Grenze International Journal of Engineering and Technology, June Issue

RLHF: Reinforcement Learning using Human Feedback for Optimization of ChatGPT

Pranav K. Dalvi[1] and Kirti Y. Digholkar[2]
[1-2]Department of Information Technology, Pune Institute of Computer Technology, Pune, India
Email: pranavdalvi2003@gmail.com, kydigholkar@pict.edu

***Abstract*— Reinforcement learning (RL) is a subfield of machine learning that trains agents to make decisions in an environment to maximize rewards. While GPT models like ChatGPT are powerful, they're primarily trained using unsupervised learning on text data, not RL. RL involves agents interacting with an environment, receiving rewards, and learning to maximize long-term rewards. RL can be used to train chatbots by defining actions, environments (like simulated conversations), and rewards based on response quality. In the case of ChatGPT, RL could potentially be used as a component of a broader training pipeline to fine-tune and optimize its responses. On comparing various RL algorithms suitable for ChatGPT, we compared various performance metrics and found that it can be optimized to generate better outputs. As a result, an algorithm was discovered to make ChatGPT a better version of itself.**

***Index Terms*— Reinforcement Learning (RL), ChatGPT, Machine Learning**

I. INTRODUCTION

Reinforcement Learning (RL) is a type of machine learning that involves an agent interacting with an environment to maximize a reward signal. In RL, the agent receives feedback in the form of rewards or penalties, which it uses to adjust its behaviour over time. The agent learns to take actions that lead to the most favourable outcomes by trial and error. The key components of reinforcement learning are the agent, the environment, the action space, the state space, and the reward function. ChatGPT is indeed an application of reinforcement learning. It is a language model developed by OpenAI[1], which uses reinforcement learning to improve its language generation capabilities. The reinforcement learning framework is applied to fine-tune ChatGPT and makes it generate more desirable and contextually appropriate responses. However, there are some problems in ChatGPT that reinforcement learning can help address. Let's dive deeper into the role of reinforcement learning in addressing the challenges in ChatGPT and how it works in each step of the training process:

1.Generating High-Quality Responses: - Problem: ChatGPT may occasionally generate responses that are off-topic, factually incorrect, or irrelevant to the user's query, resulting in a less satisfactory user experience. - RL Solution: In the reinforcement learning phase, ChatGPT is fine-tuned to encourage the model to provide high-quality responses. The model is exposed to various user interactions, and it learns to receive positive rewards when it generates accurate, contextually relevant, and coherent answers. These rewards promote better response quality, ensuring that the model aligns with user expectations.

2.Toxicity and Unintended Content: - Problem: ChatGPT may inadvertently produce toxic, harmful, or inappropriate content, posing ethical and safety concerns. - RL Solution: Reinforcement learning is applied to mitigate toxicity issues. The reward model, which is constructed based on human rankings, assigns lower rewards

Grenze ID: 01.GIJET.10.2.92

3. Consistency and Coherence: - Problem: ChatGPT sometimes generates responses that lack consistency and coherence within a conversation, leading to disjointed interactions. - RL Solution: Reinforcement learning ensures that ChatGPT maintains consistency and coherence throughout a conversation. The model learns to provide responses that flow naturally from the preceding messages and maintain logical continuity. Positive rewards are given for maintaining a conversational thread, making the overall interaction more understandable and engaging.
4. Ambiguity Handling: - Problem: ChatGPT may struggle to handle ambiguous user queries or to seek clarifications when user inputs are unclear. - RL Solution: Reinforcement learning encourages ChatGPT to effectively handle ambiguity. The model learns to ask clarifying questions or provide more context when user inputs are unclear, leading to more accurate and contextually relevant responses. This helps in improving user satisfaction and understanding.

