feedback [J]. 2022, 35(27730-44.

[77] SCHULMAN J, LEVINE S, ABBEEL P, et al. Trust region policy optimization; proceedings of the International conference on machine learning, F, 2015 [C]. PMLR.

[78] LIU A, FENG B, XUE B, et al. Deepseek-v3 technical report [J]. arXiv preprint arXiv:241219437, 2024,

[79] ZHENG C, ZHANG Z, ZHANG B, et al. Processbench: Identifying process errors in mathematical reasoning [J]. arXiv preprint arXiv:241206559, 2024,

[80] MITRA A, KHANPOUR H, ROSSET C, et al. Orca-math: Unlocking the potential of slms in grade school math [J]. arXiv preprint arXiv:240214830, 2024,

[81] LI J, BEECHING E, TUNSTALL L, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions [J]. Hugging Face repository, 2024, 13(9.

[82] JAIN S M. Hugging face [M]. Introduction to transformers for NLP: With the hugging face library and models to solve problems. Springer. 2022: 51-67.

[83] ZHENG Y, ZHANG R, ZHANG J, et al. Llamafactory: Unified efficient fine-tuning of 100+ language models [J]. 2024,

[84] GODDARD C, SIRIWARDHANA S, EHGHAGHI M, et al. Arcee’s mergekit: A toolkit for merging large language models; proceedings of the Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, F, 2024 [C].

[85] SHENG G, ZHANG C, YE Z, et al. Hybridflow: A flexible and efficient rlhf framework [J]. arXiv preprint arXiv:240919256, 2024,