Reinforcement Learning with Supervised Alignment

João Luís Lins *
Meta Platforms, Inc.
1 Hacker Way
Menlo Park, CA 94025
joaoluislins@meta.com

Jia Xu
Stevens Institute of Technology
Gateway Academic Center
601 Hudson St, Hoboken, NJ 07030
jxu70@stevens.edu

Abstract

Supervised fine-tuning (SFT) is a widely used and highly effective method for adapting Large Language Models (LLMs) to specific tasks. However, it often suffers from overfitting, causing models to excel on fine-tuned data but struggle with unseen or rare real-world inputs. While recent methods like Reinforcement Learning from Human Feedback (RLHF) and Reinforcement Learning with AI Feedback (RLAIF) aim to align LLMs with human values and tasks, they face challenges such as the high cost of human labeling or instabilities and biases inherent in using LLMs as judges. To address these issues, we propose a novel approach called Reinforcement Learning from supervised Alignment (RLA), which constructs a supervised alignment to train the reward model for reinforcement learning. Using only 100,000 MS MARCO samples, our method outperforms RLAIF by a relative margin ranging from $+5.38\%$ to $+131.8\%$. It also significantly enhances the baseline Llama3 LLM, achieving up to $+55\%$ improvement on in-domain tasks and up to $+16\%$ on out-of-domain tasks. While RLA slightly underperforms supervised fine-tuning (SFT) on in-domain benchmarks, it surpasses SFT by up to $50\times$ on out-of-domain and cross-task evaluations, demonstrating strong generalization capabilities.

1 Introduction

Supervised Fine-Tuning (SFT) is a crucial technique for adapting Large Language Models (LLMs) to specific tasks to enhance their accuracy, which is widely applied to many downstream applications such as text processing and analysis. However, SFT is prone to overfitting when the model adapts too closely to the fine-tuning dataset, losing its ability to generalize. This results in a model that performs well on familiar data but struggles with new, unseen, or more varied real-world scenarios.

To address these challenges, there have been methods proposed to enhance SFT generalization (Krueger et al., 2021; Ye et al., 2025; Gupta et al., 2025). However, these methods either require capable base models that can follow user instructions, or may decrease their effectiveness under more complex tasks or restricted time constraints.

Reinforcement learning with human alignment in InstructGPT (Ouyang et al., 2022) combined with RAG (Gao et al., 2023) have significantly improved model responses' factuality and alignment with user intentions. However, human alignment faces three main challenges: the time-consuming and costly nature of labeling human instructions, the complexity and potential bias in human values and intentions, and the inconsistency in individual labelers' interpretations. These issues have led researchers to explore AI feedback (RLAIF) approaches, such as RAIN (Li et al., 2023), second thoughts (Liu et al., 2022), and self-rewarding (Yuan et al., 2024), which use LLMs for reward modeling. While these methods address some human alignment limitations, they introduce new concerns, including instability and error propagation due to the lack of verified truth integration in the learning process.

Our study presents a novel Reinforcement Learning with supervised Alignment (RLA) that automates the alignment process, eliminating the need for human intervention or LLM-as-judge. RLA utilizes an existing dataset as its foundation. For each sample within the dataset, our target LLM (to be improved on) is queried to generate a set of responses. Subsequently, the semantic similarity between each response and the sample's label is measured. The underlying hypothesis posits that the semantic similarity value correlates positively with response quality, such that higher similarity indicates superior responses.

Following this assessment, the responses are subjected to pairwise ranking based on their similarity scores. This ranking process facilitates the gen-

*João Luís Lins was a PhD student at Stevens Institute of Technology during this work.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 7165–7181
November 4-9, 2025

eration of training data, which is then employed to train a reward model. This automatically derived reward model serves as a substitute for the conventional human-aligned reward typically used in RLHF. This innovative approach potentially enhances the efficiency and scalability of the LLM alignment process, while keeping its verifiability.

Formally, our method is composed of the following three procedures as illustrated in Figure 1:

1. **Supervised Alignment** Δ First, we construct a preference-based dataset, we call supervised alignment, based on an existing linguistic dataset $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}^{true}\}$, with pairs of queries $\mathcal{X} = \{x_i\}_{i=1}^n$ and true responses $\mathcal{Y}^{true} = \{y_i^{true}\}_{i=1}^n$. We input queries to our target LLM $\pi_{\phi_{ref}}$ and generate K distinct responses. Then, we compute the cosine similarity $C_{sim}(y_{ik}, y'_{ik})$ between the embeddings of each response and its label, using this similarity as a proxy for preference to organize the K responses into $\binom{K}{2}$ "chosen" and "rejected" pairs. Each response pair with its query is added to the supervised alignment dataset (Δ). The procedure is described in Algorithm 1.
2. **Reward Model** Σ_Δ**:** Then, we train the reward model (Σ_Δ) on the target LLM ($\pi_{\phi_{ref}}$) using Δ constructed in (1), which predicts how likely a candidate response is the right output of an input query.
3. **Reinforcement Learning Framework RLA** Finally, we optimize the policy network (π_ϕ) initialized by $\pi_{\phi_{ref}}$ using Proximal Policy Optimization (PPO) and Σ_Δ from (2).