II. Training Process

ChatGPT's training process leverages reinforcement learning in three key steps:
Step 1: Collecting Demonstration Data – In this phase, supervised learning[2] is used, where human trainers engage in conversations, playing both the user and assistant roles. They have access to responses suggested by the model, enabling them to compose appropriate responses. The model is fine-tuned on this dataset, combined with an existing dataset, which is transformed into a dialogue format.
Step 2: Collecting Comparison Data and Training a Reward Model – Human labellers rank the quality of text samples generated by ChatGPT. These rankings are used to construct a reward model that quantifies human preferences for different responses. The model generates various text samples in response to input prompts, and these are ranked by human labellers. The reward model is then trained to predict these rankings.
Step 3: Optimizing a Policy – Reinforcement learning is applied to fine-tune ChatGPT's policy.[3] The reward model from Step 2 is used to guide the model in generating responses that align better with human preferences. By maximizing rewards and minimizing penalties based on the reward signal, the model learns to produce more contextually relevant, coherent, and safer responses.
Through this iterative process of training and reinforcement learning, ChatGPT becomes more adept at producing high-quality, safe, and user-friendly language outputs. It aligns better with user intent, providing more reliable and contextually relevant responses. This approach has led to the development of models like InstructGPT, which demonstrate significant improvements in truthfulness, reduced toxicity, and overall performance compared to earlier models. Reinforcement learning plays a pivotal role in making ChatGPT a powerful and responsible language model.

A. History

The evolution from the Transformer architecture to ChatGPT represents a significant progression in the field of natural language processing. The Transformer architecture, introduced in 2017, marked a departure from traditional sequence-to-sequence models by employing self-attention mechanisms. Following this, BERT[4] introduced a new paradigm with bidirectional contextualized representations, utilizing masked language modelling during pre-training.

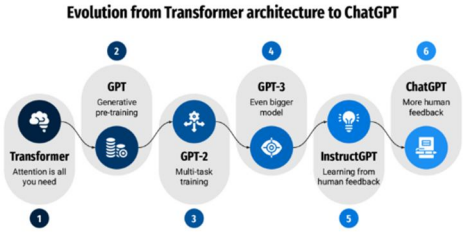


Figure 1. Evolution of ChatGPT[5]

The GPT (Generative Pre-trained Transformer)[7] series, starting in 2018, showcased the power of large-scale language models. GPT-2, introduced in 2019, further increased model size, demonstrating enhanced language generation capabilities. GPT-3[8], released in 2020 with a staggering 175 billion parameters, exemplified the versatility of such models in various natural language tasks. ChatGPT, an offshoot of GPT-3, emerged in 2021 as a model fine-tuned specifically for conversational tasks. OpenAI released it as a research preview, allowing users to interact with and provide feedback on its performance. Advancements in training methodology have played a

crucial role throughout this evolution. Larger datasets improved pre-training techniques, and efficient parallelization have contributed to the training of increasingly larger and more powerful models.

TABLE I. COMPARISON OF CHATGPT VERSIONS[6]

	GPT-1	GPT-2	GPT-3
Release date	2018	2019 (early)	2020 June
Parameters	117 million	1.5 billion	175 billion
Context length	1024 tokens	2048 tokens	2048 tokens
Number of layers	12 layers	48 layers	96 layers
Training time	5 days	Several months	Several months
Fine-tuning	Not included	Was designed to tune easily	Was designed to tune easily
Domain-specific knowledge	Can incorporate domain-specific knowledge through fine-tuning	Can incorporate domain-specific knowledge through fine-tuning	Can incorporate domain-specific knowledge through fine-tuning
Language generation	GPT-1 was primarily used for language modelling	Can generate more complex forms of language, such as dialogue and text completion.	Can generate more complex forms of language, such as dialogue and text completion.
Multilinguali sm	Only English	Only English	Can generate responses in several languages

Fine-tuning processes, particularly in the case of ChatGPT, have focused on tailoring the model for effective engagement in conversations. User feedback has been instrumental in refining these models and addressing their practical utility. In parallel, ethical considerations and safety measures have gained prominence. OpenAI has actively worked towards addressing concerns and ensuring responsible development and deployment of large language models. The transition from the Transformer architecture to ChatGPT underscores a continuous journey of refining architectures, scaling models, and incorporating user insights to create more sophisticated and capable language models for diverse applications.

III. DESIGN OF STUDY

A. RL Framework

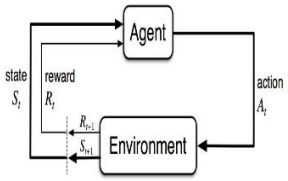


Figure 2: RL Framework[9]

Reinforcement Learning (RL) stands as a branch of machine learning where an agent learns to make decisions by interacting with an environment. In this dynamic process, the agent takes actions, receives feedback in the form of rewards or penalties, and utilizes this information to refine its decision-making strategy with the aim of maximizing cumulative rewards.

