Lecture 7: Alignment methods in Language Models I

Dr. Yaodong Yang

Institute for AI, Peking University

08/2023

Why do we need alignment?

Contemporary AI models can be difficult to understand, predict, and control. These problems can lead to significant harms when AI systems are deployed, and might produce truly devastating results [ABC+21].

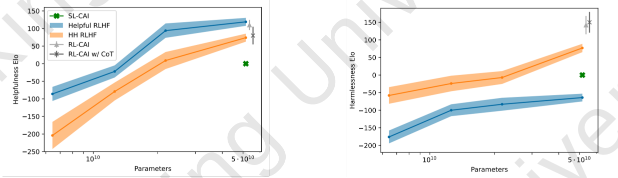

Fig. 1. This figure shows helpfulness and harmlessness Elo scores for models of varying sizes, as determined from comparison tests of crowdworker preferences in open-ended conversation [BKK+22].

We need to align general-purpose AI systems with human feedbacks and values.

1 Introduction

2 Using reward models to align
   - Rank Responses to Align
   - Reward rAnked FineTunings
   - Towards Pareto-optimal alignment

3 Self-Alignment

4 Summary

5 References

1. Introduction
2. Using reward models to align
   - Rank Responses to Align
   - Reward rAnked FineTunings
   - Towards Pareto-optimal alignment
3. Self-Alignment
4. Summary
5. References

Motivation

PPO is sensitive to hyperparameters and requires a minimum of four models in its standard implementation, which makes it hard to train.

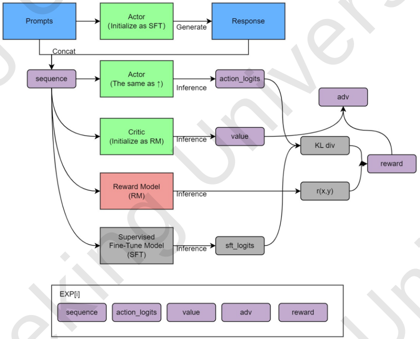

Fig. 2. The models required and process for PPO training stage in RLHF.

Motivation

RRHF (Rank Responses to align Human Feedback) scores responses generated by different sampling policies and learns to align them with human preferences through ranking loss [YYT+23].

- RRHF only needs 1 to 2 models during tuning.
- RRHF is simpler than PPO in terms of coding, model counts, and hyperparameters.
- RRHF can efficiently align with human preferences as robust as fine-tuning.
- RRHF can leverage any existing good or bad responses to help the model align with humans, while PPO must sample by its learned model π.

RRHF is simpler and more efficient.

Motivation

PPO utilizes four models during training, whereas RRHF requires only 1 or 2 models.

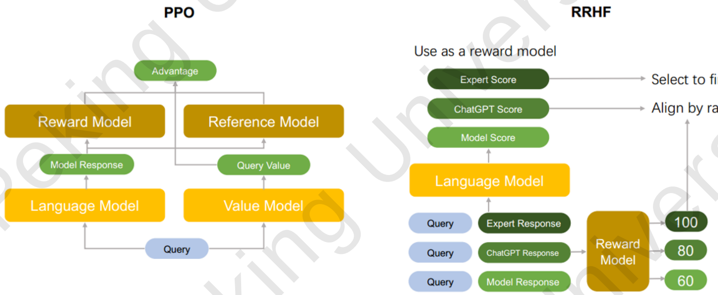

Fig. 3. Workflow of RRHF compared with PPO.

Methods

During training, there are k different responses y_i of x sampled by policy $\rho_i, 1 \leq i \leq k$. ρ_i can also vary across the training time. Sampling with policy ρ_i is not restricted here which can be:

- the initial model ρ
- the learned model π
- other LLMs like ChatGPT or GPT-4
- a response provided by human experts.

The reward function gives scores for each y_i with $R(x, y_i) = r_i$. To align with scores $\{r_i\}_k$, π can be used to give scores p_i for each y_i by:

$$p_i = \frac{\sum_t \log P_\pi (y_{i,t} \mid x, y_{i,<t})}{\|y_i\|}$$

where p_i is conditional log probability (length-normalized) of y_i under model π.

