Investigating Quantity and Quality of HF in RLHF

Woo Jin Choi
Department of Computer Science
Johns Hopkins University
wchoi20@jh.edu

Abstract

The desire for more data is increasing while the amount of clean data is limited. While increasing the scale of language models will increase performance, the cost of scaling increases massively as well. To deviate from obtaining massive data for pre-training, one of the alternative methods that the field has sought is Reinforcement Learning from Human Feedback (RLHF). Despite plenty of recent works and successful implementation of the learning algorithm, it is unclear what kind of human feedback is useful. To investigate the importance of quantity and quality of human feedback (HF) in RLHF, three different models are trained to function as a reward model that produces scalar reward values during the reinforcement learning loop. Consequently, three models are produced; however, the text generation ability of these models is damaged, summarizing the project as an unsuccessful attempt of investigating the significance of human feedback[1]

1 Introduction

Reinforcement Learning from Human Feedback (RLHF) is a training technique that uses reinforcement learning to optimize a language model using human preference. Recent successful implementation of RLHF includes InstructGPT [1] and ChatGPT. Despite plenty of recent works on RLHF that reveals its success, it is unclear what kind of human feedback is useful. This project attempts to investigate the importance of the quantity and quality of human feedback in RLHF.

1.1 Motivation

Supervised and unsupervised learning have been a few of the widely used algorithms to train and fine-tune a pre-trained language model. It has been demonstrated to be successful in manipulating language models to perform desired tasks. However, there still exist limitations in various domains. Such domains include generating creative outputs to limit the toxicity of the product. Interested in discovering a technique to improve models, I decided to conduct this project on human feedback.

1.2 Related Works

1.2.1 Learning to summarize from human feedback [1]

This paper applied RLHF to train a language model that better summarizes the text than a larger language model. Using the Reddit TL;DR dataset to develop a reward model and implementing an RL training policy with PPO, the model trained with RLHF was able to outperform larger models in text summarization tasks.

[1]The source code is available at `https://github.com/wchoi20/CS-601.471-final-project`

1.2.2 Training language models to follow instructions with human feedback [2]

This paper applied RLHF to fine-tune GPT-3 and developed InstructGPT, which demonstrates a better performance than the original LM while showing improvements in truthfulness and reductions in toxic output generations.

1.2.3 Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback [3]

This paper applied preference modeling and RLHF to fine-tune models, which improved the model's performance on almost all NLP evaluations. I have adopted the training techniques and data from this paper on a smaller scale on a different model (as the model the paper used was their original LM).

1.2.4 Illustrating Reinforcement Learning from Human Feedback (RLHF) [4]

This article from HuggingFace on RLHF introduces how RLHF can be used to fine-tune large language models to produce more desirable outputs as defining or creating a loss function for some subjective categories. The article then provides popular techniques for implementing RLHF on language models.

2 Approach

My approach to this project follows a conventional reinforcement learning loop. That is, an agent model learns from an environment through a reward produced by a reward model.

2.1 Datasets

There are three datasets used in my approach to investigating the quantity and quality of human feedback. One dataset is used as a human feedback in training a reward model and two other datasets are used as a query in reinforcement training loop.

2.1.1 Human preference data about Helpfulness and Harmlessness

Collecting human feedback is an expensive and complicated task. Instead of manually generating a dataset, I used 'human preference data about helpfulness and harmlessness [3]' [2] as an alternative. The dataset contains 42,537 rows in which each row contains a pair of text that is either 'chosen' or rejected'. Each text follows the format of a human asking or instructing an assistant. Then the assistant responds, which is evaluated by humans to decide which response is better. The better response is marked as 'chosen'.

2.1.2 SQuAD2.0 and TriviaQA

To follow the query + response format and to fine-tune a language model to perform question-answering tasks for evaluation, I decided to use SQuAD2.0 [5] [3] and TriviaQA [6] [4] as datasets during the reinforcement learning cycle. Two datasets above are chosen instead of datasets like wikitext [7] used for other tasks like text generation in order to be able to evaluate the trained model on question-answering tasks as it was done in the previous relevant work [3].