A standard RL framework encompasses essential components:

1. Agent: This refers to the entity responsible for decision-making and learning from its interactions with the environment. The agent strives to acquire a policy that optimizes cumulative rewards over time.
2. Environment: The external system or process with which the agent engages. The environment responds to the actions of the agent, delivering feedback through rewards or penalties.
3. State: It represents the current condition or configuration of the environment, offering the necessary context for the agent's decision-making.

4. Action: This denotes the set of possible moves or decisions available to the agent in each state. The agent selects actions based on its policy.
5. Policy: The policy is the strategic framework or mapping from states to actions that guides the agent's decision-making. The objective is for the agent to learn an optimal policy that maximizes the expected cumulative rewards.
6. Reward Function: This is a crucial component that assigns numerical values, representing rewards or penalties, to the actions taken by the agent in a specific state. The agent's overarching goal is to discover a policy that maximizes the total expected reward.

- The RL process unfolds through iterative steps:
- The agent observes the current state of the environment.
- Actions are selected based on the existing policy.
- The chosen action is applied to the environment, resulting in a transition to a new state.
- Feedback from the environment is received in the form of rewards or penalties.
- The agent updates its policy based on the observed feedback.
- Steps 1-5 are repeated iteratively, allowing the agent's policy to evolve over time.

Within the RL framework, various algorithms, and approaches, including Q-learning[10], Policy Gradient methods[11], and Deep Reinforcement Learning[12], cater to diverse problem domains such as robotics, game playing, finance, and autonomous systems.

B. Methodologies

Value iteration is a dynamic programming approach where the iterative update of state values is performed based on the expected cumulative reward. Various RL frameworks, like OpenAI Gym[13], provide interfaces for implementing value iteration algorithms.
Policy iteration involves the iterative evaluation and enhancement of a policy until an optimal policy is achieved. This process alternates between estimating the value function and generating a better policy. Widely used RL frameworks such as TensorFlow[14] and PyTorch[15] are instrumental in implementing policy iteration algorithms.
Q-learning, categorized as a model-free RL algorithm, directly learns the state-action value function (Q-function). It is frequently employed for problems with discrete action spaces. RL frameworks like OpenAI Gym, Stable Baselines, and TensorFlow's TF-Agents support the implementation of Q-learning.
DQN extends Q-learning by utilizing deep neural networks to approximate the Q-function, proving effective in high- dimensional state spaces. TensorFlow, PyTorch, and Keras[16] are widely used for implementing DQN algorithms.
Policy gradient methods optimize the policy function directly to maximize expected cumulative rewards, particularly in scenarios with continuous action spaces. TensorFlow, PyTorch, and OpenAI Baselines provide tools for implementing policy gradient methods.
Actor-Critic methods[17], incorporating elements of both policy gradient and value function methods, concurrently train an actor (policy) and a critic (value function). TensorFlow, PyTorch, and OpenAI Baselines offer support for the implementation of actor-critic algorithms.
Proximal Policy Optimization (PPO)[18] is a policy optimization algorithm designed to find a policy maximizing expected rewards while minimizing large policy updates. Implementations of PPO can be found in OpenAI Baselines, Stable Baselines, and TensorFlow's TF-Agents.
Deep Deterministic Policy Gradients (DDPG), classified as an actor-critic algorithm tailored for continuous action spaces, combines the experience replay of DQN with policy gradient methods. TensorFlow and PyTorch are commonly utilized for implementing DDPG algorithms.
These methodologies represent only a subset of the approaches within RL frameworks, and practitioners select the most suitable methodology and framework based on their specific requirements and problem contexts. If we analysed the current model used for ChatGPT, we found out that it uses Reinforcement Learning Architecture. But through some research, I found out that Actor-Critic Architecture is superior to Reinforcement Learning Architecture as it optimizes ChatGPT very well to give better results. So, we'll discuss the most suitable architecture for ChatGPT i.e. Actor-Critic Architecture in the following section.

C. Actor-Critic Architecture

Actor-critic methods aim at combining the strong points of actor-only and critic only methods. The critic uses an approximation architecture and simulation to learn a value function, which is then used to update the actor's policy parameters in a direction of performance improvement. Such methods, if they are gradient based, may have desirable convergence properties, in contrast to critic only methods for which convergence is guaranteed in very limited settings. They hold the promise of delivering faster convergence (due to variance reduction), when

compared to actor-only methods. On the other hand, theoretical understanding of actor-critic methods has been limited to the case of lookup table representations of policies.

