

# LEARNING TO GENERATE BETTER THAN YOUR LLM

**Anonymous authors**

Paper under double-blind review

## ABSTRACT

Reinforcement learning (RL) has emerged as a powerful paradigm for fine-tuning Large Language Models (LLMs) for text generation. In particular, recent LLMs such as ChatGPT and GPT-4 can engage in fluent conversations with users after finetuning with RL. Inspired by *learning-to-search* algorithms and capitalizing on key properties of text generation, we seek to investigate RL algorithms beyond general purpose algorithms like Proximal Policy Optimization (PPO). In particular, we extend RL algorithms to allow them to interact with a dynamic black-box guide LLM and propose RL *with guided feedback* (RLGF), a suite of RL algorithms for LLM fine-tuning. We experiment on the IMDB positive sentiment, CommonGen, and TL;DR summarization tasks. We show that our RL algorithms achieve higher performance than supervised learning (SL) and RL baselines, demonstrating the benefit of interaction with the guide LLM. On both CommonGen and TL;DR, we not only outperform our SL baselines but also improve upon PPO across a variety of metrics beyond the one we optimized for.

## 1 INTRODUCTION

Large Language Models (LLMs) have become very capable in various real-world applications ranging from being able to answer open-ended questions on numerous topics (Zhang et al., 2022), write articles from short descriptions (Goyal et al., 2022), generate code (Github, 2023), follow robot commands (Huang et al., 2022), solve puzzles (Bubeck et al., 2023), and even showcased as assistive models for education (Khan Academy, 2023) and healthcare (Lee et al., 2023b).

However, using supervised learning (SL) to train LLMs presents a challenging metric mismatch (Wiseman & Rush, 2016) between the training and testing regimes. The metric mismatch arises from the training metric being the log-loss while the testing metrics are task-specific such as BLEU or user satisfaction rating. This discrepancy is magnified when fine-tuning LLMs on downstream tasks where the main goal is not just producing fluent text but also being proficient at solving the specific task.

Reinforcement Learning (RL) by definition address this metric mismatch by directly optimizing the metrics through reward feedback. Recently, OpenAI fine-tuned LLMs with RL from human feedback (RLHF) to better align LLMs to human intentions, leading to the great success of ChatGPT (OpenAI, 2023). Recently, GRUE benchmark (Ramamurthy et al., 2022) systematically studied RL versus SL when finetuning LLMs on downstream tasks with predefined rewards. GRUE’s preliminary results demonstrate the benefit of RL when fine-tuning LLMs, leading to the release of popular codebases such as RL4LMs (Ramamurthy et al., 2022), TRLx (CarperAI, 2023) and AlpacaFarm (Dubois et al., 2023), that enables RL for language models. However, ChatGPT, RL4LMs, TRLX, and AlpacaFarm all use vanilla policy gradient methods known to be sample inefficient and sensitive to local minima due to the combinatorially large search space of natural language generation (Ramamurthy et al., 2022).

In this work, we focus on more efficient ways of fine-tuning LLMs on downstream tasks with predefined rewards. Our approach is motivated by prior work on Imitation Learning (IL) for structured prediction, which often leverages an existing guide policy (not necessarily an optimal policy) to reduce the search space for more efficient and optimal learning. Our key observation is that since modern LLMs exhibit impressive general language capabilities, they can serve as guide policies to improve the RL procedure. Our framework, which we call, *RL with guided feedback* (RLGF), integrates a guide policy into a policy gradient framework. The guide policy can provide reasonable but sub-optimal predictions for downstream tasks, which our framework can then leverage to learn

a near-optimal strategy. We introduce novel algorithms for fine-tuning LLMs using our RLGf framework while capturing various existing IL for structured prediction and RL algorithms.

We evaluate on three tasks. The first is IMDB where the goal is to generate a positive and fluent review given an initial context. The second is CommonGen where the goal is to write a fluent text that uses a given set of words. Finally, we test on the TL;DR summarization task where the objective is to learn to generate summaries using human preference data. For all tasks, we find evidence of metric mismatch from SL-based fine-tuning approaches and show that RL-based methods which utilize reward signals outperforms on the task metric. We then demonstrate RLGf outperforming PPO on reward, fluency, as well as automated lexical metrics such as Rouge. Finally, we investigate how various baselines and RLGf algorithms balance the inherent trade-off between reward optimization and the KL constraint in the RLHF objective. We provide both theoretical justification and empirical evidence to show the benefit of using feedback in RL for fine-tuning LLMs on downstream tasks.

## 2 RELATED WORK

Here we present the most relevant works at the intersection of IL, RL, and natural language generation. Please see Appendix A for a more thorough treatment of the literature.

**IL for Structured Prediction:** Algorithms such as Schedule Sampling (SS) (Bengio et al., 2015), methods using SS (Duckworth et al., 2019; Mihaylova & Martins, 2019; Goyal et al., 2017), SEARNN (Leblond et al., 2017), Bridging the Gap (Zhang et al., 2019b), Mixer (Ranzato et al., 2015) been inspired by IL for structured prediction algorithms DAGGER (Ross et al., 2011), DAD (Venkatraman et al., 2015), and SEARN (Daumé et al., 2009). Our work is inspired by AggreVaTeD (Sun et al., 2017) (Differentiable AggreVaTeD Ross & Bagnell (2014)) where the algorithm makes use of differentiable policies and multi-step feedback rather than immediate one-step predictions to imitate. Similarly, we present a differentiable version of LOLS (Chang et al., 2015) as well as an improvement, D<sup>2</sup>LOLS.

**LLM Fine-tuning from Human Preferences:** Recent advancements in fine-tuning of Large Language Models (LLMs) have shown incredible success in tasks through learning from human preferences. Being simpler to accumulate human preferences, Reinforcement Learning from Human Feedback (RLHF) (Stiennon et al., 2020) introduced a paradigm to utilize RL to improve downstream performance on translation (Kreutzer et al., 2018b), summarization (Stiennon et al., 2020), storytelling (Ziegler et al., 2019), and instruction following (OpenAI, 2023). Although effective, following works have shown RLHF to be challenging due to reward hacking, difficulties in scaling, and training instability (Zhao et al., 2023; Rafailov et al., 2023; Liu et al., 2023). To circumvent these difficulties, recent works have proposed methods to optimize for human preferences without RL (Zhao et al., 2023; Yuan et al., 2023; Rafailov et al., 2023; Liu et al., 2023). DPO, SLiC, RRHF, and RSO are methods that optimize for compatibility with a preference dataset under a preference reward model such as the Bradley Terry model (Bradley & Terry, 1952). In contrast, our work takes a different approach to improving RLHF by investigating improvements to PPO (Schulman et al., 2017), the base RL algorithm used.

