RLHF(PPO) vs DPO

BavalpreetSinghh Follow 9 min read · Jun 8, 2024

Although large-scale unsupervisly trained language models (LLMs) gain broad world knowledge and some reasoning abilities, precisely controlling their behavior is difficult because of the fully unsupervised nature of their training.

Existing methods like Reinforcement Learning from Human Feedback (RLHF), also known as policy tuning, for gaining such steerability is a complex process. It involves initially training a reward model to reflect human preferences, and then fine-tuning the large unsupervised language model using reinforcement learning to maximize this estimated reward, all while ensuring the model does not deviate significantly from its original state. By that being said let's delve into these concepts step by step. We'll start by understanding what RLHF is, followed by reward modeling. Next, we'll explore the importance of policy in reinforcement learning, and then discuss PPO and DPO, including the mathematics behind them.

Source: DALL-E 3

What is RLHF?

Reinforcement Learning from Human Feedback (RLHF) is a machine learning technique designed to optimize models based on human feedback. Unlike traditional reinforcement learning, which relies solely on predefined reward functions, RLHF incorporates direct human input into the reward

function. This integration allows the model to align more closely with human goals, preferences, and needs.

How Does RLHF Work?

RLHF involves several key stages to refine a language model:

1. **Data Collection:**

- A set of human-generated prompts and responses is created. These serve as the training data for the model.
- For example, a prompt might be, "What is the approval process for social media posts?" and a human knowledge worker provides an accurate, natural response.

2. **Supervised Fine-Tuning:**

- A commercial pretrained model is fine-tuned with this human-generated data, using techniques like retrieval-augmented generation (RAG) to adapt the model to specific contexts.
- The responses generated by the model are compared to human responses, with scores assigned based on similarity and accuracy.

3. Building a Reward Model:

- Human evaluators rate the quality of the model's responses to various prompts, indicating which responses are more aligned with human preferences.

- This feedback is used to train a reward model that estimates the quality of responses.

4. Policy Optimization:

- The language model uses the reward model to refine its response generation policy through reinforcement learning. The aim is to maximize the reward signal, producing responses that better meet human preferences.

If you have gone through the these papers — paper1 and paper2, You might be wondering what is the policy they are talking about?

The Importance of Policy in RLHF

In the context of RLHF, "policy" refers to the set of rules or strategies that a language model follows to generate responses. There are two key aspects to consider:

1. **Policy of the Language Model (LM):**

- This is the main policy that dictates how the LM generates responses to prompts.
- Initially developed through supervised learning, it is further refined using reinforcement learning to maximize rewards based on human feedback.

2. Role of the Reward Model:

- Although the reward model itself does not have a policy, it critically influences the LM's policy.
- By predicting the quality of responses based on human feedback, the reward model provides the necessary reward signals for optimizing the LM's policy.