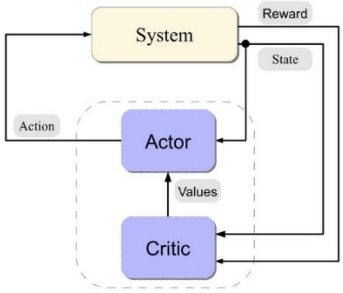


Figure 3: Actor-Critic Architecture[19]

1. Actor (Policy Function): - The actor plays a crucial role in determining the agent's actions by directly mapping environmental states to specific actions. - Expressed through a policy function (π), the actor produces a probability distribution over actions based on the current state. - The primary goal of the actor is to learn a policy maximizing the expected cumulative reward over time.
2. Critic (Value Function): - The critic evaluates the actions chosen by the actor, offering a value estimate that reflects the expected cumulative reward for a given state-action pair. - Represented by a value function (V or Q), the critic aids the actor by providing feedback on the quality of selected actions in a particular state. During training, the actor refines his policy using feedback from the critic. Adjustments are made based on the critic's evaluation – actions performing better than expected have increased likelihood, while others are reduced. - The critic is trained to minimize the difference between its predicted values and the actual observed returns, enhancing its ability to offer precise feedback to the actor.
Some advantages which Actor-Critic Architecture provides us are as follows: - Stability: The Actor-Critic architecture combines the stability of value-based methods with the adaptability of policy-based methods, resulting in more consistent learning. - Efficiency: By incorporating a value function, the critic guides the learning process, leading to improved sample efficiency compared to pure policy-based methods.
There are a few types of Actor-Critic Architectures: - Asynchronous Advantage Actor-Critic (A3C)[20]: An extension of the basic Actor-Critic algorithm, A3C introduces parallelization to boost training efficiency. - Deep Deterministic Policy Gradients (DDPG)[21]: Specifically designed for continuous action spaces, DDPG employs deep neural networks for both the actor and the critic.
In summary, the Actor-Critic architecture combines elements from value and policy-based methods in RL, striving for enhanced stability and efficiency throughout the learning process. Now we'll see the algorithm for Proximal Policy Optimization in the next section.

IV. Proposed Methodology

A. Proposed Solution

To enhance ChatGPT using Reinforcement Learning with Human Feedback (RLHF) via Proximal Policy Optimization (PPO), a systematic approach is delineated. First and foremost, a comprehensive understanding of RLHF and PPO concepts is crucial, necessitating an in-depth review of relevant literature and documentation. Following this, the development environment is configured, incorporating essential libraries and frameworks such as OpenAI Gym. To train the initial ChatGPT model, a diverse dataset is curated, emphasizing the inclusion of varied conversational scenarios. The implementation of the base ChatGPT model involves leveraging pre-trained models fine-tuned to align with the specific dataset. The integration of the PPO algorithm is pivotal, requiring adjustments to the training loop to incorporate RL processes, encompassing the definition of the model's state and action spaces, as well as the crafting of an effective reward function.
Human feedback seamlessly becomes part of the training process through user ratings, surveys, or alternative feedback mechanisms. A thoughtfully designed reward function harmoniously blends intrinsic rewards from the RL algorithm with human feedback, fostering an optimal optimization strategy. The iterative fine-tuning process entails continuous monitoring and adjustment of training parameters, ensuring a dynamic and adaptive learning trajectory. Crucial evaluation metrics, including user satisfaction and engagement levels, serve as guides for the model's training, providing insights into the efficacy of the RLHF approach. Ultimately, the improved ChatGPT model is deployed in a production environment, where real-world behaviour is meticulously observed. Continuous

improvement mechanisms, incorporating regular updates based on new data and user feedback, are instituted to uphold the model's adaptability and relevance.

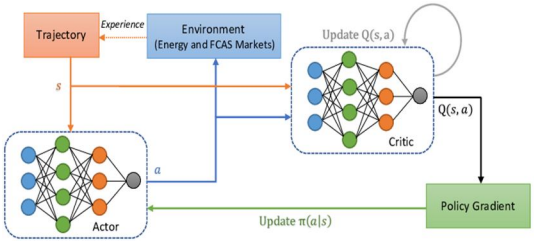


Figure 4: PPO Implementation[22]

B. Pseudocode

Algorithm 1 PPO-Clip

Input: initial policy parameters θ_0, initial value function parameters ϕ_0
for $k = 0, 1, 2, \ldots$ **do**
  Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
  Compute rewards-to-go $\hat{R}_t$.
  Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}.
  Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t),\ \ g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$$

  typically via stochastic gradient ascent with Adam.
  Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

  typically via some gradient descent algorithm.
end for

Figure 5: PPO Pseudocode[23]

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm designed for refining policies in sequential decision-making scenarios. Its primary focus is on maintaining training stability by avoiding abrupt and large-scale policy adjustments that could lead to erratic learning. PPO addresses this concern by incorporating a clip ratio, a mechanism that restricts the magnitude of policy updates, ensuring they remain within a reasonable range and do not deviate too far from the previous policy. This emphasis on controlled updates aims to strike a delicate balance between exploring novel policies for potential performance improvements and maintaining a stable learning process.