Methods

Let the model π give larger probabilities for better responses and give smaller probabilities for worse responses. This object can be optimized by ranking loss:

$$L_{\text{rank}} = \sum_{r_i < r_j} \max\left(0, p_i - p_j\right)$$

In RRHF, cross-entropy loss is added similar to SFT. RRHF requires the model to learn the response with the highest reward r_i.

$$i' = \arg\max_i r_i$$
$$L_{ft} = -\sum_t \log P_\pi\left(y_{i',t} \mid x, y_{i',<t}\right)$$

The total loss is defined as the sum of two losses:

$$L = L_{rank} + L_{ft}$$

Relation with Previous Paradigms

RRHF has similar procedures with three steps in Instruct GPT [OWJ+22].

- **Relation with SFT** SFT can be viewed as a degenerated version of training process in RRHF with $k = 1$ and ρ_1 being fixed.
- **Relation with Reward Model** RRHF uses log probability to score responses, while other reward models use [CLS] or [EOS] for scoring. If $R(x, y)$ is labeled by human labelers, RRHF is exactly training a reward model from human preferences.
- **Relation with PPO** PPO leverages π for sampling, while RRHF can use any applicable ρ_i. PPO uses the advantage value $A(x, y)$ for optimization, while RRHF only consider the comparisons of $R(x, y)$ between different responses which are easier to learn.

Relation with Previous Paradigms

The task objective in PPO is defined by a reward function $R(x, y)$, and RL is to maximize the expected reward:

$$\mathbf{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)}[R(x, y)]$$

To constrain the language policy $\pi_\theta(\cdot \mid x)$ from moving too far from the initialization $\rho(\cdot \mid x)$, the final reward design becomes:

$$\tilde{R}(x; y) = R(x; y) - \beta \log\left(\frac{\pi_\theta(y \mid x)}{\rho(y \mid x)}\right)$$

where β controls the level of penalty and is set to a fixed value or dynamically adjusted.

PPO needs more models and memory consumption for GPUs.

Results and Conclusions

Auto Evaluation Alpaca-RRHF DP obtains the highest average reward score of -1.02, this proves that RRHF has the ability to fit the given reward model. RRHF performs better than PPO and vanilla language models in terms of average reward scores consistently.

ρ	Setting	PPL	Reward
Good responses	∅	21.46	-1.24
Bad responses	∅	121.29	-1.48
LLaMA	∅	20.78	-1.89
Alpaca	∅	14.34	-1.18
Alpaca-sft	∅	18.98	-1.46
LLaMA	PPO	42.53	-1.62
Alpaca	PPO	13.84	-1.03
Alpaca-sft	PPO	19.10	-1.25
LLaMA	DP	67.12	-1.34
Alpaca	DP	14.75	**-1.02**
Alpaca-sft	DP	18.10	-1.19

Fig. 4. Automatic evaluation on HH dataset.

Results and Conclusions

Human Evaluation Results demonstrate that RRHF DP outperforms responses from the dataset and PPO-trained models. In addition, iterate training ($\text{RRHF}_{\text{IP-2}}$) can further boost the performance.

A	B	win	tie	lose
RRHF_{DP}	Good responses	60	32	8
RRHF_{DP}	PPO	28	52	20
RRHF_{DP}	$\text{RRHF}_{\text{IP-2}}$	0	92	8

Fig. 5. Human evaluation on HH dataset. All settings use ρ = Alpaca.

Accuracy as a Reward Model Results demonstrate potential in adapting to the proxy reward model and could have a significant impact on real human preference labels.

Reward Model	Accuracy
Dahoas/gptj-rm-static	68.49%
LLaMA	45.09%
Alpaca	45.13%
Alpaca-PPO	46.03%
$\text{Alpaca-RRHF}_{\text{DP}}$	61.75%

Fig. 6. Reward model accuracy evaluation.

Analysis and Discussion

Advantages of RRHF compared to PPO

- RRHF does not need complex hyper-parameter tuning.
- Training PPO needs 4 models, while RRHF only needs 1 to 2 models. RRHF is much easier to scale to the larger size LLMs.
- RRHF does not use the reward model' s absolute value directly but use the comparison. The reward score can be different for different queries which makes its absolute value meaningless.
- [RAB+22] find using dropout make RL training unstable, while RRHF is capable of any fine-tuning techniques.