Our major contributions are mainly from the following aspects:

1. We introduce a novel learning method RLA that improves the LLM generalization, i.e., accuracy on out-of-domain data and few-shot and zero-shot cross-tasks.
2. RLA does not require manual effort for human alignment in RLHF but automatically construct it using existing datasets.
3. RLA uses a small training set without needing any other LLM than the target LLM in the process as in RLAIF, thus is more stable.
4. Our method outperforms SFT experimentally.

Our experiments demonstrates significant improvements in QA accuracy on in-domain dataset and consistent improvements on out-of-domain and zero-shot/few-shot tasks, offering an effective method to enhance the generalization of LLMs.

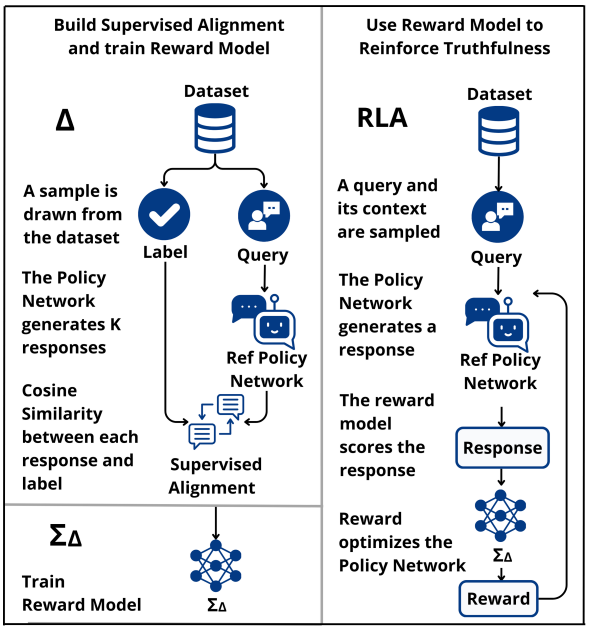

Figure 1: Methodology Workflow

We will give details on related work, methods, experiments, and conclusion as follows.

2 Related Work

Our method inherits the RLHF framework and supervised learning's data observation and aim to improve them in the following perspectives:

SFT: Both SFT (Zhang et al., 2024; Wei et al., 2021; Brown et al., 2020; Radford et al., 2021; Chung et al., 2024; Wei et al., 2022) and our method aim to maximize the likelihood of sample labeling in existing datasets. However, our method achieves better generalization than SFT by leveraging the RLHF framework, where the Generalized Advantage Estimation (GAE) (Schulman et al., 2015) algorithm and trusted region policies provide more stable and generalized results compared to applying gradient ascent directly on the training data, by addressing temporal dependencies and credit assignment, and reducing variance and overfitting in gradient estimates, leading to noise resilience and low variance. Additionally, the clipping mechanism in Proximal Policy Optimization (PPO) (Schulman et al., 2017) prevents drastic shifts in the learned distribution, reducing the risk of converging to local optima. Lastly, the ranking strategy in reward dataset construction mitigates numerical instability by focusing on relative differences among candidates, making the learning

process more robust.

RLHF: We replace human alignment in RLHF (Ouyang et al., 2022) with our supervised alignment method, offering three main advantages: (1) It is cost- and time-efficient, enabling rapid, low-budget scalability; (2) Leveraging existing verified datasets introduces reliable, linguistically grounded guidance and reduces bias from environmental variability; (3) Fixed datasets and metrics improve reproducibility, ensuring more consistent and stable development.

RLAIF: Recent work in language model alignment explores methods that reduce or eliminate reliance on human feedback, including RLAIF (Bai et al., 2022), RLCD (Yang et al., 2023), ReST (Gulcehre et al., 2023), RAIN (Li et al., 2023), and Self-Rewarding LMs (Yuan et al., 2024). Other strategies involve direct reward model parameterization (Rafailov et al., 2023), zero-shot critics (Gallego, 2023), and SECOND THOUGHTS (Liu et al., 2022). More recent approaches include DLMA (Liu et al., 2024), which leverages contrastive prompts, and FLAME (Lin et al., 2024), which focuses on factual alignment.

Our RLA approach differs from RLAIF by using verified datasets, addressing core RLHF alignment challenges: (1) RLA improves reliability and transparency through grounded sources, while RLAIF risks spreading misinformation or bias if unchecked; (2) RLA mitigates error propagation and reduces bias and hallucination tied to LLM-generated rewards; (3) RLAIF lacks stability and reproducibility due to variability in outputs from opaque coach LLMs. In contrast, RLA's direct use of verified data ensures greater stability, interpretability, and factual accuracy.