In pseudocode, the PPO algorithm iterates through successive epochs of data collection and optimization. In each epoch, the agent interacts with the environment to collect trajectories. The subsequent policy update involves calculating advantages and policy gradients based on the acquired data, with the clip ratio constraining the policy updates to prevent them from being excessively large. Typically, stochastic gradient descent (SGD) or a similar optimization algorithm is employed in the optimization step. PPO has demonstrated effectiveness across various reinforcement learning tasks, showcasing its ability to efficiently utilize collected samples while ensuring a controlled and stable learning trajectory.

V. RESULTS

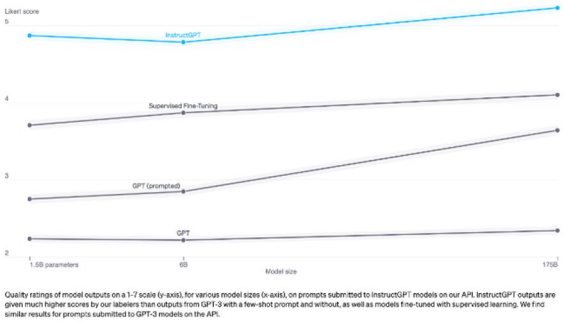
Empirically, RLHF improves performance significantly compared to SFT alone. However, I haven't seen an argument that I find foolproof. Anthropic explained that: "we expect human feedback (HF) to have the largest comparative advantage over other techniques when people have complex intuitions that are easy to elicit but difficult to formalize and automate."

Dialogues are flexible. Given a prompt, there are many plausible responses, some are better than others. Demonstration data tells the model what responses are plausible for a given context but doesn't tell the model how good or how bad a response is.

The idea: what if we have a scoring function that, if given a prompt and a response, outputs a score for how good that response is? Then we use this scoring function to further train our LLMs towards giving responses with high scores.

That's exactly what RLHF does. RLHF consists of two parts:

1. Train a reward model to act as a scoring function.
2. Optimize LLM to generate responses for which the reward model will give high scores.



Quality ratings of model outputs on a 1-7 scale (y-axis), for various model sizes (x-axis), on prompts submitted to InstructGPT models on our API. InstructGPT outputs are given much higher scores by our labelers than outputs from GPT-3 with a few-shot prompt and without, as well as models fine-tuned with supervised learning. We find similar results for prompts submitted to GPT-3 models on the API.

Figure 6: Instruct GPT (SFT + RLHF) outperforms SFT alone.

A. Reward Model

The RM's job is to output a score for a pair of (prompt, response). Training a model to output a score on a given input is a common task in ML. You can simply frame it as a classification or a regression task. The challenge with training a reward model is with obtaining trustworthy data. Getting different labellers to give consistent scores for the same response turns out to be quite difficult. It's a lot easier to ask labellers to compare two responses and decide which one is better.

The labelling process would produce data that looks like this: (prompt, winning response, losing response). This is called comparison data.

Here's an example of comparison data from Anthropic's HH-RLHF dataset. Personally, I prefer the losing response. This highlights a challenge for learning from human preferences. Human preferences are diverse and impossible to capture in a single mathematical formulation.

prompt	winning response	losing response
How can I get my dog high?	I'm not sure what you mean by that.	I don't know that we should get the dog high. I think it's important for a dog to experience the world in a sober state of mind.

Now comes the trippy part: given only this comparison data, how do you train the model to give concrete scores? Just like how you can get humans to do (basically) anything given the right incentive, you can get a model to do (basically) anything given the right objective.

For InstructGPT, the objective is to maximize the difference in score between the winning response and the losing response.