2.2 Methods and Experiment Details

Training a language model under the reinforcement learning technique can be formulated into three steps [4]. I follow these steps while repeating the second step to achieve the objective of this project. All training was done on NVIDIA A100 from either Rockfish or Google Colab.

[2]The dataset is available at `https://huggingface.co/datasets/Anthropic/hh-rlhf`

[3]The dataset is available at `https://huggingface.co/datasets/squad_v2`

[4]The dataset is available at `https://huggingface.co/datasets/trivia_qa`

2.2.1 Pretraining a Language Model

The first step requires selecting which language model to function as a base model that will act as an agent. Possible options include pretraining a language model from scratch or utilizing available transformer models as pre-trained language model.

Pretraining a language model from scratch would be an unnecessary step considering the scope and the computation limit of this project. Thus, I decided to use the Transformer library available from HuggingFace. My original choice for the pre-trained large language model was T5-base. The choice was made considering its competitive performance on generative tasks with its text-to-text framework [8]. However, after receiving feedback that the pretraining objective of the T5-base model and its unavailability to generate text without direct training to generate text, I decided to use GPT-2 [9] instead. Among GPT-2 families available, I chose GPT-2 as it is the smallest version of GPT-2. Scaling language models have already been shown to improve their performance on various tasks. But because the objective of this project is to discover the actual influence of human feedback, I believe that the size of a model should not present a significant issue.

Choosing an appropriate large language model is important for this project as it will be used in two instances:

- A base model to be an agent in the reinforcement learning loop.
- A reference model to penalize outputs that deviate from natural language (and our task).

2.2.2 Training a Reward Model

In addition to choosing an agent model (as was done in the previous step), a base model for the reward model needs to be selected. Training a reward model is an important step in RLHF as it provides a guideline for an agent's future action. There are multiple ways to generate rewards for an agent to receive feedback from the environment. Possible methods include a function that outputs a scalar value as a reward, synchronous decisions from human participants, or a model that outputs a scalar value as a reward. In this project, I selected BERT [10] as a base model to be trained as a reward model. The model is then trained as a binary classifier in which two labels are derived from 'chosen' and 'rejected' from 2.1.1.

In the earlier stages of the project, I initially used GPT-2 as a base for the reward model. However, considering that the initial purpose of training a language model to function as a binary classifier is to produce scalar outputs, I revoked and chose BERT as a base model as bert-base-uncased has 110M parameters [10] while the smallest version of GPT-2 has 124M parameters [9].

Before entering the training phase, I preprocessed that data from 2.1.1 by manually adding labels - 'chosen' texts were labeled as 1, and 'rejected' texts were labeled as 0. Thus, the model was trained as a binary classifier. The difference in the dataset provided to BERT is the main experimental action in this project. The followings are the three reward models generated by altering datasets:

- Model A: A model trained with the original data (42,537 rows)
- Model B: A model trained with half of the original data (21,269 rows)
- Model C: A model trained with the original data with some incorrect labels (42,537 rows)

In training the third model, which corresponds to validating the quality of human feedback, each row had a 1⁄20 chance of labeling the row incorrectly by obtaining an opposite label. To keep the consistency of the experiment, hyperparameters were kept the same. All models were trained using AdamW optimizer. The batch size was 32 and the learning rate was 0.001. Only a single epoch was used to avoid overfitting models to human feedback. Logits for the 'chosen' text from the reward model were later used as a scalar reward value in the reinforcement learning loop.

2.2.3 Fine-Tuning the Language Model with Reinforcement Learning

As mentioned in 2.2.1, two GPT-2 models were prepared for the reinforcement learning loop. One model served as an agent while the other functioned as a reference model to ensure the generated text does not deviate from the original behavior extensively.

Before beginning the reinforcement learning, I preprocessed data from 2.1.2 to build a dataset that contains a query, which will become a prompt to both the agent and reference model. I extracted only the 'questions' column as other information was unnecessary in generating queries for models. Initially, an entire dataset was used. But due to the poor results produced, the dataset was broken into a smaller shard, which contains 200 queries to be asked to each model.