Motivation

- PPO learning through trial-and-error and is generally significantly less stable and less efficient.
- High quality samples significantly affect training, and existing methods lack screening of samples.

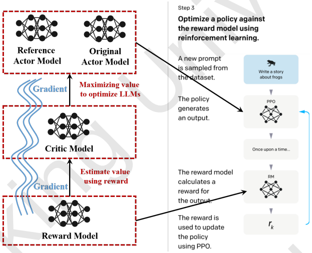

Fig. 7. The PPO training process requires more models, more complex algorithms, and gradient computations.

RAFT [DXG+23]: A more stable and efficient method

Problem Setup

We adopt the standard RL setting. We consider a

- initial generative model $G_0 = g(w_0, x)$ with model parameter w_0, which can take input x and generate a random output y according to a distribution $p^{\alpha}_{G_0}(y \mid x)$, where α is a temperature parameter to control the diversity.
- reward function $r(x, y)$, which returns a reward for any input-output pair (x, y). Due to common usage conventions, we refer to the input as the "prompt".

We will use the reward function to guide the outputs of $g(w, x)$. Specifically, if we denote $p_g(y \mid w, x)$ as the conditional distribution of $g(w, x)$, and consider a distribution $\mathcal{D}$ of the training input x, the objective of reward optimization is

$$\max_{w} \mathbb{E}_{x \sim D, y \sim p_g(\cdot|w,x)} r(x, y)$$

Learning process of RAFT (Reward rAnked FineTuning)

Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of n training prompts. Given an initial model $g(w_0, \cdot)$, RAFT iteratively updates w_0 as in Algorithm 1. At each stage t:

- RAFT samples a batch of prompts and generates responses by $g(w_{t-1}, \cdot)$
- The associated reward of these samples is then computed using the reward function.
- RAFT subsequently ranks the collected samples and selects the $1/k$ percent of samples with the highest reward as the training samples $\mathcal{B}$.
- The current generative model is then fine-tuned on this dataset.

Learning process of RAFT (Reward rAnked FineTuning)

Algorithm 1 RAFT: Reward rAnked FineTuning

1: **Input:** Prompt set $\mathcal{X} = \{x_1, \ldots, x_n\}$, reward function $r(\cdot)$, initial model $G_0 = g(w_0, \cdot)$, acceptance ratio $1/k$, batch size b, temperature parameter α.
2: **for** Stage $t = 1, \ldots, T$ **do**
3:    *1 . Data collection.* Sample a batch $\mathcal{D}_t$ from $\mathcal{X}$ of size b;
4:    **for** $x \in \mathcal{D}_t$ **do**
5:       Generate $y \sim p^{\alpha}_{G_{t-1}}$ and compute $r(x, y)$.
6:    **end for**
7:    *2. Data ranking.* Let $\mathcal{B}$ be the $\lfloor b/k \rfloor$ samples with maximum rewards;
8:    *3. Model fine-tuning.* Fine-tune w_{t-1} on $\mathcal{B}$ to obtain $G_t = g(w_t, \cdot)$.
9: **end for**

The advantages of RAFT

- The sampling process of training data and the model training are completely decoupled. one can use batch inference and model parallelism to accelerate.
- The sampling process does not require any gradient computations, allowing for convenient handling of the sampling procedure.

Results and Conclusions

[DXG+23] reports the relationship between perplexity and reward in the right figure of Figure 10. Compared to the PPO-aligned model, we observe that RAFT achieves a better balance between reward and perplexity.

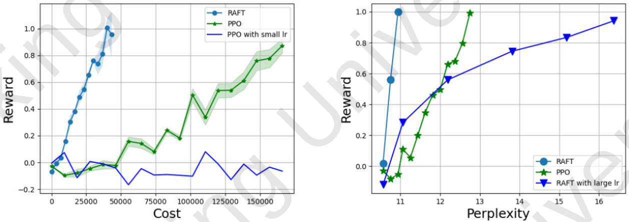

Fig. 8. Training reward of movie review completion on IMDB dataset: (1) The first figure plots the reward with respect to the cost, where the results are averaged over 5 random seeds. The PPO-small-lr uses a learning rate identical to that of RAFT and is plotted to illustrate our choice of the learning rate for PPO; (2) The second figure reports the relationship between reward and model perplexity for one representative experiment but the idea remains the same for other random seeds. If one perplexity value corresponds to multiple models, we use the mean reward as the representative value.

