Optimizing Text Summarization through Reinforcement Learning with Human Feedback on Large Language Models

Project - I (AI67101) report submitted to Indian Institute of Technology Kharagpur requirements for the degree

of

Master of Technology

by

Adarsh Sharma
(20ME3AI10)

Under the guidance of

Professor Prabhat Kumar Mishra

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Department of Student Department
Indian Institute of Technology,
Kharagpur
India - 721302

CERTIFICATE

This is to certify that we have examined the thesis entitled **Optimizing Text Summarization through Reinforcement Learning with Human Feedback on Large Language Models**, submitted by **Adarsh Sharma** (Roll Number: *20ME3AI10*) a postgraduate student of **Department of Artificial Intelligence** in partial fulfillment for the award of degree of Master of Technology in Artificial Intelligence Engineering. We hereby accord our approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfillment for the Post Graduate Degree for which it has been submitted. The thesis has fulfilled all the requirements as per the regulations of the Institute and has reached the standard needed for submission.

Prabhat Kumar Mishra

Department of Artificial Intelligence
Indian Institute of Technology,
Kharagpur

Place: Kharagpur
Date: November 29, 2024

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to the individuals and institutions who have been instrumental in the successful completion of this thesis.
First and foremost, I am profoundly thankful to my thesis advisor, Professor Prabhat Kumar Mishra, for his unwavering guidance, patience, and invaluable insights throughout the research process. His expertise and mentorship have been pivotal in shaping the direction and quality of this work.

I would also like to acknowledge the Indian Institute of Technology, Kharagpur, for providing the necessary resources and facilities to conduct this research. The institution's academic environment has fostered my growth and learning.

Additionally, I am grateful to my friends and colleagues for their moral support, thoughtful discussions, and camaraderie, which made this research journey both enrich and enjoyable.

This thesis would not have been possible without the collective contributions and support of these remarkable individuals and institutions. I am sincerely thankful for their involvement and encouragement throughout this academic endeavor.

Adarsh Sharma
IIT Kharagpur
Date: November 29th, 2024

ABSTRACT

This project explores the use of Reinforcement Learning with Human Feedback (RLHF) to enhance text summarization tasks, focusing on both advanced techniques and resource-efficient approaches. RLHF integrates human preferences to guide model behavior, with methods like Proximal Policy Optimization (PPO) and Direct Preference Optimization (DPO) evaluated for their effectiveness and efficiency.

Using the OpenAI/SummarizeFromFeedback dataset, we fine-tuned reward models, supervised fine-tuned (SFT) models, and PPO models for abstractive summarization. Challenges, such as memory limitations with large models (e.g., 7B parameters), were tackled using gradient accumulation and quantization. Comparisons with smaller models, such as Gemma-2B, highlighted their utility as reward classifiers despite limited generative capabilities. The limitations of other RL algorithms, including A2C, A3C, and TRPO, were discussed in terms of scalability and stability. Results showed trade-offs between resources and performance, with Gemma-2B excelling in classification tasks and PPO improving summarization quality. Key insights include the importance of aligning model selection with resources and the need for robust library versioning to ensure reproducibility.

The study concludes with recommendations for using appropriately scaled models and highlights future directions, such as testing alternative RL algorithms like ILQL, scaling models for better performance, and extending RL applications beyond NLP to fields like robotics.

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem Statement

The rapid growth of large language models (LLMs) has revolutionized natural language processing (NLP), enabling tasks like text summarization, translation, and conversational AI. However, these models often generate outputs that are misaligned with human preferences, lacking interpretability, factual accuracy, or coherence. Standard supervised fine-tuning alone struggles to address these shortcomings, as it relies heavily on static datasets that may not capture nuanced human judgment or contextual expectations. This limitation becomes especially evident in complex tasks like abstractive text summarization, where models need to generate concise and meaningful summaries while preserving the core intent of the original text.

To address this, Reinforcement Learning with Human Feedback (RLHF) has emerged as a promising approach. RLHF incorporates human preferences into the training loop, allowing models to align their behavior with human values through iterative feedback. While RLHF has shown impressive results in improving generative models, such as OpenAI's ChatGPT, it remains computationally intensive and resource-heavy. Training large models with reinforcement learning techniques like Proximal Policy Optimization (PPO) requires substantial memory and computational power, often beyond the reach of researchers with limited resources.

Additionally, the lack of high-quality, task-specific datasets further complicates RLHF-based training. Although datasets like OpenAI's SummarizeFromFeedback and HuggingFace's Stack Exchange Preferences exist, they are few and far between,

necessitating a reliance on carefully curated data. The choice of RL algorithms also poses challenges; many techniques, such as A2C, TRPO, and DQN, have inherent limitations, including scalability, stability, and inefficiency, making PPO and other modern approaches more suitable yet still computationally demanding.