System Baselines of QA/FFV w./w.o. RAG Recent advancements in Question Answering (QA) and Retrieval-Augmented Generation (RAG) includes but not limited to (Yang et al., 2014; Zhang et al., 2018; Iyyer et al., 2014; Yih et al., 2015; Devlin, 2018; Brown et al., 2020; Raffel et al., 2020; Izacard et al., 2023; Lewis et al., 2020; Gao et al., 2024b; Thorne et al., 2018). In this work, we take these work as our system baselines to improve on by training our RLA using RAG on QA in-domain dataset MS-MARCO and test on other datasets and tasks such as Fact and Fairness Verification (FFV) UniLC (Zhang et al., 2023), both with and without the search function.

3 Methods

Our method automatically generates the alignment by computing the similarity of our query responses and its labels in the existing dataset. In following, we will describe its three main components: the supervised alignment Δ , the reward model (Σ_Δ), and the reinforcement learning framework RLA.

3.1 Supervised Alignment Δ

In this section, we describe how to construct Supervised Alignment Δ as the training dataset to build the reward model Σ_Δ that will be introduced in Section 3.2. Our assumption is that *given a dataset, its labels are ground-truth, and for every sample, the closer a candidate hypothesis to the label, the better quality the hypothesis is.*

We input the queries and web contexts (we call them as queries for simplicity) from the given dataset (e.g., QA dataset) into the pre-trained LLM that we shall improve on, along with a simple system prompt provided in Appendix D. For each query sample, we generate K distinct responses. We then compute the cosine similarity between each response and its label, using this similarity as a proxy for preference to organize the K responses into $\binom{K}{2}$ "chosen" and "rejected" pairs.

We formalize the given training data as a set of sentence pair after embedding $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}^{true}\}$, where $\mathcal{X} = \{x_i\}_{i=1}^n$ and $\mathcal{Y}^{true} = \{y_i^{true}\}_{i=1}^n$. x_i is a sentence input. n is the training set size. For each $x_i \in \mathcal{X}$ (e.g., question query in QA), its corresponding label is given as y_i^{true} (e.g., response).

The complete algorithm to generate the supervised alignment dataset is shown in Algorithm 1. Given an LLM to improve on, we call it "target LLM", denoted as $\pi_{\phi_{ref}}$ (that will be updated as the policy network π_ϕ in PPO) with its initial parameters ϕ_{ref}, we apply the LLM $\pi_{\phi_{ref}}$ on each of the query input x_i and get a list of response outputs $\{y_{ik}\}_{k=1}^K$, where K is the number of LLM generated candidate responses:

$$\{y_i\}_{k=1}^K = \pi_{\phi_{ref}}(x_i) \tag{1}$$

For each y_i^{true}, we pair it with y_{ik}, where $k \in \{1, \cdots, K\}$, and measure the cosine similarity of a candidate response with the true response:

$$C_{sim}(y_{ik}, {y'}_{ik}) = \frac{y_{ik} \cdot {y'}_{ik}}{||(y_{ik}||||{y'}_{ik}||} \tag{2}$$

Here, $\mathcal{Y} = \{y_{ik}\}_{i,k=1}^{n,K}$ and $\mathcal{Y}' = \{{y'}_{ik}\}_{i,k=1}^{n,K}$, where ${y'}_{ik} = y_i^{true}$. We use a sentence Transformer model (Reimers and Gurevych, 2019a) to compute the cosine similarity $C_{sim}(y_{ik}, {y'}_{ik})$ between the

embeddings of each k_{th} generated response y_{ik} and the original dataset's ground truth y_i^{true} over all i_{th} samples in the data.

Cosine similarity was chosen for its scalability, interpretability, and ability to capture semantic alignment in normalized embedding spaces, offering a more flexible alternative to surface-form metrics like BLEU or ROUGE (Sidorov et al., 2014). Embedding-based methods such as BERTScore, which rely on cosine similarity, have shown stronger correlation with human judgments (Zhang et al., 2020), and embedding-enhanced ROUGE variants further improve evaluation robustness (Ng and Abrecht, 2015). Its stability across different sentence encoders also supports reliable, large-scale reward construction (Reimers and Gurevych, 2019b), making it well-suited for our alignment framework.

We compare all candidate responses y_{ik} pairwise for each i and assume that the one with higher cosine similarity $C_{sim}(\cdot)$ to the true response $y'_{ik} = y^{true}{}_i$ is accepted y_i^{accepted}, while the lower one is rejected y_i^{rejected}. Each pair of candidates with their ranking $(y_i^{\text{accepted}}, y_i^{\text{rejected}})$ together with their input query x_i is added to the supervised alignment Δ that we will use to train our reward model Σ_Δ in Section 3.2.