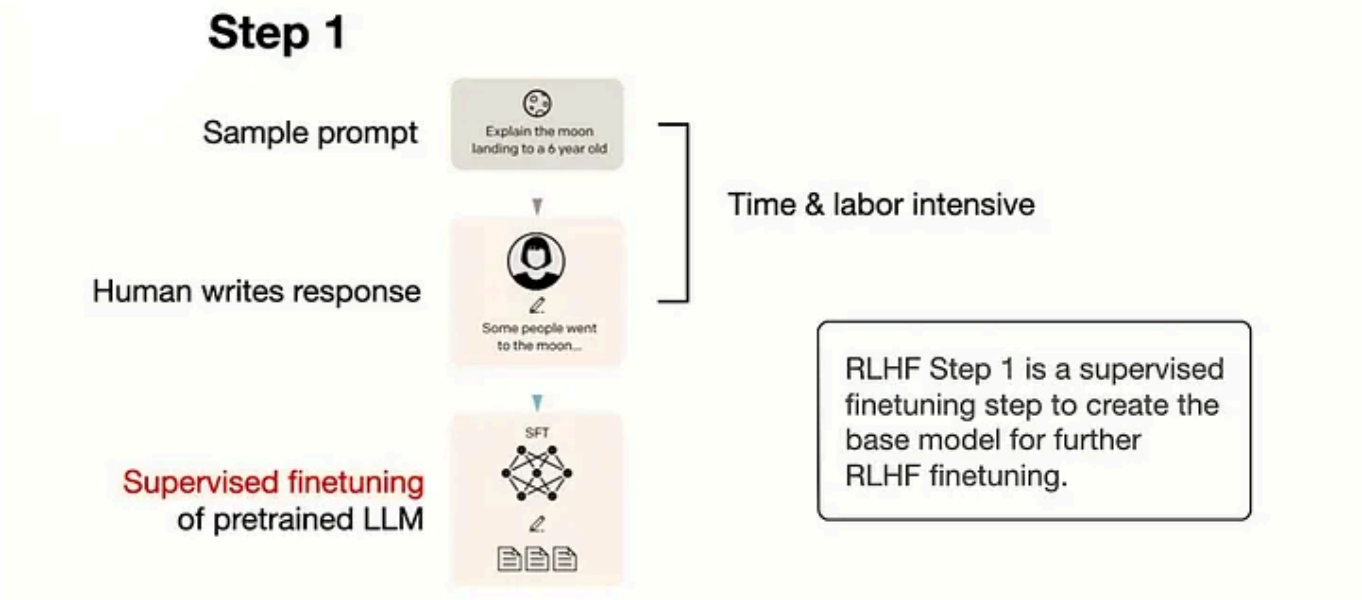
Now that you have got an clear idea what RLHF is and what is the importance of policy, let's jump towards learning what is PPO?

Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm commonly used in the RLHF process. Here's how PPO integrates into RLHF:

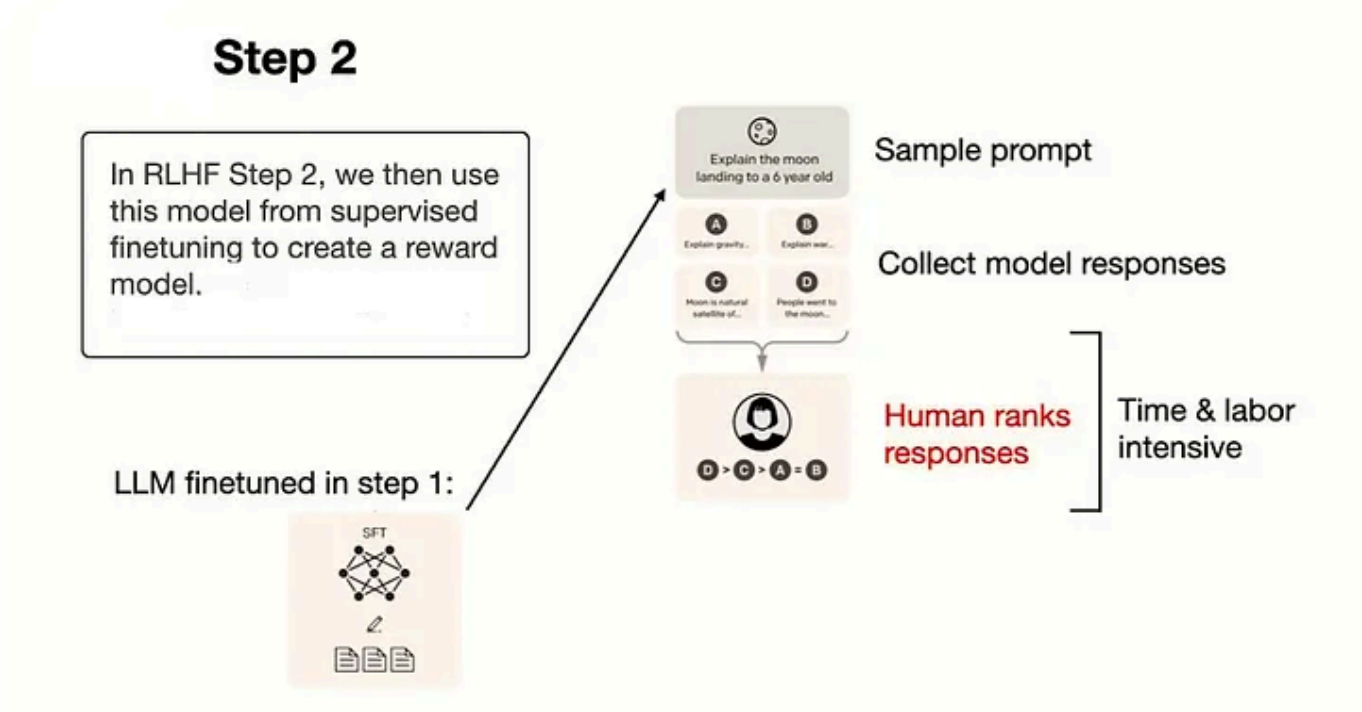
Initialization: Start with a pre-trained language model fine-tuned through supervised learning.