**LLM Distillation:** With an ever growing arsenal of powerful, black-box LLMs, recent work has aimed to distill specific capabilities into a smaller model. Knowledge distillation (Buciluă et al., 2006; Hinton et al., 2015) in autoregressive models investigated matching sequence level log probabilities (Kim & Rush, 2016), model hidden states (Jiao et al., 2019), or attention scores (Wang et al., 2020). Recently, more sophisticated methods, inspired from the IL literature, are being proposed to better imitate the expert LLM’s performance (Lin et al., 2020a; Agarwal et al., 2023; Mukherjee et al., 2023), with ORCA (Mukherjee et al., 2023) reaching parity performance with ChatGPT (OpenAI, 2023) by distilling the reasoning traces from GPT4 (OpenAI, 2023). Distinct from this line of work, RLGf does not aim to replicate the guidance policy. Rather, our objective is to leverage generation traces derived from a guide policy to condense the search space for RL algorithms. More importantly, our goal goes beyond imitation of the guidance policy and focuses on algorithms that better optimize a reward with guidance policy feedback.

## 3 PRELIMINARIES

Text generation with LLMs can be viewed as a structured prediction problem, consisting of an input space  $\mathcal{X}$ , an output space  $\mathcal{Y}$  and non-negative loss function  $\ell(x, \hat{y}, y^*) \mapsto \mathbb{R}^{\geq 0}$  such that the loss

function  $\ell$  represents how close  $\hat{y}$  is to the ground truth  $y^*$  given the input  $x$ . We are provided with a training set of  $N$  labeled input-output pairs  $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$  drawn from some unknown distribution over  $\mathcal{X} \times \mathcal{Y}$ . The goal is to learn a mapping  $f : \mathcal{X} \mapsto \mathcal{Y}$  that minimizes the loss function  $\ell$  with respect to  $\mathcal{D}$ . We adopt the approach of solving the text generation structured prediction problems using sequential decision-making as formalized in learning-to-search (L2S) (Daumé et al., 2009; Collins & Roark, 2004; Ratnaparkhi, 1996).

We view our L2S problem as a token-level finite-horizon MDP  $\langle \mathcal{S}, \mathcal{A}, P, R, H, \mu \rangle$  using a finite vocabulary  $\mathcal{V}$ . We are given a labeled dataset  $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$  of  $N$  samples, where  $x^i$  is a prompt text and  $y^i$  is the target text generation. We define  $\mu \in \Delta(\mathcal{D})$  as the initial distribution over prompts in the dataset, and the action space  $\mathcal{A}$  as the set of tokens in our vocabulary  $\mathcal{V}$ . The state space  $\mathcal{S} = \cup_{h=1, \dots, H} \mathcal{V}^h$  is the set of all possible token sequences and a state  $s_h \in \mathcal{S}$  is the prompt  $x$  and previously generated tokens  $(a_0, a_1, \dots, a_{h-1})$ , i.e.,  $s_h = (x, a_0, a_1, \dots, a_{h-1})$ . The transition function  $P : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$  is a deterministic known transition function that appends the next action  $a_h$  to the state  $s_{h+1}$ . The time horizon  $H \in \mathbb{Z}_+$  is the maximum generation length. Finally,  $R : \mathcal{S} \rightarrow \mathbb{R}$  is the reward function such as the task evaluation metric.

Let  $d_h^\pi$  represent the state distribution of visiting a state at time  $h$ . Let  $d^\pi = \frac{1}{H} \sum_{h=0}^H d_h^\pi$  be the average visitation if we follow  $\pi$  for  $H$  steps in a trajectory. With an LLM policy  $\pi$ , we define the value function and  $Q$ -function as  $V_h^\pi(s) = \mathbb{E}_\pi[\sum_{h'=h}^H R(s_{h'}) | s_h = s]$  and  $Q_h^\pi(s, a) = R(s) + \mathbb{E}_{s' \sim P(\cdot | s, a)}[V_{h+1}^\pi(s')]$  respectively. Finally, we define the advantage function for an LLM policy  $\pi$  as  $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$ .

**Guide policy  $\pi^g$**  In our setting, we additionally assume access to an LLM guide policy  $\pi^g$  that can assist our policy  $\pi$ . The guide policy can be used to alter the initial state distribution  $\mu$  and to compute the advantage function  $A^{\pi^g}(s, a)$ . In this work,  $\pi^g$  is a supervised fine-tuned (SFT) model on the downstream task and generate feedback from  $\pi^g$  with a more effective decoding strategy like nucleus sampling (Holtzman et al., 2019). Note, RLGf treats  $\pi^g$  as a query-able, black-box model that we cannot update. This allows for  $\pi^g$  to be any black-box model such as GPT4 or a human-expert.

#### 4 REINFORCEMENT LEARNING FROM GUIDED FEEDBACK

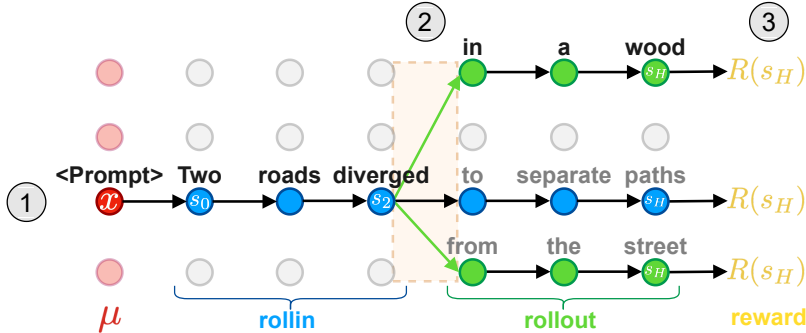


Figure 1: RLGf’s main mechanism of incorporating guidance through interactions between two LLMs: **rollin** and **rollout** policies. (1) the **rollin** policy generates a trajectory. (2) the **rollout** policy restarts to a sampled point in the generation (i.e.  $s_2$ ) and completes the generation. (3) the **rollout** policy receives a score (i.e. reward) for the generation.

Unlike other tasks studied in RL, structured prediction problems such as text generation, have two key properties: a deterministic transition function and a policy’s ability to restart to any state. Because our transition function is the set of previously generated tokens, we can easily alter the words in the generation (add, remove or swap), and restart our policy  $\pi_\theta$  to any point of the generation.

Restarts allow us to execute **rollin** and **rollout** policies as seen in Figure 1. The **rollin** policy is used to generate sequences that the **rollout** policy evaluates. Specifically, we sample a prompt  $x$  and target sentence  $y$  from our initial distribution  $\mu$ . We then generate an entire trajectory using our **rollin** policy starting from the sampled prompt. We combine the state-action pairs from the collected **rollin**

trajectory with the initial state distribution – creating a modified initial state for the **rollout** policy. The **rollout** policy samples a state along the **rollin** generation, restarts to this state and performs a one-step deviation action. The **rollout** policy then completes the generation and collects a reward. The **rollin** and **rollout** policies can be our LLM policy  $\pi_\theta$ , guide policy  $\pi^g$  or a mixture that interpolates between the two. Depending on the choice of **rollin** and **rollout** policies, we invoke different algorithms.

**PPO: Rollin  $\pi_\theta$  and Rollout  $\pi_\theta$**  Under this schematic, notice how when both the rollin and rollout policies are our current LLM policy  $\pi_\theta$  that is being fine-tuned, the resulting RL algorithm is PPO. That is, we would be collecting generations from a single LLM. This configuration does not take advantage of the ability to modify the initial state distribution nor the availability of a guide policy  $\pi^g$ .