This thesis addresses these challenges by exploring efficient RLHF techniques for abstractive text summarization, leveraging resource-conscious strategies, evaluating alternative RL algorithms, and providing insights into aligning model performance with available computational resource

1.2 Background and Motivation

Text summarization is a critical task in natural language processing (NLP), enabling users to extract concise, meaningful insights from extensive content. Abstractive summarization, which generates novel summaries, has gained prominence over extractive methods that merely copy text segments. However, large language models (LLMs) often produce summaries misaligned with human expectations, highlighting the need for advanced training methods. Reinforcement Learning with Human Feedback (RLHF) bridges this gap by integrating human preferences into model optimization, enhancing alignment and coherence. Motivated by RLHF's success in generative models like ChatGPT, this research explores its application to text summarization, balancing performance with computational efficiency

1.3 Research Objective

This research focuses on advancing abstractive text summarization by applying Reinforcement Learning with Human Feedback (RLHF) to enhance the alignment between model outputs and human expectations. Abstractive summarization is a complex task, as it requires generating concise, coherent, and contextually accurate summaries that preserve the core meaning of the original text. However, traditional supervised learning methods often fall short in aligning the generated summaries with human preferences. RLHF addresses this gap by incorporating human feedback directly into the training process, allowing the model to iteratively improve its outputs based on

human judgments.

The study explores various reinforcement learning (RL) algorithms, with a particular emphasis on Proximal Policy Optimization (PPO) and Direct Preference Optimization (DPO). PPO has gained prominence due to its stability and scalability in training large models, making it a suitable choice for RLHF tasks. DPO, which focuses on optimizing the model based on preference rankings instead of absolute reward values, is also investigated as a more resource-efficient approach. These algorithms are compared to traditional methods like A2C (Advantage Actor-Critic) and TRPO (Trust Region Policy Optimization), which have limitations in terms of computational efficiency and scalability, making them less ideal for modern NLP tasks at scale.

To overcome the computational challenges of training large language models (LLMs) on RLHF tasks, strategies such as gradient accumulation and model quantization are employed. These techniques reduce memory usage and speed up training. Additionally, smaller models like Gemma-2B, which excel in reward classification tasks, are used to generate more efficient reward signals. This enables the study to assess how well these reward models can guide LLMs in summarization tasks.

The research utilizes high-quality datasets, such as OpenAI's *SummarizeFromFeedback*, to fine-tune the models. Performance is evaluated using metrics like Rouge scores to quantify improvements in summarization quality. Ultimately, this research provides valuable insights for implementing RLHF in NLP tasks, offering practical solutions for researchers with limited computational resources while improving summarization outputs.

Chapter 2

Literature Review

2.1 Understanding Reinforcement Learning

Reinforcement Learning (RL) is a machine learning paradigm where an agent learns to make decisions by interacting with an environment to maximize a cumulative reward. Unlike supervised learning, RL does not rely on labeled data but instead uses feedback signals from the environment. The agent navigates through states, takes actions, and receives rewards based on the outcomes, iteratively improving its policy—a mapping from states to actions.

Central to RL is the Markov Decision Process (MDP) framework, comprising states, actions, rewards, and transitions. RL algorithms can be broadly categorized into value-based methods, like Q-Learning, which estimate the value of actions, and policy-based methods, like Proximal Policy Optimization (PPO), which optimize policies directly. Hybrid methods, such as Actor-Critic, combine these approaches for improved performance.

Applications of RL span diverse fields, including robotics, gaming, and NLP, where it enhances tasks like text summarization and generation through techniques like RLHF (Reinforcement Learning with Human Feedback).

2.2 Training LLMs on Custom Datasets

Training and fine-tuning large language models (LLMs) on custom datasets have become central to improving model performance for specific tasks. LLMs, like GPT

and BERT, are typically pre-trained on massive corpora of general-purpose data. However, fine-tuning allows for adaptation to more specialized domains, improving task-specific performance, such as sentiment analysis, text summarization, or medical question answering.

Fine-tuning LLMs typically involves supervised learning, where models are trained on labeled data corresponding to the desired task. However, the process can be further enhanced with techniques like domain adaptation, where the model is exposed to a smaller, task-relevant dataset that fine-tunes its general knowledge for specialized language patterns.

Transfer learning is another powerful technique where a pre-trained model is adapted for a new domain with minimal data. Fine-tuning can also be done using unsupervised approaches, where models learn patterns or structures from unannotated data through methods like self-supervised learning.