Data Collection: Generate responses using the current policy and collect human feedback on these responses.



Source : Paper

Reward Model Training: Train the reward model to estimate the quality of responses based on human feedback.

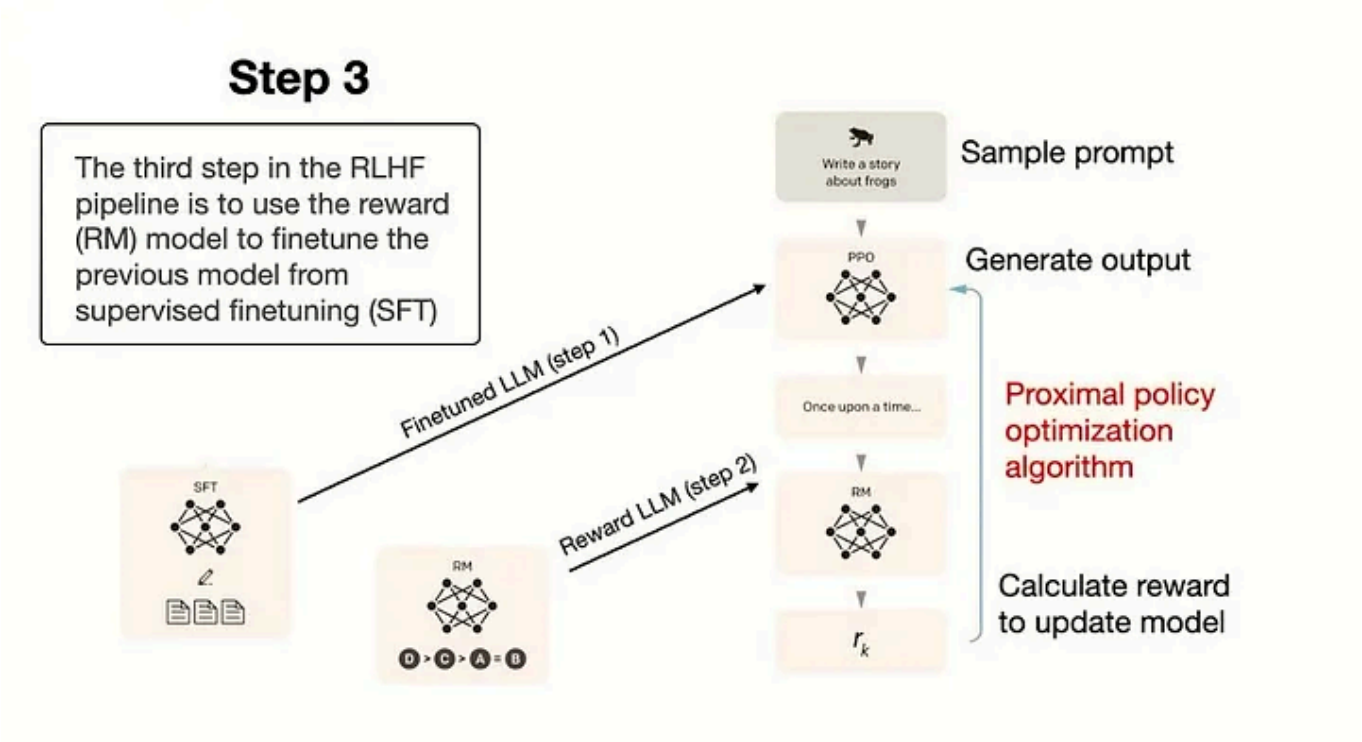


Source : Paper

Policy Optimization with PPO:

- **Sample Collection:** Generate responses and gather data on their associated states and estimated rewards.
- **Advantage Estimation:** Calculate the “advantage” of each response, determining how much better or worse a response is compared to the average.
- **Policy Update:** Adjust the LM’s policy to maximize the expected reward using PPO’s objective function.

- **Clipping Mechanism:** Ensure stable learning by preventing drastic changes in the policy through PPO's clipping mechanism.



Source : Paper

Iterative Refinement: Repeat the data collection, reward model training, and policy optimization steps, continuously refining the LM's policy to produce more human-aligned responses.

Understanding the Math behind PPO (Proximal Policy Optimization)

A. Bradley-Terry Model and Optimal Probability Distribution :

The Bradley-Terry model is used to represent human preferences as probabilistic rather than deterministic. Here's what the variables mean:

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}. \quad (1)$$

Source: Paper

- ***p****: The optimal probability distribution representing true human preferences.
- **y_1 and y_2:** Two different completions (responses) from the language model that we are comparing.
- **x:** The prompt given to the language model.
- ***r****:The optimal reward function that helps the model learn the true human preferences.

Equation 1 from the paper shows the relationship between these variables, indicating that the probability of a completion being preferred depends on this optimal reward function.

Simplified Explanation: Imagine you have two possible endings for a story (y_1 and y_2) given the same beginning (x). The model needs to decide which ending humans would prefer. The optimal reward function (r*) helps the model learn this preference.

B. Training the Reward Model (Equation 2):