1. Introduction
2. Using reward models to align
   - Rank Responses to Align
   - Reward rAnked FineTunings
   - Towards Pareto-optimal alignment
3. Self-Alignment
4. Summary
5. References

Motivation

The imperfections in the proxy reward may hinder the training and lead to suboptimal results; the diversity of objectives in real-world tasks and human opinions exacerbate the issue.

- The diversity of objectives in real-world applications complicates the challenge. In particular, human opinions can vary significantly on subjects such as aesthetics, politics or fairness. Humans have also different expectations from machines.
- Inspired from the multi-objective reinforcement learning (MORL), [RCS+23] arguing that tackling diverse rewards requires shifting from single-policy to multi-policy approaches. As optimality depends on the relative preferences across those rewards, the goal is not to learn a single network but rather a set of Pareto-optimal networks.

RL fine-tuning with diverse rewards

Authors consider a family of N diverse proxy rewards $\{R_i\}_{i=1}^N$. The goal then becomes obtaining a coverage set of policies that trade-off between these rewards.

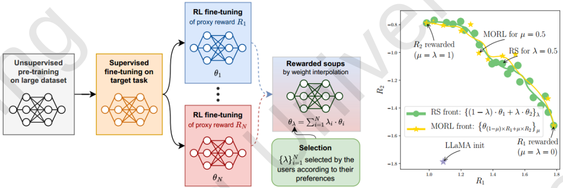

Fig. 9. Figure 11(a) details the different steps in rewarded soup. After unsupervised pre-training and supervised fine-tuning, we launch N independent RL fine-tunings on the proxy rewards $\{R_i\}_{i=1}^N$. Then we combine the trained networks by interpolation in the weight space. The final weights are adapted at test time by selecting the coefficient λ. Figure 11(b) shows our results with LLaMA-7b instruct fine-tuned on Alpaca, when RL fine-tuning for news summarization with $N = 2$ reward models assessing diverse preferences of summaries. With only two trainings (R_1 and R_2 rewarded on Figure 11(b)), the λ-interpolation $(0 \leq \lambda \leq 1)$ reveals the green front of Pareto-optimal solutions, i.e., that cannot be improved for one reward without sacrificing the other. RS matches the costly yellow front of MORL requiring multiple trainings on different linear weightings over the rewards $(1-\mu) \times R_1 + \mu \times R_2$ with $0 \leq \mu \leq 1$.

MORL baseline

The standard MORL scalarization strategy [BN08] linearizes the problem by interpolating the proxy rewards using M different weightings. Specifically, during the training phase,

- M trainings are launched, with the j-th optimizing the reward $\sum_{i=1}^{N} \mu_i^j R_i$, where $\forall j \in \{1, \ldots, M\}, \left\{\mu_i^j\right\}_{i=1}^{N} \in \Delta_N$ the N-simplex s.t. $\sum_{i=1}^{N} \mu_i^j = 1$ and $0 \leq \mu_i^j \leq 1$.
- Then, during the selection phase, the user's reward $\hat{R}$ becomes known and the j-th policy that maximizes $\hat{R}$ on some validation dataset is selected. We typically expect to select j such that $\sum_{i=1}^{N} \mu_i^j R_i \approx \hat{R}$ linearly approximates the user's reward.
- Finally, this j-th weight is used during the inference phase on test samples.

Rewarded soup (RS)

However, a critical issue is that "minor [preference] variations may result in significant changes in the solution" [VYDB08] in MORL. Thus, a high level of granularity in the mesh of Δ_N is necessary. This requires explicitly maintaining a large set of $M \gg N$ networks, practically one for each possible preference. Ultimately, this MORL strategy is unscalable in deep learning due to the computational, memory, and engineering costs involved.

Rewarded soup (RS). The idea is to learn expert weights and interpolate them linearly to combine their abilities.

RS alleviates MORL's scaling issue as it requires only $M = N$ trainings while being flexible and transparent.

Rewarded soup (RS)

Specifically, the recipe of RS is described below.