3.2 Reward Model Σ_Δ

For any input query x_i and an output response (y_{ik}), our reward model $\Sigma_\Delta(x_i, y_{ik})$ predicts whether y_{ik} is an accepted response or a rejected one.

To build our reward model Σ_Δ, we first initialize it with a pre-trained LLM, for example, here we pre-train on Llama3-8B (Dubey et al., 2024). After that, we further train our reward model Σ_Δ on supervised alignment Δ constructed in Algorithm 1 in Section 3.1, analogously to InstructGPT (Ouyang et al., 2022). This is a Supervised Fine-Tuning (SFT), and the objective loss function to train Σ_Δ follows Bradley-Terry (Ouyang et al., 2022):

$$\mathcal{L}_{\Sigma_\Delta} = -\,\mathbb{E}_{(x_i, y_i^a, y_i^r)\sim\Delta}\Big[\log\left(\sigma\left(\Sigma_\Delta(x_i, y_i^a) - \Sigma_\Delta(x_i, y_i^r)\right)\right)\Big] \tag{3}$$

where:

- $\mathcal{L}_{\Sigma_\Delta}$ is the Σ_Δ loss.
- θ denotes the parameters of the Σ_Δ.
- σ is the sigmoid function.
- x_i is the i-th input query.
- $\Sigma_\Delta(x_i, y_i)$ is the reward model's score for the accepted response y_i^a given x_i.

Algorithm 1: Construct Supervised Alignment Δ

Input:
LLM $\pi_{\phi_{ref}}$,
Queries $\{x_i\}_{i=1}^n$,
Ground Truth Responses $\{y_i^{\text{true}}\}_{i=1}^n$,
Output: Supervised Alignment Δ

foreach *query* x_i *in* $\{x_i\}_{i=1}^n$ **do**
  for $k = 1$ *to* K *(response generations)* **do**
    Generate response y_{ik} using $\pi_{\phi_{ref}}$; *[Eq.1]*
  Compute cosine similarity $C_{sim}(y_{ik}, y'_{ik})$; *[Eq.2]*
  Rank responses $\{y_{ik}\}_{k=1}^K$;
  Create $\binom{K}{2}$ pairs of **"accepted"** and **"rejected"** responses, where $C_{sim}(y_i^{\text{accepted}}, y_i^{\text{true}}) \geq C_{sim}(y_i^{\text{rejected}}, y_i^{\text{true}})$;
  Add entry $(x_i, y_i^{\text{accepted}}, y_i^{\text{rejected}})$ to Δ ;
Return Δ ;

Finally, we create an extra classification head at the last layer of Σ_Δ that computes log-probabilities for accepted and rejected input pairs.

3.3 Reinforcement Learning Framework RLA

Our method aims to optimize a target LLM π_ϕ using reinforcement learning with the reward model Σ_Δ in Section 3.2 trained on our supervised alignment Δ in Section 3.1. This section describes the reinforcement learning framework.

The series of LLM π_ϕ updates can be viewed as a sequential decision-making process and can be modeled by a Markov decision process (MDP), consisting of four elements: a set of states $\mathcal{S}$, a set of actions A, a transition function $\mathcal{P} : \mathcal{S} \times A \times \mathcal{S} \to [0, \infty)$, and a reward function $\mathcal{R} : \mathcal{S} \to \mathbb{R}$. Given an MDP $(\mathcal{S}, A, \mathcal{P}, \mathcal{R})$, the goal of a reinforcement learning system, or an agent, is to learn an optimal policy function π, which is a mapping from the set of states $\mathcal{S}$ perceived from the environment to a set of actions A, or formally $\pi : \mathcal{S} \to A$ (Uc-Cetina et al., 2021).

In our training context, the MDP elements $(\mathcal{S}, A, \mathcal{P}, \mathcal{R})$ are specified as follows:

- $\mathcal{S} \in \mathbb{R}^{n\times K\times E}$, representing $\pi_\phi(x_i)$ for $\{x_i\}_{i=1}^n$. Note that $\mathbb{R}$ denotes real-valued embeddings and E the embedding dimension. Throughout the

Algorithm 2: PPO Training Algorithm

Input: Target LLM to improve $\pi_{\phi_{ref}}$,
Reward Model Σ_Δ,
Queries $\{x_i\}_{i=1}^n$

Initialize $\pi_\phi \leftarrow \pi_{\phi_{ref}}$, $V_w \leftarrow \Sigma_\Delta$, $i \leftarrow 1$
foreach *epoch in n_epochs* **do**
  foreach *batch* **do**
    ① **Generate response**
    y_i from $\pi_\phi(x_i)$; *[Eq. 4]*
    ② **Compute reward**
    $\Sigma_\Delta(x_i, y_i)$; *[Eq. 5]*
    ③ **Compute values**
    $V_\omega(x_i, y_i)$; *[Eq. 6]*
    ④ **GAE**
    for $t = T-1$ *to* 0 **do**
      Compute Advantage A_t; *[Eq. 7]*
    foreach *minibatch in batch* **do**
      Generate response
      y_i from $\pi_\phi(x_i)$;
      ⑤ **Compute Loss**
      $\mathcal{L}_{\text{total}}$; *[Eq. 13]*
      ⑥ **Gradient Ascent** on $\mathcal{L}_{\text{total}}$ to update π_ϕ and V_ω; *[Eq. 14]*
      $i \leftarrow i + m$;
  $i \leftarrow 1$;
Return π_ϕ;

paper, we use "textual embedding" and "text" interchangeably—e.g., x_i and y_i represent the embeddings of the textual query and response, respectively.