---

**Algorithm 1** PPO<sup>++</sup>


---

- 1: **Input:**  $\pi_\theta$ , guide  $\pi^g$ , iterations  $T$ , mixing parameter  $\beta \in [0, 1]$ , dataset  $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$
  - 2: **for**  $t \in [T]$  **do**
  - 3:   Rollin with  $(s, a) \sim \beta d^{\pi^g} + (1 - \beta)d^{\pi_\theta^t}$  starting from  $x \sim \mathcal{D}$
  - 4:   Rollout with  $\pi_\theta^t$  to collect trajectories
  - 5:   Update  $V_\phi^{\pi_\theta^t}$  with trajectories and compute advantage estimates  $A^{\pi_\theta^t}$
  - 6:   Update  $\pi_\theta$  using PPO loss with  $A^{\pi_\theta^t}$
  - 7: **return**  $\pi_\theta$
- 

**PPO<sup>++</sup>: Rollin  $\pi^g$  and Rollout  $\pi_\theta$**  The second scheme we consider is rollin with our guide policy  $\pi^g$  and rollout with our LLM policy  $\pi_\theta$ . This strategy is motivated from a popular Approximate Policy Iteration algorithm (Bertsekas, 2011): Conservative Policy Iteration (CPI) (Kakade & Langford, 2002). CPI proposes to use a diverse initial state distribution to address the exploration issue in PG methods. Particularly, it proposes to use an initial state distribution that covers some high-quality policy distribution. The first key idea of PPO<sup>++</sup> is to take advantage of a guide policy  $\pi^g$  to provide an enlarged initial state distribution – so that the rollout policy,  $\pi_\theta$ , can visit diverse and relevant states it would otherwise not visit. The second key idea of PPO<sup>++</sup> is using a mixture policy with state distribution  $\beta d^{\pi^g} + (1 - \beta)d^{\pi_\theta}$ , for rollin (see Algorithm 1 Line 3). This ensures that with probability  $(1 - \beta)$ , PPO<sup>++</sup> is executing the default PPO update, making sure PPO<sup>++</sup> never underperforms PPO.

---

**Algorithm 2** AggreVaTeD
 

---

- 1: **Input:**  $\pi_\theta$ , guide  $\pi^g$ , iterations  $T$ , mixing parameter  $\beta \in [0, 1]$ , dataset  $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$
  - 2: **for**  $t \in [T]$  **do**
  - 3:   Rollin with  $(s, a) \sim (1 - \beta)d^{\pi_\theta} + \beta d^{\pi^g}$  starting from  $x \sim \mathcal{D}$
  - 4:   Rollout with  $\pi^g$  to collect trajectories
  - 5:   Update  $V_\phi^{\pi^g}$  with trajectories and compute advantage estimates  $A^{\pi^g}$
  - 6:   Update  $\pi_\theta$  using PPO loss with  $A^{\pi^g}$
  - 7: **return**  $\pi_\theta$
- 

**AggreVaTeD: Rollin  $\pi_\theta$  and Rollout  $\pi^g$**  The next scheme performs rollin with our LLM policy  $\pi_\theta$  and rollout with our guide policy  $\pi^g$  – the opposite of PPO<sup>++</sup>. This scheme is an interactive imitation learning algorithm, AggreVaTeD (Sun et al., 2017), a differentiable policy gradient version of AggreVaTe (Aggregate Values to Imitate (Ross & Bagnell, 2014)) as seen in Algorithm 2. AggreVaTeD is an API algorithm similar to CPI and also uses a mixture policy with state distribution  $\beta d^{\pi^g} + (1 - \beta)d^{\pi_\theta}$  for rollin. This algorithm first generates rollins with the mixture policy to collect sequences. Then AggreVaTeD generates rollouts with the guide policy and evaluates the quality of the generated rollouts. It then uses the rollouts to train a value network  $V_\phi^{\pi^g}$  that measures the reward-to-go of  $\pi^g$ , which in turn is used to construct the advantage of  $\pi^g$ :  $A^{\pi^g}$ . With this advantage  $A^{\pi^g}$ , AggreVaTeD updates the policy like PPO. Intuitively, the algorithm aims to learn the policy  $\arg \max_a A^{\pi^g}(s, a)$ . Rolling out with  $\pi^g$  ensures that the LLM policy  $\pi_\theta$  can be *at least* as good as or better than the guide policy  $\pi^g$ .

**Algorithm 3**  $D^2\text{LOLS}$ 

- 
- 1: **Input:**  $\pi_\theta$ , guide  $\pi^g$ , iterations  $T$ , dataset  $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$
  - 2: Run  $\pi_\theta^1 = \text{AggreVaTeD}(\pi_\theta, \pi^g, \alpha T, \beta_1, \mathcal{D})$
  - 3: Run  $\pi_\theta^2 = \text{PPO}^{++}(\pi_\theta^1, \pi^g, (1 - \alpha)T, \beta_2, \mathcal{D})$
  - 4: **return**  $\pi_\theta^2$
- 

**$D^2\text{LOLS}$ : combines  $\text{PPO}^{++}$  and  $\text{AggreVaTeD}$**  Given the previous approaches of interaction, we can come up with multiple ways to combine PPO,  $\text{PPO}^{++}$ , and  $\text{AggreVaTeD}$ . In Algorithm 3, we present Direct and Differentiable Locally Optimal Learning to Search ( $D^2\text{LOLS}$ ), which is a simple approach to combine the previous methods.  $D^2\text{LOLS}$  is a differentiable policy gradient version of Locally Optimal Learning to Search ( $\text{LOLS}$ ) (Chang et al., 2015) and addresses limitations of how  $\text{LOLS}$  combines PPO,  $\text{PPO}^{++}$ , and  $\text{AggreVaTeD}$ . The original formulation of  $\text{LOLS}$  requires computing cost-sensitive classification similar to  $\text{AggreVaTe}$ ; instead we take inspiration from  $\text{AggreVaTeD}$ 's differentiable approach to develop a differentiable version of  $\text{LOLS}$ . Furthermore,  $\text{LOLS}$  (Algorithm 4) has a mixing probability parameter  $\alpha$  which directly merges the advantage function between PPO and  $\text{AggreVaTeD}$ , leading to theoretical issues.  $D^2\text{LOLS}$  removes this mixing probability and replaces it with a mixing time variable  $\alpha$  that decides how many iterations to perform  $\text{AggreVaTeD}$  before switching to  $\text{PPO}^{++}$ . This simple strategy fixes  $\text{LOLS}$ 's issue arising from interweaving guidance.

## 5 THEORETICAL JUSTIFICATION

In this section, we provide theoretical justification for various rollin and rollout schemes mentioned in Section 4. Each algorithmic scheme takes advantage of a guide policy  $\pi^g$ , the ability to restart the policy to any state, and access to the reward signal. Our theoretical justification are derived from the original algorithms that each method has built upon.