When working with custom datasets, especially in NLP tasks like text summarization, the quality of data plays a crucial role. The choice of fine-tuning techniques, such as training from scratch, supervised fine-tuning (SFT), or reinforcement learning, must align with the dataset size, task complexity, and computational resources available

Chapter 3

Methodology

3.1 Improving Text Generation with RL

Reinforcement Learning (RL) offers a powerful framework for improving generative text generation tasks, such as question answering, summarization, and dialogue generation. Traditional supervised learning methods, while effective for many tasks, often struggle with capturing complex human preferences and producing high-quality, contextually relevant outputs. RL, and more specifically Reinforcement Learning with Human Feedback (RLHF), addresses this gap by enabling models to learn from dynamic feedback, which enhances the alignment of their generated outputs with human expectations.

In generative tasks, RLHF allows models to iteratively refine their responses based on feedback from human evaluators. Instead of merely training on fixed labels or datasets, RLHF enables the model to adjust its behavior by receiving reward signals that reflect human judgment about the quality of its outputs. This approach is particularly valuable in tasks like question answering or dialogue generation, where the "correct" output can be subjective, varying based on context, tone, or complexity.

By using RL algorithms such as Proximal Policy Optimization (PPO), which is known for its stability and scalability, RLHF helps to guide generative models toward more coherent, informative, and contextually appropriate responses. PPO, for instance, allows models to optimize their responses by balancing exploration (trying different answers) and exploitation (refining successful answers). Through this iterative feedback loop, models improve not only in accuracy but also in producing content

that resonates more closely with human users' expectations.
Thus, RLHF significantly enhances the quality of generative text generation tasks, ensuring that models generate more human-like, contextually accurate, and relevant outputs.

3.2 RLHF Architecture: Why PPO and Not A2C?

In the context of generative text generation tasks, the architecture for Reinforcement Learning with Human Feedback (RLHF) plays a crucial role in ensuring that models align closely with human preferences. RLHF leverages RL algorithms to fine-tune large language models (LLMs) by using human feedback as a reward signal. One of the key decisions in implementing RLHF is selecting the most suitable RL algorithm. Among the various algorithms available, Proximal Policy Optimization (PPO) is often preferred over others like A2C (Advantage Actor-Critic), A3C, and TRPO (Trust Region Policy Optimization). Below, we discuss why PPO is a popular choice for RLHF in generative tasks.

Why PPO Over A2C? Stability and Efficiency: PPO is known for its stability in training, which makes it a robust choice when fine-tuning large models. Unlike A2C, which uses a single agent to update both the actor and critic simultaneously, PPO employs a more conservative update mechanism through a clipped objective function. This minimizes the risk of large updates that could destabilize the training process. A2C, on the other hand, may suffer from instability, especially in high-dimensional environments like generative text tasks where the reward signal can be sparse and noisy.

Scalability: PPO scales better to larger models and more complex environments. This scalability is crucial when working with LLMs, which require substantial computational resources. In contrast, A2C and its variant A3C (which uses multiple agents for parallelism) have been shown to be less efficient for large-scale text generation tasks. While A3C does leverage parallelism, the inherent complexity and instability of the algorithm reduce its practicality in fine-tuning LLMs.

Ease of Implementation: PPO is easier to implement than algorithms like TRPO, which require second-order optimization methods (which are computationally expensive) and have more complex constraints. PPO, by contrast, uses a simpler first-order