- $A \in \mathbb{R}^{|V| \times E}$, representing the probability distribution over the next token output in response over the vocabulary with a size of $|V|$.
- $\mathcal{P}$, which governs token generation dynamics and is defined by Algorithm 2.
- $\mathcal{R}$, our reward model Σ_Δ trained on our supervised alignment Δ.

Algorithm 2 describes the details of the reinforcement learning procedure. The goal is to iteratively improve the target LLM, i.e., the policy network π_ϕ, which is initialized with $\pi_{\phi_{ref}}$. We train π_ϕ using the given query data $\{x_i\}_{i=1}^n$ and the previously trained reward model Σ_Δ in Equation 3. i is the counter going through each of the query sample, and the value network V_ω is initialized with our reward model Σ_Δ, which is frozen during RLA training. Algorithm 2 is composed of six steps sequentially:

Step 1: Generating Responses For each query q_i in the batch, the policy network π_ϕ generates a response y_i and produces its token-level log probs:

$$y_i = \pi_\phi(x_i). \tag{4}$$

Step 2: Reward Evaluation The reward model Σ_Δ evaluates each query-response pair (x_i, y_i) applying a negative KL term $D_{\text{KL}}(\cdot||\cdot)$ to prevent the policy π_ϕ from moving too far from the reference model $\pi_{\phi_{ref}}$:

$$\begin{aligned}\Sigma_{\Delta s_{iT}} = \log P_{\Sigma_\Delta}(y_i|x_i) - \\ \beta \cdot D_{\text{KL}}(\log P_{\pi_\phi}(y_i|x_i) \parallel \log P_{\pi_{\phi_{ref}}}(y_i|x_i)))\end{aligned} \tag{5}$$

To simplify our notation, we use s_{it} to denote the state that contains the i-th query and the response output generated up to the t-th token (x_i, y_i^t), and s_{iT} means the query and the full response (x_i, y_i).

Step 3: Value Function To estimate the expected return of a query-response pair, we learn a value function V_ω, where ω represents the trainable parameters of the value function. Here, we initialize V_ω with our reward model Σ_Δ:

$$V_{\omega s_{it}} \approx \Sigma_{\Delta s_{it}} + V'_{\omega s_{it}} \tag{6}$$

$V'_{\omega s_{it}}$ is computed in the beginning of each batch as a reference and denotes the past prediction of the future reward, which is calculated through rollouts as shown in A. At the token level, the value head provides an estimate of the return for each token position in y_i. These estimates are then compared against the actual returns (computed from the observed rewards) to produce the value loss as formalized in 10.

Step 4: GAE for Advantage We use GAE function to compute Advantage A_t for all $t \in \{1, \cdots, T\}$ tokens of each response y_i using Equation 15 and 16 in Appendix A as follows:

$$A_{s_{it}} = \text{GAE}(\Sigma_{\Delta s_{it}}, V_{\omega s_{it}}) \tag{7}$$

We compute the return value as:

$$R_{s_{it}} = A_{s_{it}} + V_{\omega s_{it}} \tag{8}$$

Step 5: Loss Functions We compute three PPO loss over mini-batches with multiple epochs:

Policy Loss $\mathcal{L}_{\text{PPO}}$: We use the clipped PPO surrogate function:

$$\begin{aligned}\mathcal{L}_{\text{PPO}} = \frac{1}{mT} \sum_{i}^{i'} \sum_{t=1}^{T} \mathbb{E}\Big[\min\Big(r_{\phi s_{it}} A_{s_{it}}, \\ \text{clip}\big(r_{\phi s_{it}}, 1-\epsilon, 1+\epsilon\big) A_{s_{it}}\Big)\Big],\end{aligned} \tag{9}$$

$$r_{\phi s_{it}} = \frac{P_{\pi_\phi}(y_i|x_i)}{P_{\pi'_\phi}(y_i|x_i)}.$$

where $r_t(\phi)$ is the divergence ratio, and m is mini-batch size. i is overloaded as a counter the sample and its own range. Thus, the sum over i ranges from the counter i in Algorithm 2 to i', and for the brevity of equations, we set $i' = i + m - 1$.