Since it's difficult to know the perfect human preference distribution (p*), we train a reward model (rϕ) to approximate it. Here's what the variables mean:

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right] \quad (2)$$

Source : Paper

- **rϕ:** The reward model we are training.
- **D:** A set of training samples showing human preferences.
- **yw:** The preferred completion.
- **yl:** The dispreferred completion.

Equation 2 is used to train the reward model by comparing preferred and dispreferred completions, treating it as a binary classification problem.

Simplified Explanation: We have a bunch of examples where humans have shown which ending they prefer (yw) and which they don't (yl). By using these examples, we train our reward model to predict these preferences correctly.

C. Fine-tuning the Language Model with KL Divergence (Equation 3):

Once we have the reward model trained, we use it to fine-tune the language model. Here's what happens:

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(y|x)}\left[r_\phi(x,y)\right] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi_\theta(y \mid x) \,||\, \pi_{\mathrm{ref}}(y \mid x)\right] \quad (3)$$

Source: paper

- We adjust the language model's policy (πθ) to maximize the reward given by our reward model.
- We compare the new policy (πθ) to the old policy (πref) using KL divergence to prevent the model from changing too drastically.

KL Divergence, short for Kullback–Leibler divergence, is a metric used to quantify the difference between two continuous probability distributions. For further details, you can learn more by following this link.

Simplified Explanation: Imagine the language model is a chef who has learned to make a decent dish. We give feedback (rewards) on how to make the dish better. While the chef tweaks the recipe, we make sure they don't change it too much, so the dish still retains its original good qualities.

Why is this Important?

- **Efficiency:** The model is already well-trained, and we don't want to throw away all that hard work. Instead, we fine-tune it carefully.
- **Maintaining Quality:** By using KL divergence, we ensure the model doesn't lose its original capabilities while improving its performance based on human preferences.