**Interactive Imitation Learning:  $\text{AggreVaTeD}$**  In our interactive IL setting, we assume access to the ground truth reward and to a guide policy  $\pi^g$  that may not necessarily be an expert policy  $\pi^*$  (i.e. optimal at the task). Our  $\text{AggreVaTeD}$  (Algorithm 2) implementation is a modification of the original  $\text{AggreVaTeD}$  (Sun et al., 2017) to incorporate a PPO policy gradient loss. The overall idea is to perform policy gradient updates on the loss function  $\ell_t(\pi) := \mathbb{E}_{s \sim d^{\pi^t}} \mathbb{E}_{a \sim \pi(\cdot|s)} [A^{\pi^g}(s, a)]$ , where  $\pi^t$  is our latest learned policy. We can define the average-regret and best policy performance in our policy class over  $T$ -iterations as:

$$\epsilon_{\text{regret}} = \frac{1}{T} \left( - \sum_{t=0}^T \ell_t(\pi^t) + \max_{\pi \in \Pi} \sum_{t=0}^T \ell_t(\pi) \right) \quad \epsilon_{\text{class}} = \max_{\pi \in \Pi} \frac{1}{T} \sum_{t=0}^T \mathbb{E}_{s \sim d^{\pi^t}} [A^{\pi^g}(s, \pi(s))].$$

If the gradient update procedure achieves no-regret, i.e.,  $\epsilon_{\text{regret}} \rightarrow 0$  as  $T \rightarrow \infty$ ,  $\text{AggreVaTeD}$  achieves the following guarantee; there exists  $t \in [T]$ , such that:

$$V^{\pi^t} \geq V^{\pi^g} + H \epsilon_{\text{class}}.$$

When the guide policy is included in our policy class  $\pi^g \in \Pi$ , e.g., when our policy  $\pi_\theta$  and our guide  $\pi^g$  have the same GPT2 model architecture, then our  $\epsilon_{\text{class}}$  term is guaranteed to be non-negative. Furthermore, this term is positive when  $\pi^g$  is not globally optimal with respect to its advantage function (i.e.,  $\max_a A^{\pi^g}(s, a)$  can be positive). Thus when  $\epsilon_{\text{regret}} \rightarrow 0$  (i.e., no-regret),  $\text{AggreVaTeD}$  guarantees to learn a policy  $\pi_t$  that outperforms the guide policy by a margin. This was originally confirmed empirically in Sun et al. (2017) and is also confirmed in our experiments. With our SFT model with nucleus sampling as  $\pi^g$ ,  $\text{AggreVaTeD}$  learns a policy  $\pi^t$  outperforming  $\pi^g$ .

**Reinforcement Learning with better restart distribution:  $\text{PPO}^{++}$**  Although  $\text{AggreVaTeD}$  is capable of outperforming  $\pi^g$ , it is an imitation learning algorithm, meaning by design, its performance is limited by the performance of  $\pi^g$ . In contrast, RL has the potential to learn the near optimal policy, but popular RL approaches suffer from a lack of exploration. We propose to leverage rollin's with the guide policy to overcome RL's exploration issues.  $\text{PPO}^{++}$  Algorithm 1 implements this idea using a PPO loss. We can interpret the rollin policy distribution with the guide policy, as a restart

distribution that alters the initial distribution of our policy, i.e.,  $\mu_{\text{mix}} := (1 - \beta)\mu + \beta d^{\pi^g}$ , where recall  $\mu \in \Delta(\mathcal{D})$  is the original initial state distribution over our data.

Policy gradient theory (Kakade & Langford, 2002; Bagnell et al., 2003; Agarwal et al., 2019; 2021) ensures that as long as a near optimal policy is covered by the restart distribution, we can learn to perform as well as the near optimal policy. More formally, consider the special case where  $\beta = 1/2$ , and  $\pi^*$  is the globally optimal policy; and assume that at some iteration  $t$  one-step local improvement over  $\pi^t$  is small, i.e.,  $\mathbb{E}_{s,a \sim d_{\mu_{\text{mix}}}^{\pi^t}} [\max_a A^{\pi^t}(s, a)] \leq \epsilon$ , then with some small  $\epsilon$  we have:

$$V^{\pi^t} \geq V^{\pi^*} - O\left(H^2 \max_s \left(\frac{d^{\pi^*}(s)}{d^{\pi^g}(s)}\right) \epsilon\right)$$

We refer readers to the proof of theorem 6.2 in Kakade & Langford (2002). Note that compared to the result from AggreVaTeD, we are able to compare against the globally optimal policy  $\pi^*$  under the condition that  $\pi^g$ 's state distribution covers  $\pi^*$ 's state distribution (i.e., the guide policy has a good sense of what states  $\pi^*$  will likely visit). In our experiments, we mainly use a SFT model with nucleus sampling as our guide policy  $\pi^g$ . While we do not expect the SFT policy  $\pi^g$  is as good as the optimal  $\pi^*$ , it is reasonable to expect that  $d^{\pi^g}$  provides coverage to  $d^{\pi^*}$ . Our experiments verify that restarting based on states from  $d^{\pi^g}$  improves the performance of PPO.

**Combine Reinforcement Learning and Imitation Learning: D<sup>2</sup>LOLS** D<sup>2</sup>LOLS is the simplest approach to combine AggreVaTeD and PPO<sup>++</sup>. This algorithm runs AggreVaTeD for a fixed period of time and then PPO<sup>++</sup> for the remaining time. If our policy gradient algorithm is Trust-region policy optimization (TRPO)<sup>1</sup> (Schulman et al., 2015) or CPI (Kakade & Langford, 2002), then our algorithm has a guaranteed monotonic policy improvement. This means that upon convergence, we achieve two properties: (1) our learned policy is at least as good or better than the guide policy  $\pi^g$ , (2) our policy is locally optimal, i.e., the local one-step improvement,  $\mathbb{E}_{s,a \sim d_{\mu_{\text{mix}}}^{\pi}} [\max_a A^{\pi}(s, a)]$ , has to be small (otherwise TRPO and CPI can keep improving).

There exist several algorithms in the literature that combine RL and IL (Cheng et al., 2018; Sun et al., 2018; Chang et al., 2015; Rajeswaran et al., 2017; Nair et al., 2018). The key difference between D<sup>2</sup>LOLS and LOLS is how PPO<sup>++</sup> and AggreVaTeD is combined. LOLS uses a mixing probability  $\alpha$  to combine our  $\pi_{\theta}$  and the guide policy  $\pi^g$  advantage function  $\alpha A^{\pi_{\theta}} + (1 - \alpha) A^{\pi^g}(s, a)$ ; whereas D<sup>2</sup>LOLS uses a mixing time parameter  $\alpha$  to decide when to switch from doing AggreVaTeD to PPO<sup>++</sup> for the remainder of training. LOLS can achieve the property of outperforming better than  $\pi^g$  and also being locally optimal, but *only under* the assumption that the following gap is small:

$$\forall \pi : \left| \mathbb{E}_{s \sim d^{\pi}} \left[ \max_a A^{\pi^g}(s, a) + \max_a A^{\pi}(s, a) \right] - \mathbb{E}_{s \sim d^{\pi}} \max_a \left[ A^{\pi^g}(s, a) + A^{\pi}(s, a) \right] \right| \leq \epsilon,$$

with some small  $\epsilon$ . However, such a gap can exist in practice and does not vanish even with enough training data. Intuitively this gap is non-trivial when the one-step improvement over  $\pi$  contradicts with the one-step improvement over  $\pi^g$ . The simplest approach D<sup>2</sup>LOLS works the best, and achieves the guarantee that LOLS aimed for without the additional assumption of the above gap being small.