To fine-tune a model with reinforcement learning, Transformer Reinforcement Learning (TRL) library [11] is used. The library facilitates the training of transformer language models with Proximal Policy Optimization (PPO), a widely chosen policy-gradient algorithm [12].

The learning algorithm begins with getting a response from the agent model. A batch of queries is provided to the agent model to generate texts, which are considered responses. Then queries and responses are combined and provided to the reward model from 2.2.2. The reward model generates logits for two categories: 'chosen' and 'rejected'. For this project, I used logits for 'chosen' as a scalar reward value. Then the optimization step begins. Log probabilities of both the agent and reference model for the combined inputs (queries and responses) are used to calculate Kullback-Leibler (KL) divergence to penalize an action that diverges from the reference model. This step is necessary to prevent reward-striving outputs. Lastly, the final step is sending the sum of a scalar reward and KL divergence value to optimize the model. In terms of the configuration (PPOConfig), default values were used while reducing the batch size to 16 and setting optimize-cuda-cache as 'True' due to the computation limit. The number of training epochs was two.

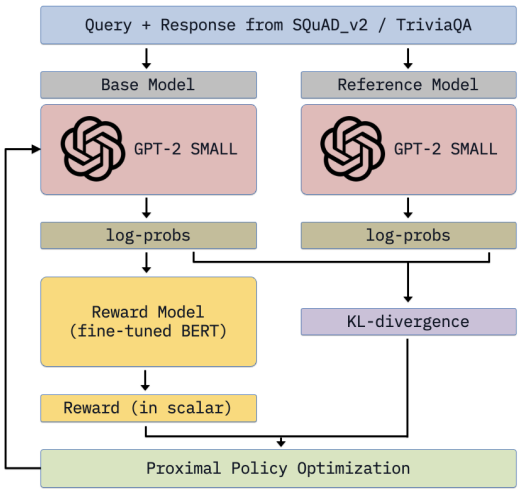

Figure 1: Model architecture for reinforcement learning.

3 Results

Models were trained by using NVIDIA A100, as mentioned in 2.2. The time taken to train those models is listed in Table 1.

Table 1: Training time for models

	Model A	Model B	Model C
Time	2:44:44	2:56:28	2:54:05

To evaluate the importance of quantity and quality of human feedback in RLHF, I planned to compare the performance of three models from 2.2.3 by evaluating on 'test' portion of the dataset from 2.1.2. However, models generated texts that deviated from my expectation. The models repetitively produced the word *Question*, *Integ*, or *Reply* instead of producing comprehensible texts. As the generated outputs of the model demonstrated the current performance of the models, an initial plan had to be modified.

3.1 Alternative Methods

To improve the performance of the models fine-tuned by reinforcement learning, I modified the hyperparameters of the training. First, I reduced the number of epochs in 2.2.3 from 2 to 1. This adjustment was made as I believed that the model was overfitting to the format of the input as it was a concatenation of query and response. However, this attempt remained unsuccessful as the models still failed to generate useful texts.

Next, I reduced the number of data in the datasets in 2.1.2 that was used as a query in 2.2.3. This adjustment was an extension of the modification above. The reduced number of data represents the reduced training of each model using reinforcement learning. The training time for models has decreased by approximately 30%. Models now have started to generate texts that resemble the human language.

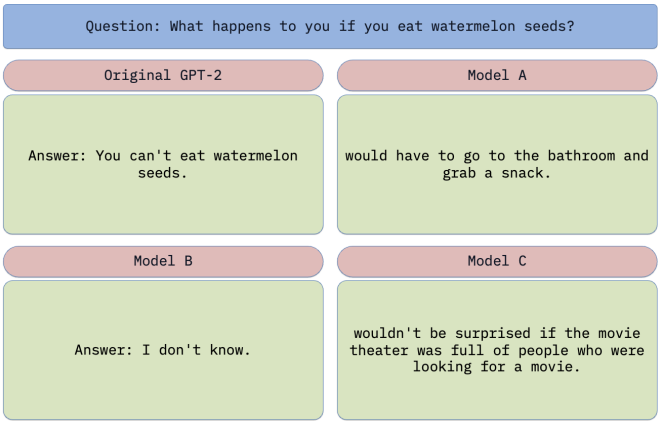

Figure 2: An example text-generation task given to four models, including an original GPT-2 for baseline comparison.