Weakness of this Method:

The major downside is that this approach requires training a separate reward model, which is costly and requires a lot of additional data.

Training a whole new model to give feedback (rewards) to the original language model is expensive and data-intensive. This is the main challenge of this method.

Direct Preference Optimization (DPO): A Simpler Alternative

While RLHF using PPO is effective, it is also complex and computationally intensive. A new approach called Direct Preference Optimization (DPO) simplifies this process by directly optimizing the language model to adhere to human preferences without explicit reward modeling.

How DPO Works:

Preference Data Collection: Similar to RLHF, DPO starts with collecting human preferences over pairs of model responses.

Implicit Reward Model: Instead of explicitly training a reward model, DPO fits an implicit reward model through a simple classification objective, using a binary cross-entropy loss function.

Policy Update: DPO updates the policy by increasing the relative log probability of preferred responses over dispreferred ones. This is achieved with dynamic, per-example importance weighting to prevent model degeneration.

Optimization: By defining the preference loss directly as a function of the policy, DPO can optimize the policy using straightforward training techniques, avoiding the complexities of reinforcement learning.

The Math behind DPO

A. Deriving the Ideal Policy: By adding the KL constraint, we can derive an ideal policy (πr) that maximizes a KL-constrained rewards model. The detailed algebraic derivation is provided in the paper(Check appendix A.1, A.2 and A.3), but the important outcome is that we can write the policy πr in terms of a simpler reward function r.

B. Equation 4: This equation gives us a policy πr that maximizes the reward function with the KL constraint. This step simplifies the optimization process by directly working on the policy.

$$\pi_r(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right), \tag{4}$$

Source: Paper

C. Solving for the Reward Function (Equation 5): We solve for the reward function r, which allows us to replace each instance of r in the ideal probability distribution equation with the derived formula.

$$r(x, y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x). \tag{5}$$

Source: Paper

D. Rewriting the Ideal Probability Distribution (Equation 6): By substituting the reward function derived in Equation 5 into the ideal probability distribution equation (Equation 1), we show that we can optimize the policy directly to match human preferences without needing a separate reward model.

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)} \tag{6}$$

Source: Paper

Understanding the Final Equation (Equation 7)

Loss Optimizing Function: The final equation (Equation 7) is the loss function we use to optimize the policy. Here's a breakdown of the key components:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]. \quad (7)$$

Source: Paper

- **πref:** The old policy (before fine-tuning).
- **πθ:** The new policy (after fine-tuning).
- **yw:** The winning (preferred) completion.
- **yl:** The losing (dispreferred) completion.

Optimizing the Policy: We compare the probabilities that the old policy (πref) and the new policy (πθ) assign to the winning and losing completions. The goal is to optimize the new policy so that it assigns higher probabilities to the winning completions, indicating better alignment with human preferences.

Recap:

PPO Approach:

- Train a separate rewards model to predict human preferences.
- Use this rewards model to fine-tune the language model.

DPO Approach:

- Directly derive the optimal policy using the KL constraint, eliminating the need for a separate rewards model.
- Use the derived formula to directly optimize the policy of the language model.

References

- *DPO Trainer*. (n.d.). Huggingface.Co. Retrieved June 8, 2024, from https://huggingface.co/docs/trl/v0.7.10/en/dpo_trainer
- Lambert, N., Pyatkin, V., Morrison, J., Miranda, L. J., Lin, B. Y., Chandu, K., Dziri, N., Kumar, S., Zick, T., Choi, Y., Smith, N. A., & Hajishirzi, H. (2024). RewardBench: Evaluating reward models for language modeling. In *arXiv [cs.LG]*. http://arxiv.org/abs/2403.13787
- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback. In *arXiv [cs.CL]*. http://arxiv.org/abs/2203.02155
- Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., & Finn, C. (n.d.). *Direct preference optimization: Your language model is secretly a reward model*. Arxiv.org. Retrieved June 8, 2024, from http://arxiv.org/abs/2305.18290