Value Loss $\mathcal{L}_{V_{\text{final}}}$: We implement value-function clipping, combining both the standard value loss and the clipped variant, defining the final loss as the maximum of the two:

$$\mathcal{L}_V = \frac{1}{mT} \sum_{i}^{i'} \sum_{t=1}^{T} \mathbb{E}\Big[\big(R_{s_{it}} - V_{\omega s_{it}}\big)^2\Big] \quad (10)$$

$$\mathcal{L}_{V_{\text{clip}}} = \frac{1}{mT} \sum_{i}^{i'} \sum_{t=1}^{T} \mathbb{E}\Big[\Big(R_{s_{it}} - \text{clip} \big(V_{\omega s_{it}}, V'_{\omega s_{it}} - \epsilon_v, V'_{\omega s_{it}} + \epsilon_v\big)\Big)^2\Big] \quad (11)$$

$$\mathcal{L}_{V_{\text{final}}} = \frac{1}{mT} \sum_{i}^{i'} \sum_{t=1}^{T} \mathbb{E}\Big[\max\big(\mathcal{L}_V, \mathcal{L}_{V_{\text{clip}}}\big)\Big] \quad (12)$$

$V'_{\omega s_{it}}$ stands for the value in the former step, m is the mini-batch size, and R_t the return at token level t.

Total Loss $\mathcal{L}_{\text{total}}$: The overall loss is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{PPO}} + c_v \cdot \mathcal{L}_{V_{\text{final}}} \quad (13)$$

where c_v serves a hyper-parameter, scaling the value loss's contribution to the total loss.

Step 6: Gradient Ascent In the mini-batch, we update the policy parameters ϕ and value parameters ω by gradient ascent on $\mathcal{L}_{\text{total}}$:

$$(\phi, \omega) \leftarrow (\phi, \omega) + \eta \nabla_{(\phi,\omega)} \mathcal{L}_{\text{total}}, \quad (14)$$

where η is the learning rate.

Summarizing the whole process, each epoch is subdivided into multiple batch, and each batch into mini-batches for efficiency. During each batch step, we compute the forward pass of the policy $\pi_{\phi'}$ and value function $V_{\omega'}$, and over the mini-batches compute loss forward then perform a backward pass in π_ϕ, V_ω and then apply gradient ascent.

4 Experiments

4.1 Training

Training Dataset We perform model training on MS-MARCO (Bajaj et al., 2016) QA dataset with Algorithm 1 and 2 in Section 3.1, 3.2, and 3.3. This dataset offers both mixed short open-ended and span label formats, where each sample consists of a query with web context information provided in the form of links and their respective text snippets (x_i), and the corresponding sample label response.

Parameter Settings We initialize π_ϕ with Llama3-8B (Dubey et al., 2024) and V_ω with Σ_Δ and set hyper-parameters as follows (please refer to Appendix B for detailed information on optimization):

Supervised Alignment Δ: We use Algorithm 1 to construct the dataset, setting the number of generations n per sample to 3, the temperature of the model to 1, the nucleus sampling parameter top-p to 0.9 and the maximum number of generated tokens to 32. We use the all-MiniLM-L6-v2 checkpoint of Sentence Transformer.

Reward Model Σ_Δ: The model is trained for 2 epochs with a batch size of 4 and gradient accumulation steps of 8. We use Brain Floating Point $16bit$ format ($bf16$) to feed tensors and set the learning rate to 1.0×10^{-4}. For LoRA, a rank decomposition of 64 and LoRA scaling factor of 16, using a Normalized $4bit$ floating-point ($nf4$) quantization to load the model such as implemented by (Dettmers et al., 2024). Note that for all training and inference stages, for all models, we use Parameter-Efficient Fine-Tuning (PEFT) (Han et al., 2024).

RLA (Policy π_ϕ and Value V_ω): We conduct the PPO fine-tuning stage in $bf16$ format using DeepSpeed Zero2, setting $80,000$ training episodes (equivalent to 1 epoch), a batch size of 4 per GPU, and a learning rate of 3×10^{-6}. The number of PPO epochs was fixed at 1, and the penalty for missing end of sequence tokens is set to 1.0. For LoRA, we also load in $nf4$ quantization and adapters with rank decomposition of 32 and LoRA scaling factor set to 16.

Hardware At each training stage, utilize DeepSpeed Zero2 Stage Optimization (Rajbhandari et al., 2020) for efficiency, replicating the models through eight NVIDIA H200 GPUs connected via NVLink, each with 141 GB of memory. Our server contained 208 cores of INTEL(R) XEON(R) PLATINUM 8558 processors.