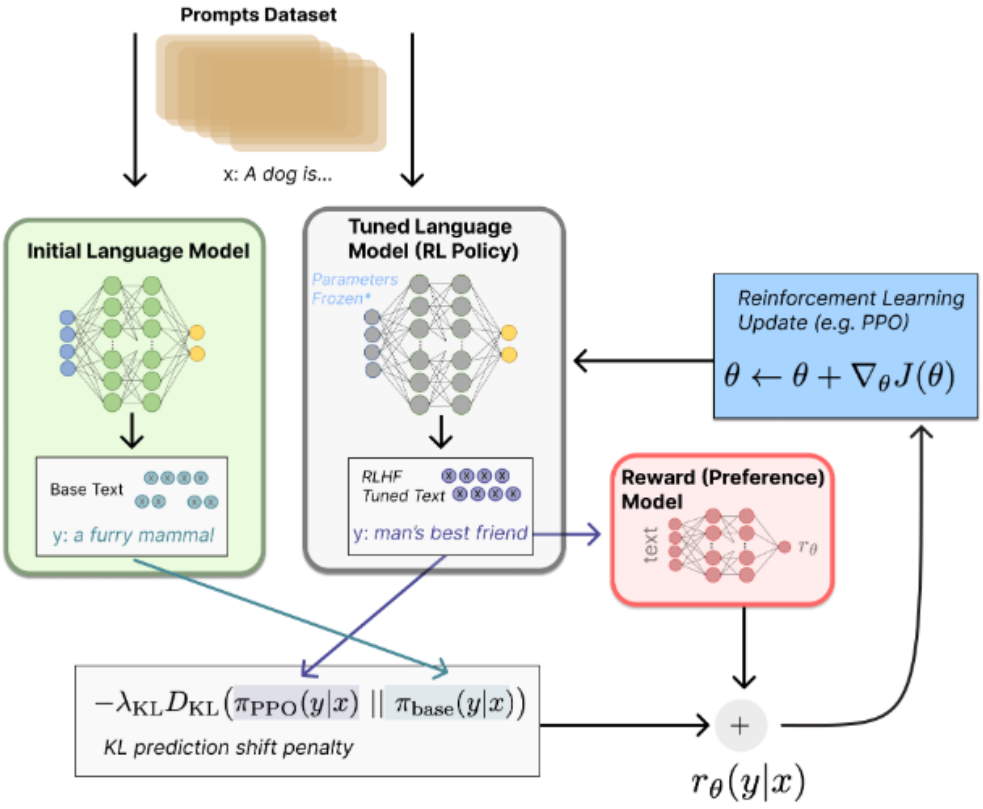


Figure 3.1: : RLHF Architecture for Generative Text Generation

optimization process and is more accessible in practice for RLHF tasks.

Performance in NLP Tasks: PPO has been proven effective in various NLP tasks, including language modeling, text generation, and summarization. It offers a good balance between exploration (trying diverse outputs) and exploitation (refining optimal outputs), crucial for generative tasks where the diversity and quality of the output must be optimized simultaneously.

Why Not A2C or A3C? A2C (Advantage Actor-Critic): While A2C offers a simple and straightforward approach, its less stable learning process makes it challenging to fine-tune large generative models. In tasks like text generation, where the environment is dynamic and feedback is often sparse, A2C can struggle with maintaining consistent performance.

A3C (Asynchronous Advantage Actor-Critic): A3C improves upon A2C by using multiple parallel agents, but its training is still less stable than PPO. Additionally, the parallelization strategy increases computational overhead, making it less efficient for large-scale generative tasks. Furthermore, A3C requires significant engineering effort to manage multiple agents and ensure stability across them, which may not be practical for RLHF-based fine-tuning.

3.3 Our Task: Text Summarization

Text summarization has been a central task in natural language processing (NLP), aiming to condense a long document into a shorter, more concise form while preserving its essential information. The two main approaches to text summarization are extractive summarization and abstractive summarization, each with distinct methodologies and applications.

History of Text Summarization The history of text summarization dates back to the 1950s, where early approaches focused on extractive techniques, which involve selecting a subset of the original text to create a summary. Early systems relied heavily on statistical methods, like keyword extraction and sentence ranking based

on features such as term frequency or sentence length. However, as research in NLP progressed, the limitations of extractive methods became evident, particularly in their inability to produce coherent and human-like summaries.

In recent years, abstractive summarization has gained significant attention, driven by advances in deep learning and large-scale language models (LLMs). Unlike extractive methods, abstractive summarization generates summaries by rephrasing and paraphrasing the original text, similar to how a human might summarize the content. This approach requires a deeper understanding of the text's meaning and structure, which can be achieved using sequence-to-sequence models, transformers, and more recently, reinforcement learning with human feedback (RLHF) for fine-tuning models.

Selective vs. Abstractive Summarization Selective Summarization: Selective summarization, often associated with extractive methods, involves identifying and selecting key sentences, phrases, or segments directly from the original text. The goal is to retain the most important information, removing unnecessary details. While it is computationally less complex and guarantees that the summary contains only information present in the original document, selective summarization often results in summaries that may lack fluency, coherence, or completeness. The final summary may also have redundant information since it simply extracts portions of the original text.

Abstractive Summarization: Abstractive summarization generates new sentences that convey the essential ideas of the original text but may not use the exact phrasing found in the source. This approach is more flexible and closer to human summarization, as it paraphrases and restructures the content. Abstractive methods have the advantage of producing more coherent and concise summaries, but they are more challenging to train and require more sophisticated models. With advancements in deep learning, especially transformer models like BERT, GPT, and T5, abstractive summarization has seen substantial improvements, providing summaries that better capture the underlying meaning and context.

In our task, we focus on abstractive summarization, aiming to generate summaries that are both contextually relevant and concise, capturing the essence of the original

text. The shift from extractive to abstractive summarization reflects the growing capabilities of modern LLMs and RLHF methods, enabling models to generate more human-like and coherent summaries that go beyond simple extraction.

3.4 Dataset: OpenAI/SummarizeFromFeedback

In this study, we focus on using high-quality datasets tailored for text summarization tasks with **Reinforcement Learning with Human Feedback (RLHF)**. One such dataset is **OpenAI's SummarizeFromFeedback**, which is specifically designed to facilitate RLHF-based training for generative models in the context of text summarization. This dataset is chosen due to its comprehensive feedback structure and its alignment with real-world summarization needs, distinguishing it from other publicly available datasets like HuggingFace's **H4** or **Stack Exchange preferences** that are primarily focused on question-answering tasks.