Llm | Fine Tuning | Model | Training | NLP

Written by BavalpreetSinghh

496 followers · 49 following

Consultant Data Scientist and AI ML Engineer @ CloudCosmos | Ex Data Scientist at Tatras Data | Reseacher @ Humber College | Ex Consultant @ SL2

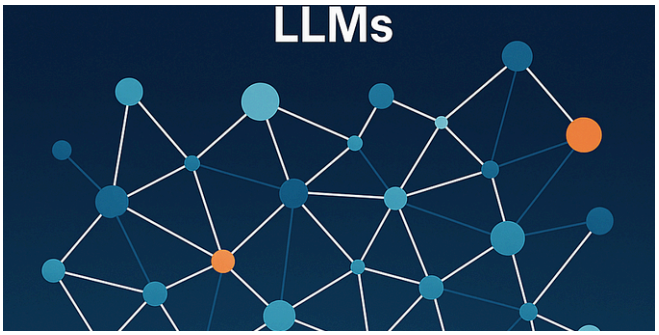
Follow

No responses yet

Scott Lai

What are your thoughts?

More from BavalpreetSinghh

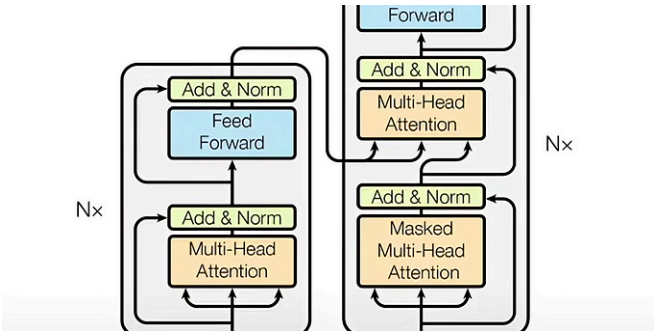


In AI Advances by BavalpreetSinghh

Smart Retrieval Meets Graphs: Neo4j, Kùzu & the Rise of Agentic...

Retrieval-Augmented Generation (RAG) has emerged as a powerful approach to make...

Jun 29 423

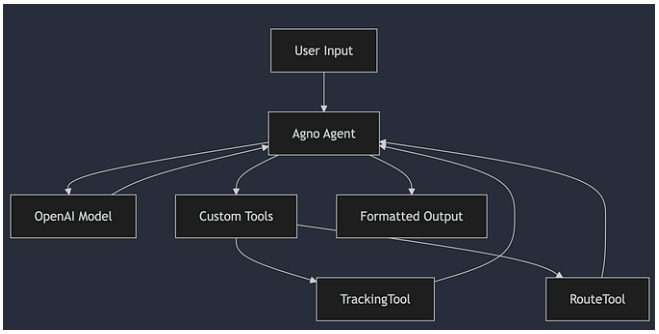


BavalpreetSinghh

Transformer from scratch using Pytorch

In today's blog we will go through the understanding of transformers architecture....

Jun 15, 2024 690 5



In Artificial Intelligence in Plain ... by BavalpreetSi...

Building an AI Agent with Agno: A Step-by-Step Guide

Introduction—In this developer-focused tutorial, we will guide you through building a...

Mar 10 426

BavalpreetSinghh

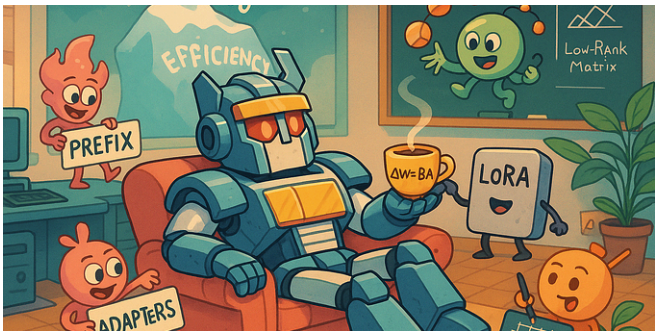
LlamaIndex: Chunking Strategies for Large Language Models. Part...