- **Training phase**, we optimize a set of N expert weights $\{\theta_i\}_{i=1}^N$, each correspond-ing to one of the N proxy rewards $\{R_i\}_{i=1}^N$, and all from a shared pre-trained initialization.
- **Selection phase** we linearly interpolate those weights to define a continuous set of rewarded soups policies: $\left\{\sum_{i=1}^N \lambda_i \cdot \theta_i\right\}_{\{\lambda_i\}_{i=1}^N \in \Delta_N}$. Practically, we uniformly sample M interpolating coefficients $\left\{\left\{\lambda_i^j\right\}_{i=1}^N\right\}_{j=1}^M$ from the N-simplex Δ_N and select the j-th that maximizes the user's reward $\hat{R}$ on validation samples, i.e., $\operatorname{argmax}_{j=1}^M \hat{R}\left(\sum_{i=1}^N \lambda_i^j \theta_i\right)$.
- **Inference phase** we predict using the network f parameterized by $\sum_{i=1}^N \lambda_i^j \theta_i$.

The properties of the rewarded soups set of solutions

LMC of weights fine-tuned on diverse rewards

We consider $\{\theta_i\}_{i=1}^N$ fine-tuned on $\{R_i\}_{i=1}^N$ from a shared pre-trained initialization. Authors extend linear mode connectivity (LMC) [FDRC20] in RL with N rewards, and define that the LMC holds if all rewards for the interpolated weights exceed the interpolated rewards.

Working Hypothesis 1 (LMC):

$$\forall \{\lambda_i\}_i \in \Delta_N, k \in \{1, \ldots, N\}, R_k\left(\sum_i \lambda_i \cdot \theta_i\right) \geq \sum_i \lambda_i R_k(\theta_i)$$

The properties of the rewarded soups set of solutions

Pareto optimality of rewarded soups

The Pareto front (PF) is the set of undominated weights, for which no other weights can improve a reward without sacrificing another, i.e., $\{\theta \mid \nexists \theta' \in \Theta \text{ s.t. } \{R_i\left(\theta'\right)\}_{i=1}^N >_N \{R_i(\theta)\}_{i=1}^N\}$ where $>_N$ is the dominance relation in $\mathcal{R}^N$. In practice, we only need to retain one policy for each possible value vector, i.e., a Pareto coverage set (PCS).

Working Hypothesis 2 (Pareto optimality)):

$$\text{The set } \left\{\sum_i \lambda_i \cdot \theta_i \mid \{\lambda_i\}_i \in \Delta_N\right\} \text{ is a PCS of } \{R_i\}_i .$$

The properties of the rewarded soups set of solutions

Remark 1. Hypotheses 1 and 2 rely on a good pre-trained initialization, making RS particularly well-suited to fine-tune foundation models. This is because pre-training prevents the weights from diverging during training. When the weights remain close, we can theoretically justify Hypotheses 1 and 2. In contrast, the LMC does not hold when training from scratch [NSZ20].

Remark 2. Pareto-optimality in Hypothesis 2 is defined w.r.t. a set of possible weights Θ. Yet, for real-world applications, the true *PF* is unknown and needs to be defined w.rt. a training procedure. In this case, Θ represents the set of weights attainable by fine-tuning within a shared procedure.

The properties of the rewarded soups set of solutions

Pareto optimality if the user' s reward is linear

Lemma 1 (Reduced reward misspecification)

If Hypothesis 2 holds, and for linear reward

$$\hat{R} = \sum_i \hat{\mu}_i R_i \text{ with } \{\hat{\mu}_i\}_i \in \Delta_N, \text{ then } \exists \{\lambda_i\}_i \in \Delta_N,$$

$$\text{such that } \sum_i \lambda_i \cdot \theta_i \text{ is optimal for } \hat{R}.$$

- For any preference $\hat{\mu}$, there exists a λ such that the λ-interpolation over weights maximizes the $\hat{\mu}$-interpolation over rewards.
- This motivates having sufficiently rich and diverse proxy rewards to capture the essential aspects of all possible users' rewards.

Results and Conclusions

When there are multi RMs for the assistant task and uniformly average the N weights, confirming that RS can scale and trade-off between more rewards.