## 6 EXPERIMENTS

We perform all of our experiments using a modified PPO objective  $J_{ppo}$  (Ouyang et al., 2022; Wu et al., 2016). This objective combines the original PPO objective with a maximum-likelihood estimation (MLE) objective of the ground-truth dataset's  $\mathcal{D}$  references:

$$J_{ppo}(\pi) = \mathbb{E}_{(s,a) \sim \pi_{\theta}} \left[ R(s) - \lambda \text{KL}(\pi_{\theta}(a|s) || \pi_0(a|s)) \right] + \eta \mathbb{E}_{(s,a) \sim \mathcal{D}} \left[ \log \pi_{\theta}(a|s) \right],$$

where  $\lambda$  is the KL coefficient and  $\eta$  is the MLE coefficient. For all of our proposed RLGF algorithms discussed in section 4 we consider setting  $\pi^g$  to the supervised fine-tuned model (SFT) with nucleus

<sup>1</sup>in our experiments, instead of using TRPO, we use PPO – a scalable version of TRPO that is more suitable for high-dimensional problems. However we emphasize the TRPO and PPO use the same principle for policy optimization: make conservative policy update (Kakade & Langford, 2002) to ensure monotonic improvement.

Algorithms	IMDB Sentiment			CommonGen			
	Semantic and Fluency Metrics			Lexical and Semantic Metrics			
	Sentiment Score (↑)	Perplexity (↓)	Output-Perplexity (↓)	Bleu-4 (↑)	BERTScore (↑)	CIDEr-D (↑)	SPICE (↑)
Zero-Shot	0.48 ± 0.00	32.55 ± 0.00	5.64 ± 0.00	0.16	0.93	1.10	0.26
SFT	0.55 ± 0.00	35.67 ± 0.00	6.19 ± 0.00	0.22	0.95	1.43	0.31
SFT+PPO	0.97 ± 0.01	44.92 ± 1.78	3.17 ± 0.62	0.26	0.95	1.65	0.32
SFT+PPO++	0.97 ± 0.01	44.83 ± 2.10	3.34 ± 0.80	0.27	0.95	1.68	0.32
SFT+AggreVaTeD	0.95 ± 0.03	52.56 ± 5.38	5.04 ± 2.30	0.27	0.95	1.65	0.32
SFT+LOLS	0.93 ± 0.05	53.30 ± 16.70	3.44 ± 4.96	0.26	0.95	1.66	0.32
SFT+D <sup>2</sup> LOLS	0.97 ± 0.00	43.88 ± 2.37	2.92 ± 0.13	0.27	0.95	1.69	0.33

Table 1: **IMDB and CommonGen Results:** We compute the mean and standard deviation over 3 seeds for the IMDB task and compute 1 seed for the CommonGen task. For our reward function each task we use the bold metric(s). The zero-shot model is the performance of the pretrained model used for IMDB and CommonGen, GPT-2 and T5 respectively. SFT+Alg indicates running Alg after supervised finetuning. SFT+nucleus is used as our guide policy  $\pi^g$  for all experiments.

sampling for decoding (i.e.,  $\pi^g = \text{SFT+nucleus}$ ). We treat SFT+nucleus as a black-box model that we can only query for text generation and do not perform updates to it. By using SFT+nucleus as our guide policy, we run all of our experiments under the exact same conditions as those of RLHF. Note, RLHF already requires keeping SFT to compute the KL constraint,  $KL(\pi_\theta || \pi_0)$ , in  $J_{ppo}$ .

**Task Details** In our experiments, *perplexity* measures how likely our learned model,  $\pi_\theta$ , is to generate the references in the task dataset, whereas *output perplexity* computes how likely a general LLM (e.g. GPTJ) is to generate the generations from our learned policy,  $\pi_\theta$ . Both perplexity metrics have been reported as a measure of fluency (Fedus et al., 2018; Ramamurthy et al., 2022).

We perform experiments on three tasks. IMDB is the first task and the objective is to generate fluent and positively sentiment-ed text continuations for IMDB (Maas et al., 2011) movie reviews prompts. We use a sentiment classifier (Sanh et al., 2019) as our reward function that is trained on review texts and sentiment labels from the dataset, which then provides sentiment scores indicating how positive a given piece of text is. For training supervised SFT baselines, we consider only the examples with positive labels. We chose GPT2 (Radford et al., 2019) as the base language model (LM) for this task. We evaluate all algorithms on three metrics: sentiment reward score, perplexity, and output-perplexity.

Next, we consider CommonGen (Lin et al., 2020b), a challenging constrained, text generation task that tests the ability of generative common sense reasoning. We optimize the SPIDER (Liu et al., 2017) reward function, a weighted combination of the CIDEr-D and SPICE metric. We chose T5-base (Raffel et al., 2020) as our base LLM and prefixed each concept set input with: "generate a sentence with:". We evaluate on four metrics: BLEU (Papineni et al., 2002), CIDEr-D (Vedantam et al., 2015), SPICE (Anderson et al., 2016), and BERTScore (Zhang et al., 2019a).

The final task we consider is Reddit TL;DR summarization dataset (Völske et al., 2017) where the objective is to generated summaries. We use the filtered dataset with additional human preference data used in Stiennon et al. (2020). The base LLM that we use for this task is GPT-J (Wang & Komatsuzaki, 2021) and we train all models using LoRA adapters (Hu et al., 2021). We evaluate all algorithms on 5 metrics: reward score, perplexity, output-perplexity, win rate and Rouge (Lin, 2004). For win rate, we use the open source Llama2-13B-chat (Touvron et al., 2023) model as our evaluator model. We compare all algorithm generations to the preferred summary references. Refer to Appendix C.2, for the exact Win Rate prompt, example evaluations and implementation details.

## 6.1 EXPERIMENTAL RESULTS

**RLGF vs. RLHF Performance** Table 1 and Table 2 compares all of the RLGf algorithms proposed in Section 4 against standard RLHF algorithms and baselines. For all tasks, our  $\pi^g$  is SFT which is sub-optimal, performing worse than all RL based algorithms across most lexical and semantic metrics. Utilizing this  $\pi^g$ , for IMDB, SFT+D<sup>2</sup>LOLS and PPO++ outperform PPO, and for CommonGen, D<sup>2</sup>LOLS outperforms PPO. Finally, for TL;DR summarization we see that PPO++ performs better than PPO as well as a competitive baseline, Best-of-N (Dubois et al., 2023).