In the previous blog, we delved into the intricacies of constructing and querying...

Mar 2, 2024 481 2

See all from BavalpreetSinghh

Recommended from Medium

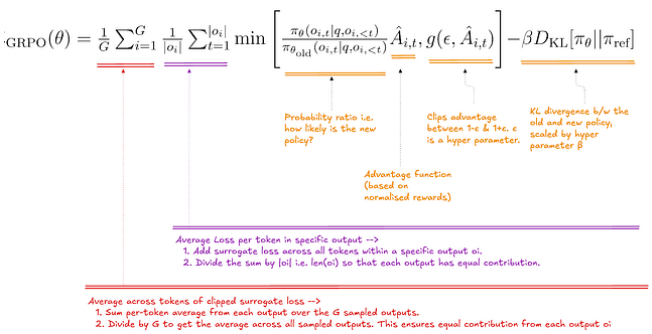


In Foundation Models Deep Dive by M

Parameter-Efficient Fine-Tuning for LLMs: LoRA, QLoRA, and Beyond

Why retrain a giant LLM when a tiny tweak will do?

Jun 19 1

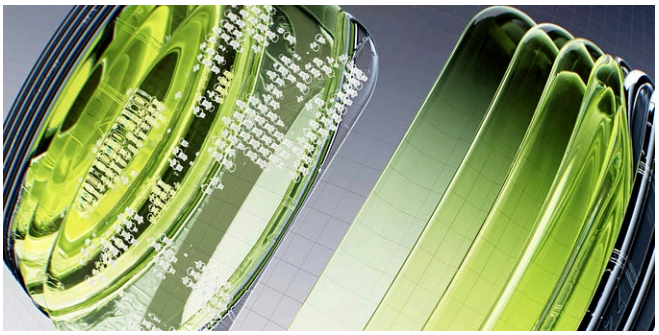


In Yugen.ai Technology Blog by Yugen.ai

Understanding the Math Behind GRPO — DeepSeek-R1-Zero

The Policy Optimisation algo in DeepSeekMath & DeepSeek's first gen…

Feb 6 158 2



In Artificial Intelligence in Plai… by Olubusolami S…

I Finally Understood "Attention is All You Need" After So Long. Here…

It's been almost 2 years since I first encountered the "Attention is all you need"…

Jul 12 178

In TDS Archive by Shirley Li

DeepSeek-V3 Explained 1: Multi-head Latent Attention

Key architecture innovation behind DeepSeek-V2 and DeepSeek-V3 for faster…

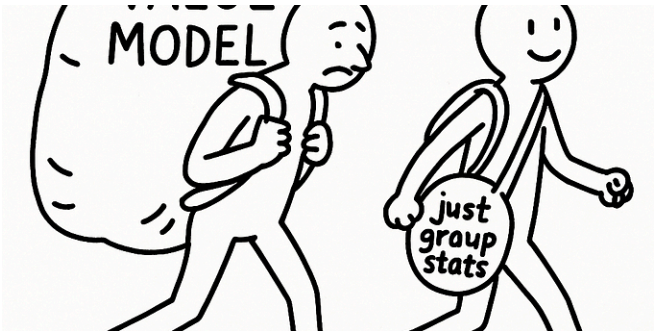
Jan 31 387 12

Arash Nicoomanesh

Fine-Tuning DeepSeek R1 Reasoning on Medical Chain of…

Apply reasoning models to amplify most valuable clinical skill: clinical judgment and…

Jun 29 3



In Better ML by Jaideep Ray

Group Relative Policy Optimization (GRPO): the deep-seek cheat-code

TL;DR—GRPO swaps PPO's expensive value network for a simple "how-do-I-rank-inside-…

3d ago

See more recommendations