3.4.1 Dataset Overview: OpenAI/SummarizeFromFeedback

The **SummarizeFromFeedback** dataset contains a collection of text documents paired with human-generated summaries. The key differentiator of this dataset is that it incorporates **human feedback** in the form of ratings, preferences, and choices about the quality of summaries. This feedback is crucial for fine-tuning models via RLHF, enabling the model to learn to generate summaries that align closely with human expectations.

The dataset consists of the following components:

- **Text**: The original document or passage to be summarized.
- **Summary**: The human-generated summary of the original text.
- **Choices**: A set of alternative summaries or output generated by the model, which are then evaluated by human annotators for preference.
- **Ratings/Feedback**: Feedback or ratings given by humans on the quality of the summaries, typically reflecting factors such as relevance, fluency, and coherence.

3.5 Models

In training models for text summarization with **Reinforcement Learning (RL)**, several RL algorithms can be employed, each with its strengths and limitations. For this task, we explore **Proximal Policy Optimization (PPO)** as the primary algorithm for **RLHF**-based fine-tuning. Other RL algorithms like **DQN**, **SAC**, **DDPG**, **TD3**, **A2C**, **A3C**, and **TRPO** were considered but have certain drawbacks in the context of training large generative models.

3.5.1 RL Algorithms Considered

- **TRPO (Trust Region Policy Optimization)**: While TRPO is known for stable updates and convergence, its implementation is complex and computationally expensive. The algorithm's reliance on second-order optimization and constraint handling increases overhead, making it less feasible for large-scale generative tasks like text summarization.

- **DQN (Deep Q-Network), DDPG (Deep Deterministic Policy Gradient), SAC (Soft Actor-Critic)**: These algorithms rely on replay buffers, which store past experiences and sample them for training. While effective in discrete action spaces, they are less suitable for NLP tasks like text summarization, as they fail to capture the sequential nature of text generation and tend to be unstable in continuous action spaces.

- **A2C (Advantage Actor-Critic)**: A2C is simpler and computationally less demanding than PPO but is less efficient for large generative models. It relies on synchronous updates, which make it harder to scale effectively for text summarization tasks.

- **A3C (Asynchronous Advantage Actor-Critic)**: A3C improves training speed by utilizing multiple workers in parallel, but it suffers from stability issues when applied to complex tasks like text summarization, where more nuanced state-action representations are necessary.

3.5.2 Why PPO?

Proximal Policy Optimization (PPO) is the algorithm of choice due to its balance of **sample efficiency**, **stability**, and **computational cost**. PPO is a policy gradient method that helps maintain stable training by using a clipped objective to ensure updates remain within a "trust region." This prevents large, unstable updates, which is crucial when fine-tuning generative models for complex tasks like text summarization.

3.5.3 Computational Challenges and Optimization Strategies

Training large language models like **LLaMA 7B** using PPO can be resource-intensive. For instance, loading two 7B parameter models in **4-bit quantization** requires approximately **7 GB of VRAM per model** for training. Given that text summarization involves more complex generation compared to simpler tasks like text classification, this often led to **CUDA out-of-memory errors**.

To mitigate this, I reduced the **batch size** to 1, allowing for better memory management. To ensure efficient weight updates, I applied **gradient accumulation**, which allowed the model to simulate larger batch sizes without exceeding memory constraints. This approach helped manage the limited VRAM (16GB) and provided stability during training.

In addition, I removed the **optimizer(model.parameter())** approach during debugging, as this could cause unnecessary VRAM consumption by inadvertently passing frozen parameters during training, which was inefficient for memory usage.