Looking at the generated texts, it is observable that each model generates different texts, signifying the influence of fine-tuning with reinforcement learning. However, whether reinforcement learning is a significant part of the model remains a question.

3.2 Evaluating a Causal Language Model

In addition to the depressing performance exhibited by the original models, an issue regarding evaluation has risen. As aforementioned in 2.2.1, GPT-2 was selected. GPT-2 is a causal language model that performs well on text-generation tasks. However, formulating an evaluation method, as planned, was complicated as the model is not particularly trained in question-answering tasks.

An initial plan of the evaluation was to use a question-answering task as the project was designed to follow some steps of the pre-existing work [3]. In that work, the performance of the models was evaluated on question-answering tasks, using datasets including TriviaQA [6] and OpenBookQA [13].

The first attempt to evaluate was to manipulate question-answering tasks as a text-generation task by inputting questions in the following format: "Question: <actual question> Answer: ". The attempt was unsuccessful as the models generated answers that were far from correct. This led to the question: should the models be fine-tuned once more on question-answering? If so, should the training use reinforcement learning, unsupervised or supervised learning? Here, I decided to not train my fine-tuned models as the importance of quantity and quality of human feedback in reinforcement learning may be influenced.

4 Conclusion

Reinforcement learning may produce unstable results, compared to unsupervised and supervised learning as an agent modifies its action to maximize a reward from an environment. Despite the formulated layout of the experiment, all models, with the initial training methods, were not able to generate texts that are valuable to be evaluated. There are multiple possible reasons that the project has failed to produce successful outcomes. For instance, the generation of a reward value may have an error. As mentioned in 2.2.2, logits for the 'chosen' text is used as a scalar reward value. Another choice of a reward value or using a reward baseline to lessen the degree of reward to the agent may have been necessary. Another possible reason is the preprocessing of the dataset used in training a reward model. Each row in a dataset mentioned in 2.1.1 includes some overlapping texts for 'chosen' and 'rejected', which could have been taken into account to train a reward model more accurately. Another possible reason is the type of dataset used as a query in 2.2.3. That is, a question-answering type of dataset may not be a suitable dataset in reinforcement learning. To maintain the format of such a task, adding an indicator for question and answer is necessary. However, because such an indicator is present in the entire input text, the agent could have interpreted it as a key action to increase the reward in the learning process.

Despite multiple hypotheses on the failure to discover meaningful findings by this project, it is difficult to point out which particular step influenced the depressing performance of the models. This project, unfortunately, was not able to fully investigate the importance of quantity and quality of human feedback in RLHF while demonstrating the existence of the influence of quantity and quality of human feedback for future works.

References

[1] Stiennon, Nisan, et al. "Learning to Summarize from Human Feedback." ArXiv.Org, 2 Sept. 2020.

[2] Ouyang, Long, et al. "Training Language Models to Follow Instructions with Human Feedback." ArXiv.Org, 4 Mar. 2022.

[3] Bai, Yuntao, et al. "Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback." ArXiv.Org, 12 Apr. 2022.

[4] Lambert, Nathan, et al. " Illustrating Reinforcement Learning from Human Feedback (RLHF)"

[5] Rajpurkar, Pranav, et al. Know What You Don't Know: Unanswerable Questions for SQuAD. arXiv, 11 June 2018.

[6] Joshi, Mandar, et al. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. arXiv, 13 May 2017.

[7] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer Sentinel Mixture Models

[8] Raffel, Colin, et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv, 28 July 2020.

[9] Radford, Alec, et al. "Language Models are Unsupervised Multitask Learners." 2019.

[10] Devlin, Jacob, et al. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. arXiv, 24 May 2019.

[11] von Werra, Leandro, et al. TRL: Transformer Reinforcement Learning. 2020.

[12] Ziegler, Daniel M., et al. "Fine-Tuning Language Models from Human Preferences." ArXiv.Org, 18 Sept. 2019.

[13] Mihaylov, Todor, et al. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. arXiv, 8 Sept. 2018.