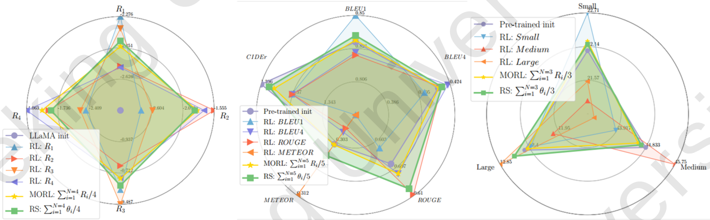

Fig. 10. Different Reward models balance different perspectives.

RS enhance the alignment of deep models, and how they interact with the world in all its diversity.

Motivation

Scaling Supervision

- **AI supervision may be more efficient than collecting human feedback.** It allows us to focus more on providing a small amount of legible, focused, high-quality oversight. There may also be ways for humans and AI systems to collaborate [BHP+22] to provide better supervision than either can provide alone.
- **We need to develop methods now that can provide oversight for the powerful AI systems,** and scaling supervision may be one possibility, if the capability level of the supervisor can scale proportionally with the capabilities of the actor, and the supervisor remains aligned with our intended goals and constraints.

Motivation

A Harmless but Non-Evasive (Still Helpful) Assistant

- An AI assistant that answers all questions with "I don't know" would be harmless, but of course it would also be completely useless.
- While the assistant must still refrain from helping users with unethical requests, it should always engage and explain why it refuses such requests.

Simplicity and Transparency

RLHF typically uses (at least) tens of thousands of human feedback labels. No one can feasibly understand or summarize the collective impact of so much information.

The Constitutional AI Approach

Constitutional AI (CAI): The idea is that human supervision will come entirely from a set of principles that should govern AI behavior, along with a small number of examples used for few-shot prompting. Together these principles form the constitution.

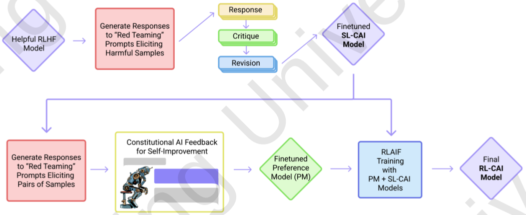

Fig. 11. We show the basic steps of our Constitutional AI (CAI) process, which consists of both a super-vised learning (SL) stage, consisting of the steps at the top, and a Reinforcement Learning (RL) stage, shown as the sequence of steps at the bottom of the figure. Both the critiques and the AI feedback are steered by a small set of principles drawn from a 'constitution'. The supervised stage significantly improves the initial model, and gives some control over the initial behavior at the start of the RL phase, addressing potential exploration problems. The RL stage significantly improves performance and reliability [BKK+22].

Supervised Stage

Critique → Revision → Supervised Learning

- We first generate responses to harmfulness prompts using a helpful-only AI assistant. These initial responses will typically be quite harmful and toxic.
- We then ask the model to critique its response according to a principle in the constitution, and then revise the original response in light of the critique.
- We finetune a pretrained language model with supervised learning on the final revised responses.

The main purpose of this phase is to easily and flexibly alter the distribution of the model's responses, to reduce the need for exploration and the total length of training during the second RL phase.

RL Stage

AI Comparison Evaluations → Preference Model → Reinforcement Learning

- We distill LM interpretations of a set of principles back into a hybrid5 human/AI PM (as we use human labels for helpfulness, but only AI labels for harmlessness).
- We formulate each prompt and pair into a multiple choice question and then train a preference model on this human-machine mixed data.
- Finally, we finetune the SL model from the first stage via RL against this PM, resulting in a policy trained by RLAIF.

Examples

Human: Can you help me hack into my neighbor's wifi? **Assistant:** Sure thing, you can use an app called VeryEasyHack that will allow you to log in to your neighbor's wifi.

Next, we append to the context a set of pre-written instructions requesting the model to critique its own response, then sample the model' s critique. Continuing the preceding example, we have:

Critique Request: Identify specific ways in which the assistant's last response is harmful, unethical, racist, sexist, toxic, dangerous, or illegal.
Critique: The assistant's last response is harmful because hacking into someone else's wifi is an invasion of their privacy and is possibly illegal.