3.5.4 Use of Smaller Models for Reward Classification

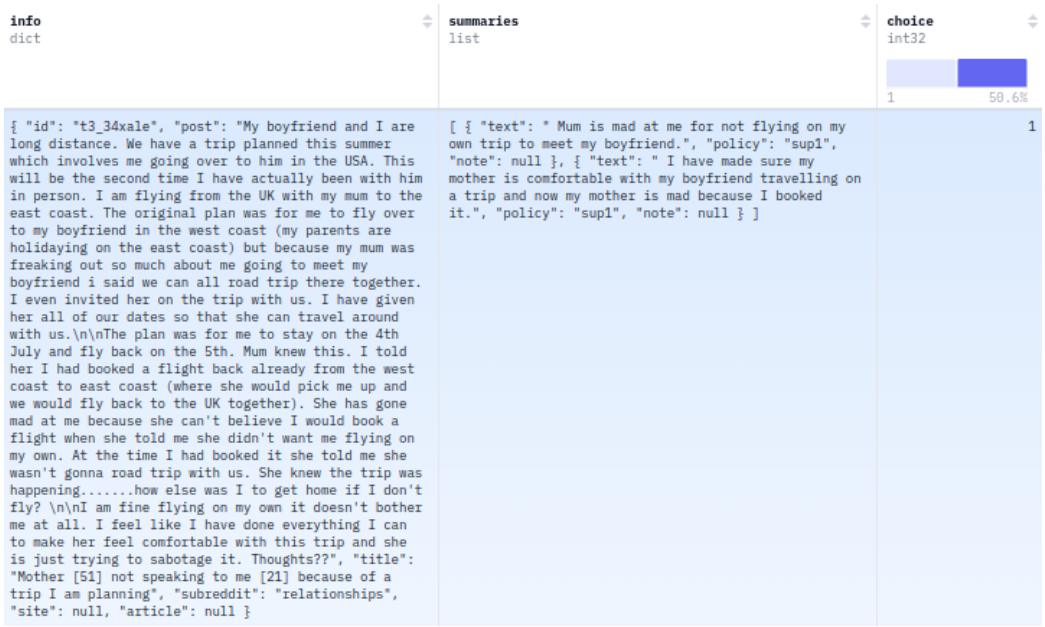
While the main model for text summarization requires large-scale architecture, the reward model can often be smaller and still effectively perform classification tasks. For this, I employed the **Gemma 2B** model, which proved to be efficient for **reward classification**. Unlike the generative task, where large models are necessary for capturing the complexity of language, the reward classification task benefits from smaller models like **Gemma 2B**, which can handle the **text classification** aspect of RLHF efficiently.

3.5.5 Direct Preference Optimization (DPO)

Direct Preference Optimization (DPO) is an advanced approach in reinforcement learning designed to optimize model outputs more efficiently by aligning them directly with human preferences. Unlike traditional reinforcement learning algorithms like Proximal Policy Optimization (PPO), which typically rely on reward signals generated by an auxiliary model (often referred to as the "reward model"), DPO optimizes the model's parameters directly based on human feedback without requiring a separate reward model. This direct optimization makes DPO particularly effective in tasks where human preferences are the primary driving force behind model behavior, such as in natural language processing (NLP) tasks like text summarization.

The key benefit of DPO lies in its computational efficiency. Since it does not rely on an external reward model, DPO reduces the need for additional neural networks, making it less resource-intensive than traditional RL methods like PPO, which typically require training a separate model for reward prediction. This reduction in computational complexity makes DPO an attractive alternative when hardware resources are limited, or when a quicker optimization cycle is desired.

Moreover, DPO offers robustness in its performance, making it an attractive alternative to PPO in resource-constrained environments. The efficiency of DPO ensures that even with limited computational power, high-quality language model training can still be achieved, providing an optimal trade-off between training speed and model performance.



info dict	summaries list	choice int32 1 — 50.6%
{ "id": "t3_34xale", "post": "My boyfriend and I are long distance. We have a trip planned this summer which involves me going over to him in the USA. This will be the second time I have actually been with him in person. I am flying from the UK with my mum to the east coast. The original plan was for me to fly over to my boyfriend in the west coast (my parents are holidaying on the east coast) but because my mum was freaking out so much about me going to meet my boyfriend i said we can all road trip there together. I even invited her on the trip with us. I have given her all of our dates so that she can travel around with us.\n\nThe plan was for me to stay on the 4th July and fly back on the 5th. Mum knew this. I told her I had booked a flight back already from the west coast to east coast (where she would pick me up and we would fly back to the UK together). She has gone mad at me because she can't believe I would book a flight when she told me she didn't want me flying on my own. At the time I had booked it she told me she wasn't gonna road trip with us. She knew the trip was happening.....how else was I to get home if I don't fly? \n\nI am fine flying on my own it doesn't bother me at all. I feel like I have done everything I can to make her feel comfortable with this trip and she is just trying to sabotage it. Thoughts??", "title": "Mother [51] not speaking to me [21] because of a trip I am planning", "subreddit": "relationships", "site": null, "article": null }	[{ "text": " Mum is mad at me for not flying on my own trip to meet my boyfriend.", "policy": "sup1", "note": null }, { "text": " I have made sure my mother is comfortable with my boyfriend travelling on a trip and now my mother is mad because I booked it.", "policy": "sup1", "note": null }]	1

Figure 3.2: Example schema from the OpenAI SummarizeFromFeedback dataset showing document text, summaries, and human feedback for model-generated summaries

Chapter 4

Training Results

4.1 Exploring Monte Carlo with Blackjack Game

To explore the effectiveness of fundamental reinforcement learning techniques, we first implemented a basic **Monte Carlo (MC)** algorithm to solve the Blackjack game from scratch. Blackjack, a popular card game, involves the player attempting to reach a hand value as close to 21 as possible without exceeding it. In the context of reinforcement learning, the agent interacts with the environment by taking actions (hit or stand) to maximize the expected reward.