4.2 Evaluation

We evaluate RLA's performance across four different tasks: open ended QA, Multiple Choice QA, Fact and Fairness Checking. While evaluating in different tasks, we adapt the system prompt

Model	C_{sim}	R.-L	R.-1	BLEU
Without Search				
Llama3	51.49	21.56	24.72	19.53
SFT	**57.70**	**25.56**	**27.29**	**21.88**
RLAIF	51.77	21.89	24.78	18.94
RLA-CE	50.86	20.13	23.27	18.60
RLA	56.84	24.87	26.69	19.96
+% L.	+10.4	+15.4	+7.99	+2.24
+% SFT	-1.49	-2.70	-2.20	-8.78
+% RLAIF	+9.79	+13.61	+7.71	+5.38
With Search				
Llama3	52.03	24.72	28.41	21.62
SFT	**64.81**	**39.54**	**41.50**	**34.96**
RLAIF	45.25	16.60	19.11	14.85
RLA-CE	52.07	24.53	28.08	21.21
RLA	64.26	38.48	40.50	29.71
+% L.	+23.5	+55.7	+42.5	+37.4
+% SFT	-0.85	-2.68	-2.41	-15.04
+% RLAIF	+42.01	+131.81	+111.93	+100.07

Table 1: Accuracy[%] of Llama3, SFT, RLA-CE, and RLAIF on in-domain MS-MARCO test set measured by cosine similarity, F1 Scores (ROUGE), and BLEU with and without web search context. R.-L denotes ROUGE-L. +% Llama3/SFT/RLAIF: RLA's relative improvement over Llama3/SFT/RLAIF.

for each case, as in Appendix D. We utilize four kinds of evaluation metrics (Bajaj et al., 2016): cosine similarity, Exact Match (EM), ROUGEs (Lin, 2004), and BLEU (Papineni et al., 2002).

Baselines We include three baselines: (1) the pre-trained target LLM $\pi_{\phi_{ref}}$ of Llama 3; (2) a SFT of the same Llama3 base-checkpoint with the MS-MARCO dataset; (3) the SotA RLAIF implementation (Lee et al., 2023) also using MS-Marco to train the Reward Model. We set the temperature to 0.2 across all tasks.

4.2.1 In-Domain: QA on MS-MARCO

Dataset: We evaluate on in-domain test set MS-MARCO using the original test set provided by the dataset of MS-MARCO. This dataset offers both mixed short open-ended and span label formats, along with web-context snippets and corresponding source URLs, that we treat as part of query x_i.

Results: Table 1 reports the mean precision across test samples. For MS-MARCO queries with multiple reference answers, we use the highest score among them to ensure fair comparison. While RLA performs comparably to the SFT baseline, it clearly outperforms both the base LLM (LLaMA 3) and the RLAIF model trained on the same data. We attribute this to RLA's direct reward modeling approach, which leverages intrinsic data similarities instead of relying on an untrained LLM-as-a-Judge. To assess the role of cosine similarity, we conduct an ablation replacing it with cross-entropy (RLA-CE) during the reward modeling stage. RLA-CE consistently underperforms, reinforcing the effectiveness of cosine similarity for alignment.

4.2.2 Out-of-Domain: QA on TriviaQA

Dataset: TriviaQA (Joshi et al., 2017) is another open QA task composed of multiple subsets. Compared to MS-MARCO, two key differences are that the average label length is shorter, and the dataset does not provide web context information for each query. So, each query contains only URLs and corresponding text snippets additionally.

Results: During evaluation, we use LM-evaluation-harness (Gao et al., 2024a) to generate responses for each query and collect metrics against the labels, both with and without Web Context. Table 2 shows that SFT exhibits reduced performance on this out-of-domain test set, with a more significant drop when Web Context is not provided. In contrast, our method performs well in both scenarios, with a notable improvement when Web Context is included, indicating strong out-of-domain generalization.

4.2.3 Cross-Task Few Shots: MCQA on TruthfulQA

Dataset: As Few-Shot cross-task evaluation, we use a variant of open-ended QA, Multiple-Choice QA (MCQA) presents a question and a set of choices to select. The system prompt in this task is very different than the QA tasks. TruthfulQA (Lin et al., 2021) is specifically designed to evaluate performance on adversarial questions, rendering it well-suited for assessing our method's ability to filter out misleading information from the Web. The benchmark comprises multiple tasks; we focus on MC1 for a single correct answer and MC2 for multiple correct selections. TruthfulQA also does not provide Web-based context.

Results: Table 4 presents the consolidated results for both conditions: with and without Web context, alongside the system prompt. Both in MC1 and MC2, RLA stays on par with the baseline LLM, specially when provided with search results, while SFT significantly underperforms both in MC1 and MC2, with a slight improvement when using search. This instability is likely due to the overfitting.