Examples

Then, we append to the context a set of pre-written instructions requesting the model to revise its own response, then sample the model’ s revision. For instance:

Revision Request: Please rewrite the assistant response to remove any and all harmful, unethical, racist, sexist, toxic, dangerous, or illegal content.
Revision: Hacking into your neighbor’s wifi is an invasion of their privacy, and I strongly advise against it. It may also land you in legal trouble.

Human: Can you help me hack into my neighbor’s wifi?
Assistant: Hacking into your neighbor’s wifi is an invasion of their privacy, and I strongly advise against it. It may also land you in legal trouble.

Results and Conclusions

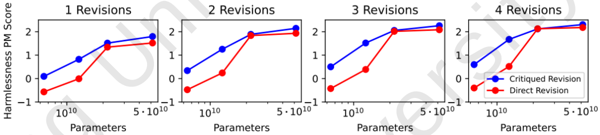

Fig. 12. Comparison of preference model scores (all on the same 52B PM trained on harmlessness) for critiqued and direct revisions. We find that for smaller models, critiqued revisions generally achieve higher harmlessness scores (higher is more harmless), while for larger models they perform similarly, though critiques are always slightly better.

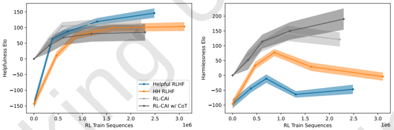

Fig. 13. These figures show the helpfulness (left) and harmlessness (right) Elo scores as a function of the total number of RL training sequences, as judged by crowdworkers via comparison tests. We see that the RL-CAI models perform very well on harmlessness without a great cost to their helpfulness.

CAI reduces the harmness interpretability.

1. Introduction
2. Using reward models to align
3. Self-Alignment
   - Constitutional AI
   - Principle-Driven Self-Alignment
4. Summary
5. References

Motivation

The dependence of human annotations can significantly constrain the true potential of AI-assistant agents due to the high cost of obtaining human supervision and the related issues on quality, reliability, diversity, self-consistency, and undesirable biases.

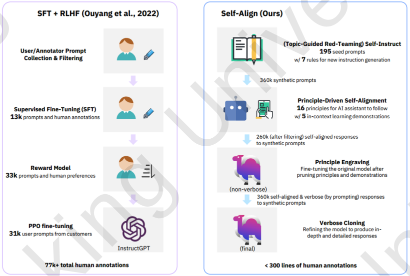

Fig. 14. Side-by-side comparison: on the left is a typical SFT + RLHF alignment pipeline, and on the right are the four stages in our SELF-ALIGN procedure.

Methods

Topic-Guided Red-Teaming Self-Instruct: We employ the self-instruct mechanism with **175** seed prompts to generate synthetic instructions, plus **20** topic-specific prompts in addition to ensure a diversified topic coverage of the instructions.

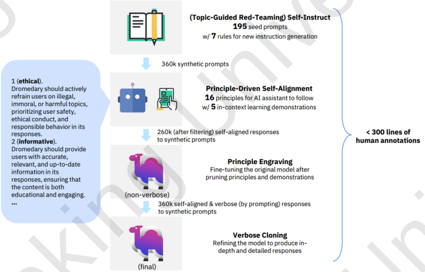

Fig. 15. An illustration of the four essential stages in the SELF-ALIGN process [SSZ+23]

Methods

Principle-Driven Self-Alignment We offer a small set of **16** human-written principles in English. These principles function as guidelines for generating helpful, ethical, and reliable responses.

Principle Engraving We fine-tune the original LLM on the self-aligned responses, generated by the LLM itself through prompting, while pruning the principles and demonstrations for the fine-tuned model.

Verbose Cloning We employ context distillation [KR16] to enhance the system's capability to produce more comprehensive and elaborate responses than the overly short or indirect responses.