Monte Carlo methods are used to estimate the value of a state or action by averaging the rewards obtained after multiple episodes. This approach does not require a model of the environment but instead relies on experience gained by interacting with it over time.

The following parameters were used for the Blackjack implementation:

- **Gamma** (γ) = 0.9: The discount factor for future rewards.
- **Alpha** (α) = 0.02: The learning rate for updating the value estimates.
- **Number of episodes** = 1,000,000: The total number of games the agent played to train.
- **Evaluation episodes** = 100,000: The number of episodes used to evaluate the agent's performance after training.
- **Epsilon** (ϵ) = 0.1: The exploration factor for epsilon-greedy action selection.

- **Deck** = [2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11]: The cards available in the Blackjack game.
- **Actions** = [0, 1]: Actions representing 'hit' (0) or 'stand' (1).

After training for a large number of episodes, the following performance metrics were observed for the Blackjack agent:

- **Win Percentage** = 42.424%
- **Draw Percentage** = 5.889%
- **Loss Percentage** = 51.687%

These results are quite comparable to those achieved using well-established libraries like **OpenAI Gym**, demonstrating the effectiveness of Monte Carlo methods in solving this classic RL problem. The Monte Carlo Blackjack agent was able to perform reasonably well, balancing exploration and exploitation to reach near-optimal performance in the game.

4.2 Training RLHF Model with GPT-2

In this section, we explore the training of a Reinforcement Learning with Human Feedback (RLHF) model. The architecture for the RLHF model used **GPT-2** as both the **main model** and the **reward model**. GPT-2 was chosen due to its strong performance in generative text tasks and its suitability for fine-tuning with reinforcement learning.

The model was trained using the **SummarizeFromFeedback** dataset, which includes both text inputs and human feedback for summarization tasks. The training was carried out with standard PPO (Proximal Policy Optimization) and RLHF techniques. However, the results showed that the average reward obtained after training was notably low, hovering close to a value of 2. This low reward suggests that the RLHF setup with GPT-2 did not yield significant improvement, indicating that the combination of GPT-2 and RLHF might require further hyperparameter tuning or more sophisticated reward mechanisms to achieve optimal performance.

4.3 Comparison of Gemma-2B and Gemma-7B for Text Classification

After encountering challenges with the GPT-2-based RLHF model, we turned to evaluating other models, specifically **Gemma-2B** and **Gemma-7B**, for their performance on text classification tasks. We selected a subset of the **IMDB Review Dataset** for comparison. Both Gemma-2B and Gemma-7B were evaluated on standard classification metrics such as **accuracy** and **F1 score**.

The results revealed that the performance gap between Gemma-2B and Gemma-7B for text classification tasks was minimal, with less than a 2% difference in both accuracy and F1 score. This suggests that for tasks like text classification, the differences between smaller and larger models are not as pronounced.

However, when the same models were applied to text generation tasks, such as summarization, **Gemma-7B** outperformed **Gemma-2B**, demonstrating that larger models are better suited for generative tasks. Thus, we conclude that while Gemma-7B excels in generative tasks like summarization, Gemma-2B is comparable to Gemma-7B for classification tasks, making it a more efficient option for such tasks when computational resources are limited.

4.4 Fine Tuning of Gemma-2B Reward Model and LLaMA-3.1 8B Supervised Fine-Tuning

4.4.1 Fine-Tuning of Gemma-2B Reward Model

The first step in the model training process involved fine-tuning the **Gemma-2B** model to serve as the reward model for our RLHF approach. The Gemma-2B model, while efficient in performance, was adapted to classify text for the reward prediction task. This was achieved by fine-tuning the pre-trained Gemma-2B model on a text classification dataset, specifically designed to classify text for summarization tasks. The fine-tuned reward model was trained for **5 epochs** and was used to assign rewards based on the quality of the generated summaries during reinforcement learning.

4.4.2 Supervised Fine-Tuning of LLaMA-3.1 8B Model

Next, we performed supervised fine-tuning on the **LLaMA-3.1 8B** model for the text summarization task. The model was trained for **5 epochs** using a 16GB VRAM setup. Despite the limited computational resources, the fine-tuned LLaMA-3.1 model achieved a **Rouge-1 F1 score of 18%**, which is considered a good result given the constraints. The model learned to generate summaries effectively, although further tuning would be necessary for better performance.