Algorithms	TL;DR Summarization						
	Semantic and Fluency Metrics						
	RM Score (↑)	Perplexity (↓)	Output-Perplexity (↓)	Win Rate (↑)	Rouge 1 (↑)	Rouge 2 (↑)	RougeL (↑)
Zero-Shot	1.57	14.07	11.51	44.12%	0.27	0.07	0.18
SFT	5.68	14.09	12.81	44.29%	0.34	0.25	0.25
Best-of-N ( $N = 8$ )	5.98	14.09	12.86	47.60%	0.36	0.13	0.27
SFT+PPO	6.01	15.05	17.67	54.25%	0.35	0.13	0.27
SFT+PPO <sup>++</sup>	6.11	14.53	16.15	55.01%	0.36	0.14	0.27
SFT+AggreVaTeD	5.93	14.69	16.41	48.98%	0.36	0.15	0.29

Table 2: **TL;DR Summarization Results:** We report the mean over 1 seed. Our RM Score is under our trained preference reward model and the Win Rate is evaluated by Llama2-13B-Chat. We use SFT+nucleus as  $\pi^g$ .

Supporting our justification from Section 5, AggreVaTeD improves beyond our guide policy, providing an alternative as a warm-starting methodology to warm-starting with SFT. As shown by Table 7, we see that warm-starting with AggreVaTeD leads to higher performance on IMDB than warm-starting with SFT, a popular learning strategy when performing RL for language (Stiennon et al., 2020; Ouyang et al., 2022). PPO<sup>++</sup>, on the other hand, is better than or competitive to our RL baseline demonstrating a simple, yet powerful alternative to PPO as the RL procedure. Even in practice, we observe the benefit of restarting from an initial state distribution that better covers an optimal policy’s state distribution. The combination of these two, D<sup>2</sup>LOLS, achieves the best of both worlds and fully leverages the capabilities of utilizing a guide policy.

**Reward Optimization Tradeoff** In Figure 2 we evaluate how well RLGF algorithms trade-off optimizing the reward while minimizing the perplexity and kl-constraint  $\sqrt{KL}$ . For both plots, the top right corner indicates the policy has both high reward and low perplexity and low divergence from  $\pi_0$ . For each algorithm we plot 5 checkpoints ranging from 20 to 100 iterations. PPO<sup>++</sup> mostly matches or has higher reward than PPO while maintaining a lower perplexity. Separately, AggreVaTeD trade-offs reward for perplexity, and has comparable reward scores as PPO while drastically reducing its perplexity. For the kl-constraints plot on the left of Figure 2 we see that although PPO has a set of points with high reward, most of these points also have high KL divergences. Whereas, a subset of PPO<sup>++</sup> matches or has higher reward than PPO while having a lower kl-constraint.

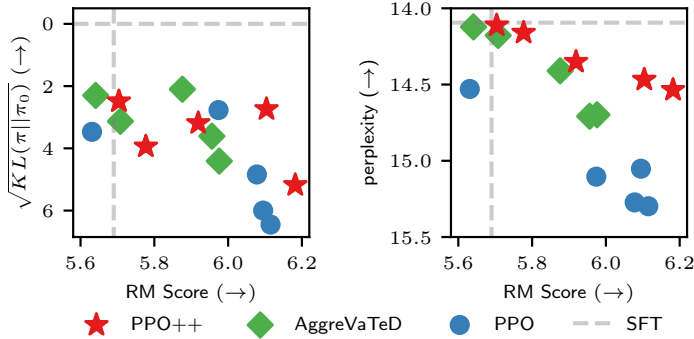


Figure 2: We investigate the reward optimization, kl-constraint, and fluency trade-off in our TL;DR summarization task. The dashed line represents our SFT policy’s performance across each metric. Both PPO<sup>++</sup> and AggreVaTeD learn a policy that has a better trade-off than PPO.

**RLGF Performance on Difficult Prompts** Our evaluation was carried out on the CommonGen task where we categorized the prompts based on their difficulty level. For CommonGen, we classify the prompts into *easy* and *hard* based on the number of unseen concepts in the prompt. Specifically, we categorized prompts with 3 concepts as easy and more than 3 concepts as hard. Figure 3 presents a comparison of scores for different algorithms grouped by prompt difficulty. The results reveal a notable performance gap between easy and hard prompts for algorithms such as SFT and PPO,



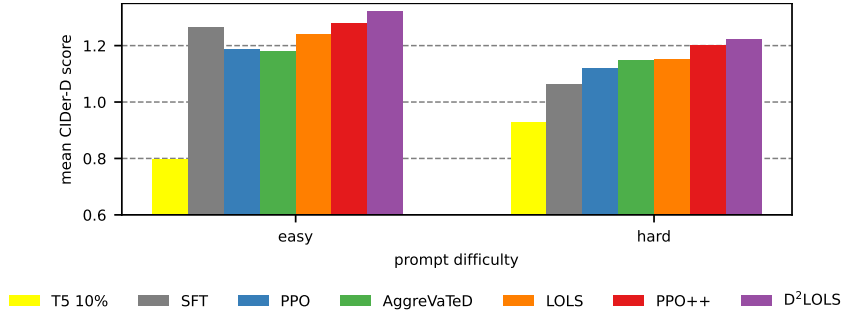


Figure 3: Comparison of CIDEr-D scores grouped by prompt difficulty on CommonGen. The performance gap between easy and hard prompts is evident for SFT, and PPO++, while our proposed algorithms AggreVaTeD, LOLS and D<sup>2</sup>LOLS exhibit a significantly smaller gap, showcasing their effectiveness on challenging prompts.

whereas our proposed algorithms PPO++, AggreVaTeD, LOLS and D<sup>2</sup>LOLS exhibit a significantly smaller gap, with D<sup>2</sup>LOLS having the least gap. In other words, even on challenging prompts, our interactive algorithms produce better text continuations. See Appendix E for example generations.

**MLE and KL coefficient Sensitivity** We test the sensitivity of PPO and RLGf algorithms to two regularization hyperparameters in the  $J_{ppo}$  objective, namely the KL coefficient,  $\lambda$ , and the MLE coefficient,  $\eta$ . The left 2 plots in Figure 4 show the reward and perplexity when we keep  $\eta$  fixed and vary  $\lambda$  while the right 2 show the performance when we keep  $\lambda$  fixed and vary  $\eta$ . All RL algorithms are robust to varying KL coefficients. We observe much more instability when relaxing our MLE regularization with PPO and RLGf algorithm’s perplexities blowing up.

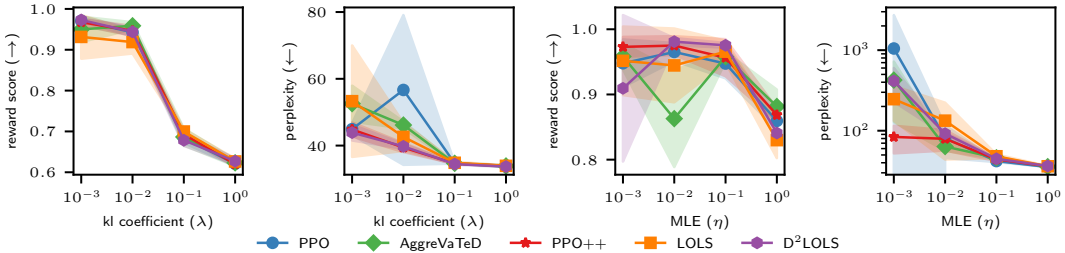


Figure 4:  $J_{ppo}$  KL coefficient ( $\lambda$ ) and MLE coefficient ( $\eta$ ) ablation. We show the sensitivity of PPO and RLGf algorithms to each regularization term in the objective. Note that all RL algorithms are robust to changes in KL coefficient with relatively minor changes in the Perplexity while being more sensitive to changes in MLE objective (Right) with blowups in the perplexity.