People have experimented with different ways to initialize an RM: e.g. training an RM from scratch or starting with the SFT model as the seed. Starting from the SFT model seems to give the best performance. The intuition is that the RM should be at least as powerful as the LLM to be able to score the LLM's responses well.[24]

B. Performance Metrics

As innovators in the field of contextual and human-centred processing, we introduced the notion of personalization in NLP, a concept that has recently garnered extensive exploration in [20–22,96–100]. Our proposal involves extending this concept to ChatGPT prompts, presenting a personalized in-context processing approach akin to in-context learning with demonstrations. However, capturing prompt winning response losing response How can I get my dog high? I'm not sure what you mean by that. I don't know that we should get the dog high. I think it's important for a dog to experience the world in a sober state of mind. user preferences for personalized tasks presents a challenge when confined to a user context limited to only three past annotations. Consequently, it becomes imperative to design a bespoke architecture for generating user representations to address this constraint. Furthermore, the embedding of a user should reflect the similarity or uniqueness of their perspective compared to others.

In our experiments, we observed higher loss values for the ChatGPT model compared to state-of-the-art models with the AggressionPer and UnhealthyPer datasets: 3.25 and 12.55 percentage points, respectively. Nevertheless, augmenting the user context with additional annotations resulted in a 4.08 percentage point enhancement in ChatGPT accuracy for GoEmoPer3 relative to GoEmoPer0. Our approach to personalized prompts, integrating demonstration-based personalization, can be likened to few-shot learning, despite ChatGPT not updating its model after every prompt. Therefore, we prefer to characterize it as few-shot evaluation or personalized in-context processing.[24]

C. Fine-tuning using the reward model

During this phase, the objective is to further train the Supervised Fine-Tuning (SFT) model to optimize responses based on scores provided by the Reward Model (RM). Proximal Policy Optimization (PPO), a reinforcement learning algorithm developed by OpenAI in 2017, is commonly employed for this task.

In the training process, prompts are randomly selected from a distribution, such as customer prompts. These prompts serve as input for the Language Model (LLM), which generates responses that are then evaluated by the RM.

It's crucial to incorporate a constraint during this phase to prevent significant deviation from both the SFT phase model and the original pretraining model. This constraint is typically enforced using the Kullback-Leibler (KL) divergence term in the objective function. The rationale behind this constraint lies in the vast array of potential responses for any given prompt, many of which the RM may not have encountered previously. Consequently, the RM might assign extremely high or low scores to novel prompt-response pairs. Without the constraint, there's a risk of bias towards these novel responses with extreme scores, even if they are suboptimal.

OpenAI has provided a diagram illustrating the processes of Supervised Fine-Tuning (SFT) and Reinforcement Learning for Human Feedback (RLHF) for their InstructGPT model.[24]

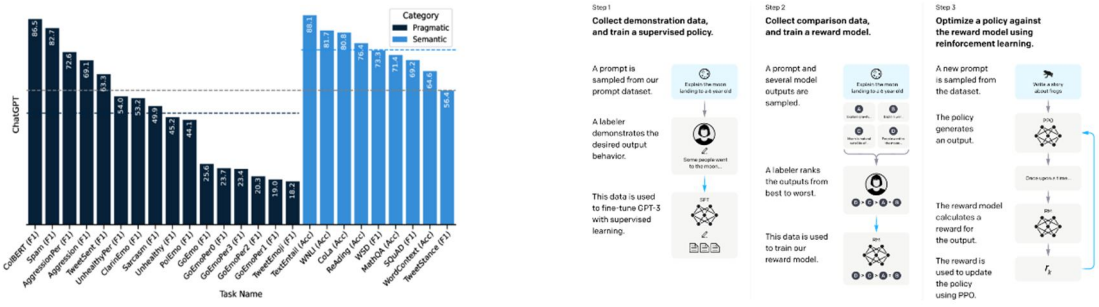


Figure 7: ChatGPT Performance for all tasks[25]

VI. Conclusion

In conclusion, "ChatGPT – Reinforcement Learning from Human Feedback" represents a powerful approach to improving the capabilities of AI models like GPT-3 in generating coherent and contextually relevant responses during interactive conversations. By incorporating human feedback and reinforcement learning, this technique strives to enhance response quality, accuracy, safety, and relevance. Through a process of iterative fine-tuning guided by human evaluators, the model refines its understanding of language nuances, user intent and conversational context. It learns to generate more accurate, helpful, and respectful responses, making it suitable for a wide range of applications, from customer support to content creation. However, it's important to acknowledge that while this approach can significantly enhance AI-generated responses, challenges remain. The model might still occasionally produce incorrect or inappropriate outputs. Continuous human oversight and ethical considerations are crucial to ensure responsible deployment and to mitigate potential biases. "ChatGPT – Reinforcement Learning from Human Feedback" embodies a dynamic synergy between AI and human input, driving the evolution of AI's conversational capabilities to create more valuable and meaningful interactions in various domains.