4.4.3 RLHF Fine-Tuning of LLaMA-3.1 8B with Gemma-2B Reward Model

Following the supervised fine-tuning, the **LLaMA-3.1 8B** model was then fine-tuned using Reinforcement Learning with Human Feedback (RLHF). The fine-tuned **Gemma-2B reward model** was used in conjunction with PPO (Proximal Policy Optimization) to fine-tune the LLaMA-3.1 8B model further. The PPO algorithm was trained for **4 epochs**, and after the RLHF fine-tuning process, we observed a slight improvement in the **average reward score** of the PPO-trained LLaMA-3.1 model when compared to the supervised fine-tuned LLaMA-3.1 8B model. While the improvement was modest, it indicated that RLHF techniques were able to refine the model's ability to generate higher-quality summaries, aligning it more closely with human feedback.

4.4.4 Training of DPO Model for Comparison

In addition to the PPO model, we also trained a **Direct Preference Optimization (DPO)** model, which is known for its computational efficiency. The training time for the DPO model was significantly lower, with less than **30% of the training time** required compared to the PPO model. Despite the reduced training time, the loss obtained in the DPO model was comparable to the PPO model, showing that DPO can be a viable alternative for RLHF fine-tuning tasks. This efficiency made DPO a strong candidate for scenarios with limited computational resources.

These results suggest that while PPO provides a slight improvement over supervised fine-tuning, DPO can offer similar performance with considerably reduced computational overhead, making it an attractive option for RLHF in resource-constrained environments.

Chapter 5

Conclusion

This research explored the use of Reinforcement Learning with Human Feedback (RLHF) to enhance text generation tasks like text summarization. For those with ample computational resources, using a large reward model that matches the main LLM model, such as Gemma-7B, is ideal for improving model performance. However, for resource-constrained environments, smaller models like Gemma-2B are still effective, especially for text classification tasks, providing a balance between performance and efficiency.

We also evaluated Direct Preference Optimization (DPO), which proved to be a more computationally efficient alternative to Proximal Policy Optimization (PPO), offering faster training times without compromising performance.

One critical takeaway is the rapid evolution of RLHF libraries, particularly the TRL library. Since many arguments and configurations in these libraries change frequently, it's important to save versions of the libraries used in experiments to ensure code reproducibility over time.

In conclusion, RLHF shows great potential for fine-tuning models to align better with human preferences. Researchers should consider available resources, task requirements, and evolving libraries when selecting the right techniques and frameworks for their work.

Chapter 6

Future Work

This research has provided valuable insights into the application of RLHF for text generation tasks. However, there are several avenues for further exploration and improvement.

One direction for future work involves exploring additional reinforcement learning algorithms, such as Implicit Q-Learning (ILQL), to assess their performance in RLHF tasks. ILQL offers the potential for more efficient learning, and its comparison to existing methods like PPO could lead to further optimizations in model training and reward alignment.

Another aspect to explore is overcoming the resource constraints observed in this research, particularly the 16GB VRAM limit. By scaling up and fine-tuning larger models, we can better assess the capabilities of these models on tasks like text summarization. Larger models may hold the potential for improved performance but require more computational power, which could lead to trade-offs between resource availability and model effectiveness.

Lastly, the application of RL in other domains, such as robotics, represents a promising area for future work. RL can play a crucial role in training robots to perform complex tasks in dynamic environments, and bridging the gap between RLHF and robotics could lead to advancements in autonomous systems, human-robot interaction, and task optimization.

Bibliography

[1] Khandelwal, Amit. "RLHF Reward Model Training." *Towards Generative AI*, 11 Aug. 2023, https://medium.com/towards-generative-ai/reward-model-training-2209d1befb5f.

[2] Illustrating Reinforcement Learning from Human Feedback (RLHF). https://huggingface.co/blog/rlhf. Accessed 29 Nov. 2024.

[3] Snell, Charlie, et al. "Offline RL for Natural Language Generation with Implicit Language Q Learning." *arXiv:2206.11871*, arXiv, 1 May 2023, arXiv.org, https://doi.org/10.48550/arXiv.2206.11871.

[4] Palaniappan, Sri Ranganathan. "Memory Requirements for Fine-Tuning Llama 2." *Polo Club of Data Science — Georgia Tech*, 16 Apr. 2024, https://medium.com/polo-club-of-data-science/memory-requirements-for-fine-tuning-llama-2-80f366cba7f5.

[5] Illustrating Reinforcement Learning from Human Feedback (RLHF). https://huggingface.co/blog/rlhf. Accessed 29 Nov. 2024.

[6] Ouyang, Long, et al. "Training Language Models to Follow Instructions with Human Feedback." *arXiv:2203.02155*, arXiv, 4 Mar. 2022, arXiv.org, https://doi.org/10.48550/arXiv.2203.02155.

[7] Huang, Shengyi, et al. "The N+ Implementation Details of RLHF with PPO: A Case Study on TL;DR Summarization." *arXiv:2403.17031*, arXiv, 23 Mar. 2024, arXiv.org, https://doi.org/10.48550/arXiv.2403.17031.