## 7 CONCLUSION AND FUTURE WORK

We presented a unifying framework of incorporating a guide policy to enhance reinforcement learning for natural language generation. Through theoretical justification and experimental validation, we demonstrate that our RLGf framework can outperform PPO for fine-tuning LLMs. Our proposed algorithms PPO++ and D<sup>2</sup>LOLS only require black-box access to the guide policy and are conceptually simple and easy to implement based on PPO. While in our experiment, we demonstrate that supervised fine-tuned models with standard decoding strategies is a good candidate of the guide policy, our framework is general enough to leverage any large LLMs as the guide policy, including those that are not open-sourced. Finally, RLGf’s contributions to the broader large language model literature is complementary to model enhancements, dataset improvements, and prompting discoveries such as in-context prompting. We leave it to exciting future work to test the full capabilities of bootstrapping the state-of-the-art advancements in each research direction with RLGf to improve reinforcement learning for natural language generation.

## REFERENCES

- Alekh Agarwal, Nan Jiang, and Sham M Kakade. Reinforcement learning: Theory and algorithms. 2019.
- Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. Optimality and approximation with policy gradient methods in markov decision processes. In *Conference on Learning Theory*, pp. 64–66. PMLR, 2020.
- Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distribution Shift. *Journal of Machine Learning Research*, 22(1):4431–4506, 2021.
- Rishabh Agarwal, Nino Vieillard, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. Gkd: Generalized knowledge distillation for auto-regressive sequence models. *arXiv preprint arXiv:2306.13649*, 2023.
- Prithviraj Ammanabrolu and Mark O Riedl. Playing text-adventure games with graph-based deep reinforcement learning. *arXiv preprint arXiv:1812.01628*, 2018.
- Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic Propositional Image Caption Evaluation. In *Proceedings of European Conference on Computer Vision*, 2016.
- James Bagnell, Sham M Kakade, Jeff Schneider, and Andrew Ng. Policy Search by Dynamic Programming. In *Proceedings of Neural Information Processing Systems*, 2003.
- Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.
- Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.
- Michiel Bakker, Martin Chadwick, Hannah Sheahan, Michael Tessler, Lucy Campbell-Gillingham, Jan Balaguer, Nat McAleese, Amelia Glaese, John Aslanides, Matt Botvinick, et al. Fine-tuning language models to find agreement among humans with diverse preferences. *Advances in Neural Information Processing Systems*, 35:38176–38189, 2022.
- Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.
- Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. In *Proceedings of Neural Information Processing Systems*, 2015.
- Dimitri P Bertsekas. Approximate Policy Iteration: A Survey and Some New Methods. *Journal of Control Theory and Applications*, 9(3):310–335, 2011.
- Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.
- Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of Artificial General Intelligence: Early Experiments with Gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.
- Cristian Buciluă, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.
- CarperAI. <https://github.com/carperai/trlx>, 2023.

- Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. Learning to Search Better than your Teacher. In *Proceedings of International Conference on Machine Learning*, 2015.
- Angelica Chen, Jérémy Scheurer, Tomasz Korbak, Jon Ander Campos, Jun Shern Chan, Samuel R Bowman, Kyunghyun Cho, and Ethan Perez. Improving code generation by training with natural language feedback. *arXiv preprint arXiv:2303.16749*, 2023.
- Ching-An Cheng, Xinyan Yan, Nolan Wagener, and Byron Boots. Fast Policy Learning through Imitation and Reinforcement. *arXiv preprint arXiv:1805.10413*, 2018.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.
- Michael Collins and Brian Roark. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pp. 111–118, 2004.
- Marc-Alexandre Côté, Akos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. Textworld: A learning environment for text-based games. In *Computer Games: 7th Workshop, CGW 2018, Held in Conjunction with the 27th International Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, July 13, 2018, Revised Selected Papers 7*, pp. 41–75. Springer, 2019.
- Hal Daumé, John Langford, and Daniel Marcu. Search-based Structured Prediction. *Machine learning*, 75:297–325, 2009.
- Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. AlpacaFarm: A simulation framework for methods that learn from human feedback, 2023.
- Daniel Duckworth, Arvind Neelakantan, Ben Goodrich, Lukasz Kaiser, and Samy Bengio. Parallel Scheduled Sampling. *arXiv preprint arXiv:1906.04331*, 2019.
- Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.
- William Fedus, Ian Goodfellow, and Andrew M Dai. MaskGAN: better text generation via filling in the\_. *arXiv preprint arXiv:1801.07736*, 2018.
- Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Conference on robot learning*, pp. 482–495. PMLR, 2017.
- Github. <https://github.com/features/copilot>, 2023. Accessed: 2023-May-13.
- Dongyoung Go, Tomasz Korbak, Germán Kruszewski, Jos Rozen, Nahyeon Ryu, and Marc Dymetman. Aligning language models with preferences through f-divergence minimization. *arXiv preprint arXiv:2302.08215*, 2023.
- Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. Differentiable Scheduled Sampling for Credit Assignment. *arXiv preprint arXiv:1704.06970*, 2017.
- Tanya Goyal, Junyi Jessy Li, and Greg Durrett. News Summarization and Evaluation in the Era of Gpt-3. *arXiv preprint arXiv:2209.12356*, 2022.
- Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pp. 1352–1361. PMLR, 2017.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

- Braden Hancock, Martin Bringmann, Paroma Varma, Percy Liang, Stephanie Wang, and Christopher Ré. Training classifiers with natural language explanations. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2018, pp. 1884. NIH Public Access, 2018.
- Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, et al. Grounded Language Learning in a Simulated 3D World. *arXiv preprint arXiv:1706.06551*, 2017.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner Monologue: Embodied Reasoning through Planning with Language Models. In *Proceedings of Annual Conference on Robot Learning*, 2022.
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.
- Sham Kakade and John Langford. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of International Conference on Machine Learning*, 2002.
- Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. A distributional approach to controlled text generation. *arXiv preprint arXiv:2012.11635*, 2020.
- Khan Academy. <https://blog.khanacademy.org/harnessing-ai-so-that-all-students-benefit-a-nonprofit-approach-for-equal-access/>, 2023. Accessed: 2023-May-14.
- Samuel Kiegl and Julia Kreutzer. Revisiting the weaknesses of reinforcement learning for neural machine translation. *arXiv preprint arXiv:2106.08942*, 2021.
- Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*, 2016.
- Tomasz Korbak, Hady Elsahar, Germán Kruszewski, and Marc Dymetman. On reinforcement learning and distribution matching for fine-tuning language models with no catastrophic forgetting. *Advances in Neural Information Processing Systems*, 35:16203–16220, 2022.
- Julia Kreutzer, Shahram Khadivi, Evgeny Matusov, and Stefan Riezler. Can neural machine translation be improved with user feedback? *arXiv preprint arXiv:1804.05958*, 2018a.
- Julia Kreutzer, Joshua Uyheng, and Stefan Riezler. Reliability and learnability of human bandit feedback for sequence-to-sequence reinforcement learning. *arXiv preprint arXiv:1805.10627*, 2018b.
- Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models. *arXiv preprint arXiv:2303.00001*, 2023.
- Rémi Leblond, Jean-Baptiste Alayrac, Anton Osokin, and Simon Lacoste-Julien. SEARNN: Training RNNs with Global-local losses. *arXiv preprint arXiv:1706.04499*, 2017.
- Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023a.
- Peter Lee, Sebastien Bubeck, and Joseph Petro. Benefits, Limits, and Risks of GPT-4 as an AI Chatbot for Medicine. *New England Journal of Medicine*, 388(13):1233–1239, 2023b.

- Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep Reinforcement Learning for Dialogue Generation. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2016.
- Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. *arXiv preprint arXiv:2009.07253*, 2020a.
- Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. CommonGen: A Constrained Text Generation Challenge for Generative Commonsense Reasoning. In *Findings of Association for Computational Linguistics: EMNLP*, 2020b.
- Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.
- Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. Improved Image Captioning via Policy Gradient Optimization of Spider. In *Proceedings of International Conference on Computer Vision*, 2017.
- Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*, 2023.
- Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *Proceedings of Annual Meeting of the Association for Computational Linguistics: Human language technologies*, 2011.
- Tsvetomila Mihaylova and André FT Martins. Scheduled Sampling for Transformers. *arXiv preprint arXiv:1906.07651*, 2019.
- Dipendra Misra, John Langford, and Yoav Artzi. Mapping Instructions and Visual Observations to Actions with Reinforcement Learning. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2017.
- Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*, 2023.
- Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 6292–6299. IEEE, 2018.
- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.
- Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. Language Understanding for Text-based Games using Deep Reinforcement Learning. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2015.
- Khanh Nguyen, Hal Daumé III, and Jordan Boyd-Graber. Reinforcement learning for bandit neural machine translation with simulated human feedback. *arXiv preprint arXiv:1707.07402*, 2017.
- OpenAI. <https://openai.com/blog/chatgpt>, 2023.
- OpenAI. Gpt-4 technical report, 2023.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training Language Models to Follow Instructions with Human Feedback. In *Proceedings of Neural Information Processing Systems*, 2022.
- Richard Yuanzhe Pang and He He. Text Generation by Learning from Demonstrations. In *Proceedings of International Conference on Learning Representations*, 2021.

- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of Association for Computational Linguistics*, 2002.
- Ivaylo Popov, Nicolas Heess, Timothy Lillicrap, Roland Hafner, Gabriel Barth-Maron, Matej Vecerik, Thomas Lampe, Yuval Tassa, Tom Erez, and Martin Riedmiller. Data-efficient deep reinforcement learning for dexterous manipulation. *arXiv preprint arXiv:1704.03073*, 2017.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL <http://jmlr.org/papers/v21/20-074.html>.
- Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.
- Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is Reinforcement Learning (Not) for Natural Language Processing?: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization. *arXiv preprint arXiv:2210.01241*, 2022.
- Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.
- Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Conference on empirical methods in natural language processing*, 1996.
- Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. Deep Reinforcement Learning-Based Image Captioning With Embedding Reward. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- Paul Roit, Johan Ferret, Lior Shani, Roei Aharoni, Geoffrey Cideron, Robert Dadashi, Matthieu Geist, Sertan Girgin, Léonard Hussenot, Orgad Keller, et al. Factually consistent summarization via reinforcement learning with textual entailment feedback. *arXiv preprint arXiv:2306.00186*, 2023.
- Stephane Ross and J Andrew Bagnell. Reinforcement and Imitation Learning via Interactive No-regret Learning. *arXiv preprint arXiv:1406.5979*, 2014.
- Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-regret Online Learning. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, 2011.
- Tim Salimans and Richard Chen. Learning montezuma’s revenge from a single demonstration. *arXiv preprint arXiv:1812.03381*, 2018.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. *arXiv preprint arXiv:1910.01108*, 2019.
- Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. Training language models with language feedback at scale. *arXiv preprint arXiv:2303.16755*, 2023.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust Region Policy Optimization). In *Proceedings of International Conference on Machine Learning*, 2015.

- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum risk training for neural machine translation. *arXiv preprint arXiv:1512.02433*, 2015.
- Charlie Snell, Ilya Kostrikov, Yi Su, Mengjiao Yang, and Sergey Levine. Offline rl for natural language generation with implicit language q learning. *arXiv preprint arXiv:2206.11871*, 2022.
- Artem Sokolov, Stefan Riezler, and Tanguy Urvoy. Bandit structured prediction for learning from partial feedback in statistical machine translation. *arXiv preprint arXiv:1601.04468*, 2016.
- Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to Summarize with Human Feedback. *In Proceedings of Neural Information Processing Systems*, 2020.
- Theodore R Sumers, Mark K Ho, Robert D Hawkins, Karthik Narasimhan, and Thomas L Griffiths. Learning rewards from linguistic feedback. *In Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 6002–6010, 2021.
- Wen Sun, Arun Venkatraman, Geoffrey J Gordon, Byron Boots, and J Andrew Bagnell. Deeply Aggregated: Differentiable Imitation Learning for Sequential Prediction. *In Proceedings of International Conference on Machine Learning*, 2017.
- Wen Sun, J Andrew Bagnell, and Byron Boots. Truncated Horizon Policy Search: Combining Reinforcement Learning & Imitation Learning. *arXiv preprint arXiv:1805.11240*, 2018.
- Arash Tavakoli, Vitaly Levnik, Riashat Islam, Christopher M Smith, and Petar Kormushev. Exploring restart distributions. *arXiv preprint arXiv:1811.11298*, 2018.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based Image Description Evaluation. *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4566–4575, 2015.
- Arun Venkatraman, Martial Hebert, and J Bagnell. Improving multi-step prediction of learned time series models. *In Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.
- Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. TL;DR: Mining Reddit to learn automatic summarization. *In Proceedings of the Workshop on New Frontiers in Summarization*, pp. 59–63, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4508. URL <https://aclanthology.org/W17-4508>.
- Ben Wang and Aran Komatsuzaki. Gpt-j-6b: A 6 billion parameter autoregressive language model, 2021.
- Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.
- Sam Wiseman and Alexander M Rush. Sequence-to-Sequence Learning as Beam-Search Optimization. *In Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2016.
- Jeff Wu, Long Ouyang, Daniel M Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. Recursively summarizing books with human feedback. *arXiv preprint arXiv:2109.10862*, 2021.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

- Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. Rlcd: Reinforcement learning from contrast distillation for language model alignment. *arXiv preprint arXiv:2307.12950*, 2023.
- Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating Text Generation with Bert. *arXiv preprint arXiv:1904.09675*, 2019a.
- Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. Bridging the gap between training and inference for neural machine translation. *arXiv preprint arXiv:1906.02448*, 2019b.
- Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models. In *Proceedings of International Conference on Learning Representations*, 2022.
- Xingxing Zhang and Mirella Lapata. Sentence Simplification with Deep Reinforcement Learning. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2017.
- Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.
- Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017.
- Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*, 2023.
- Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning Language Models from Human Preferences. *arXiv preprint arXiv:1909.08593*, 2019.