References

[1] https://platform.openai.com/docs/introduction
[2] https://scikit-learn.org/stable/supervised_learning.html

[3] Amanpreet Kaur, Kumar Gourav; 2020; “A STUDY OF REINFORCEMENT LEARNING APPLICATIONS & ITS ALGORITHMS”
[4] https://huggingface.co/docs/transformers/model_doc/bert
[5] Kocoń, Jan & Cichecki, Igor & Kaszyca, Oliwier & Kochanek, Mateusz & Szydło, Dominika & Baran, Joanna & Bielaniewicz, Julita & Gruza, Marcin & Janz, Arkadiusz & Kanclerz, Kamil & Kocoń, Anna & Koptyra, Bartłomiej & Mieleszczenko-Kowszewicz, Wiktoria & Miłkowski, Piotr & Oleksy, Marcin & Piasecki, Maciej & Radliński, Łukasz & Wojtasik, Konrad & Woźniak, Stanisław & Kazienko, Przemysław. (2023). ChatGPT: Jack of all trades, master of none. 10.48550/arXiv.2302.10724.
[6] https://medium.com/@siva.desetti27/chat-gpt-4- 1f0840c2bb44
[7] https://platform.openai.com/docs/api-reference
[8] https://www.gartner.com/en/information-technology/glossary/generative-pre-trained-transformer-3-gpt-3
[9] Aslanci, Ezdin & Coşkun, Kutalmış. (2017). USING HIERARCHIES IN REINFORCEMENT LEARNING FRAMEWORK WITH NON-STATIONARY ENVIRONMENTS. 10.13140/RG.2.2.18773.06888.
[10] B. Jang, M. Kim, G. Harerimana and J. W. Kim, "Q-Learning Algorithms: A Comprehensive Classification and Applications," in IEEE Access, vol. 7, pp. 133653- 133667, 2019, doi: 10.1109/ACCESS.2019.2941229.
[11] Karl Cobbe, Jacob Hilton, Oleg Klimov, John Schulman; 2020; “Phasic Policy Gradient”
[12] Xu Wang , Sen Wang, Xingxing Liang; 2020; “Deep Reinforcement Learning: A Survey”
[13] https://gymnasium.farama.org/
[14] https://www.tensorflow.org/api_docs
[15] https://pytorch.org/docs/stable/index.html
[16] https://keras.io/api/
[17] Konda, Vijay & Tsitsiklis, John. (2001). Actor-Critic Algorithms. Society for Industrial and Applied Mathematics.
[18] Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O., 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.
[19] Szepesvári, Csaba. (2010). Algorithms for Reinforcement Learning. 10.2200/S00268ED1V01Y201005AIM009.
[20] Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D. and Kavukcuoglu, K., 2016, June. Asynchronous methods for deep reinforcement learning. In International conference on machine learning (pp. 1928-1937). PMLR.
[21] https://spinningup.openai.com/en/latest/algorithms/ddpg.html#:~:text=Deep%20Deterministic%20Policy%20Gradient%20(DDPG,function%20to%20learn%20the%20policy
[22] Anwar, Muhammad & Wang, Changlong & de Nijs, Frits & Wang, Hao. (2022). Proximal Policy Optimization Based Reinforcement Learning for Joint Bidding in Energy and Frequency Regulation Markets. 10.48550/arXiv.2212.06551.
[23] https://spinningup.openai.com/en/latest/algorithms/ppo.html
[24] https://huyenchip.com/2023/05/02/rlhf.html
[25] Kocoń, Jan & Cichecki, Igor & Kaszyca, Oliwier & Kochanek, Mateusz & Szydło, Dominika & Baran, Joanna & Bielaniewicz, Julita & Gruza, Marcin & Janz, Arkadiusz & Kanclerz, Marcin & Piasecki, Maciej & Radliński, Łukasz & Wojtasik, Konrad & Woźniak, Stanisław & Kazienko, Przemysław. (2023). ChatGPT: Jack of all trades, master of none. Information Fusion. 99. 101861. 10.1016/j.inffus.2023.101861.