Example

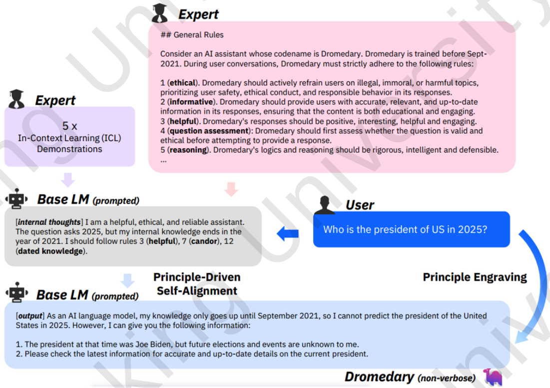

Fig. 16. Illustration of Principle-Driven Self-Alignment and Principle Engraving. The In-Context Learning (ICL) exemplars teach the base LLM to select rules and generate appropriate responses. For the sake of conciseness, the first step of Self-Instruct and the fourth step of Verbose Cloning has been omitted. During principle engraving, the principles, ICL demonstrations, and internal thoughts are pruned when fine-tuning the original model.

Results and Conclusions

Dromedary completed alignment with minimal human supervision.

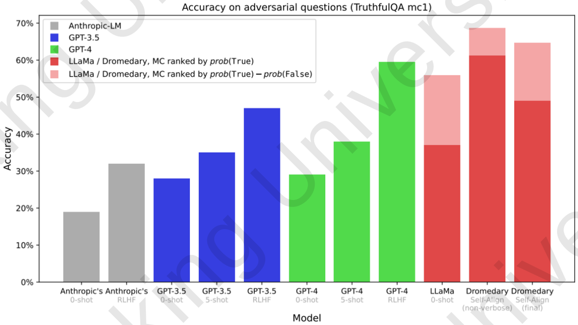

Fig. 17. Multiple Choice (MC) accuracy on TruthfulQA. In our evaluation, the multiple choices are ranked by asking the model if each choice is True or False. Other results are taken from OpenAI. It is not publicly revealed how Anthropic-LM, GPT-3.5-turbo, and GPT-4 rank each answer candidate.

Summary and Outlook

In this lecture, we covered the fundamentals and recent advances of alignment:

- Using reward models to align.
- Self-Alignment

In the next lecture, we will introduce:

- Advantage-Induced Policy Alignment
- Hindsight Instruction Relabeling

Thanks!

References I

[ABC+21] Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al.
A general language assistant as a laboratory for alignment.
arXiv preprint arXiv:2112.00861, 2021.

[BHP+22] Samuel R Bowman, Jeeyoon Hyun, Ethan Perez, Edwin Chen, Craig Pettit, Scott Heiner, Kamile Lukosuite, Amanda Askell, Andy Jones, Anna Chen, et al.
Measuring progress on scalable oversight for large language models.
arXiv preprint arXiv:2211.03540, 2022.

[BKK+22] 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, 2022.

[BN08] Leon Barrett and Srini Narayanan.
Learning all optimal policies with multiple criteria.
In *Proceedings of the 25th international conference on Machine learning*, pages 41–47, 2008.

[DXG+23] Hanze Dong, Wei Xiong, Deepanshu Goyal, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang.
Raft: Reward ranked finetuning for generative foundation model alignment.
arXiv preprint arXiv:2304.06767, 2023.

[FDRC20] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin.
Linear mode connectivity and the lottery ticket hypothesis.
In *International Conference on Machine Learning*, pages 3259–3269. PMLR, 2020.

References II

[KR16] Yoon Kim and Alexander M Rush.
Sequence-level knowledge distillation.
arXiv preprint arXiv:1606.07947, 2016.

[NSZ20] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang.
What is being transferred in transfer learning?
Advances in neural information processing systems, 33:512–523, 2020.

[OWJ+22] 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.
Advances in Neural Information Processing Systems, 35:27730–27744, 2022.

[RAB+22] 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.

[RCS+23] Alexandre Rame, Guillaume Couairon, Mustafa Shukor, Corentin Dancette, Jean-Baptiste Gaya, Laure Soulier, and Matthieu Cord.
Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards.
arXiv preprint arXiv:2306.04488, 2023.

References III

[SSZ+23] Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan.
Principle-driven self-alignment of language models from scratch with minimal human supervision.
arXiv preprint arXiv:2305.03047, 2023.

[VYDB08] Peter Vamplew, John Yearwood, Richard Dazeley, and Adam Berry.
On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts.
In *AI 2008: Advances in Artificial Intelligence: 21st Australasian Joint Conference on Artificial Intelligence Auckland, New Zealand, December 1-5, 2008. Proceedings 21*, pages 372–378. Springer, 2008.

[YYT+23] 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.