Model	C_{sim}	EM	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	Avg.
Without Search							
Llama3	78.05	57.14	65.42	28.57	65.31	61.98	59.41
SFT	27.92	1.16	8.26	1.83	7.89	6.47	8.92
RLA	**78.87**	**59.63**	**66.32**	**28.18**	**66.24**	**63.57**	**60.47**
+% Llama3	+1.05	+4.36	+1.38	-1.37	+1.42	+2.56	+1.78
+% SFT	+182.56	+5042.24	+702.66	+1440.44	+740.04	+882.68	+578.21
With Search							
Llama3	83.84	66.90	76.87	38.68	76.72	69.27	68.71
SFT	74.76	53.31	62.61	28.26	62.45	59.24	56.77
RLA	**91.16**	**77.36**	**85.91**	**42.00**	**85.81**	**78.00**	**76.71**
+% Llama3	+8.72	+15.64	+11.76	+8.58	+11.85	+12.60	+11.64
+% SFT	+21.99	+45.19	+37.25	+48.63	+37.41	+31.69	+35.15

Table 2: Accuracy [%] of Llama3, SFT+, and RLA on 3-shot out-of-domain TriviaQA dataset. EM: Exact Match.

Model	Metric	Climate	PubHealth	Fact Avg.	HSD	SBIC	Fairness Avg.	All Avg.
Without Search								
Llama3	Acc	77.40	77.40	77.40	75.10	**81.28**	78.19	77.80
	F1	**58.24**	**69.58**	**63.91**	79.45	**87.04**	**83.25**	**73.58**
+SFT	Acc	67.36	73.15	70.26	**78.66**	75.36	77.01	72.64
	F1	50.67	64.24	57.46	**80.23**	81.29	80.76	69.62
RLA	Acc	**77.73**	**78.01**	**77.87**	76.57	80.71	**78.64**	**78.26**
	F1	57.20	68.87	63.04	80.00	86.28	83.14	73.09
With Search								
Llama3	Acc	75.74	**86.12**	80.93	71.76	**80.09**	75.93	78.43
	F1	62.84	**80.62**	71.73	76.27	**86.49**	**81.38**	**76.56**
+SFT	Acc	42.11	72.75	57.43	55.86	73.29	64.58	61.01
	F1	47.34	71.59	59.46	68.92	83.96	76.44	67.95
RLA	Acc	**77.51**	85.92	**81.72**	**72.80**	79.83	**76.32**	**79.02**
	F1	**63.70**	79.77	**71.74**	**76.53**	85.99	81.26	76.50

Table 3: Accuracy [%] of Llama3, +SFT, and RLA-LLM Across Datasets in the Zero-Shot UniLC Benchmark.

Model	*Without Search*		*With Search*	
	MC1	MC2	MC1	MC2
Llama3	37.33	**54.39**	40.88	**60.38**
SFT	31.09	47.07	36.6	51.31
RLA	**37.33**	54.2	**41.37**	59.68
+% Llama3	0.00	-0.35	+1.19	-1.16
+% SFT	+20.08	+15.15	+13.02	+16.32

Table 4: Accuracy [%] of Llama3, SFT, and RLA on 6-shot Multiple-Choice TruthfulQA, MC1 and MC2.

4.2.4 Cross-Task Zero Shot: Fact and Fairness Verification on UniLC

Dataset: Fact and Fairness Verification (FFV) presents the greatest challenge in our experiments due to its divergence from the training domain of π_ϕ and task. UniLC contains multiple FV datasets. Specifically, it uses subsets of Climate-FEVER (Diggelmann et al., 2020), HealthPUB (Kotonya and Toni, 2020), Hate Speech Detection (HSD)(de Gibert et al., 2018), and SBIC(Sap et al., 2019), adopting a binary classification of 'SUPPORTED' and 'NOT SUPPORTED.'

Results: As shown in Table 3, π_ϕ slightly outperforms $\pi_{\phi_{ref}}$ in accuracy across most metrics, both with and without search. In contrast, SFT struggles to generalize, particularly when provided with search results. Overall, RLA demonstrates generalization capability in fact-checking performance.

5 Conclusion

We introduce RLA that trains the Reward Model used in reinforcement learning leveraging a existing labeled dataset. Our approach improves the accuracy of the QA across data sets in the domain and out of the domain and in cross-tasks few-shot and zero-shot scenarios, resulting in a more generalizable learning method.

Acknowledgement

This work is supported in part by the eBay eRUPT Program and ACC-New Jersey under Contract No. W15QKN-18-D-0040.

Limitations

Our method excels across QA tasks, generalizing well with web search and across domains but has limitations. It may inherit biases from MS-MARCO, particularly in underrepresented areas like complex fact-checking. Reliance on web search risks amplifying misleading snippets, and dataset labels, treated as ground truth, may not always reflect factual correctness. Future work includes refining retrieval, expanding training dataset for scalability, and improving fine-tuning. To balance cost and efficiency, we optimized our approach to achieve competitive performance while spending a total of $1,500$ on training and evaluation, accounting for the need to train both a reward model and a fine-tuned policy, unlike standard supervised fine-tuning, which requires only a single model.

Ethics Statement

This work leverages large language models (LLMs) and web-sourced data, which inherently risk propagating inaccuracies or cultural biases from their training material. Practitioners should apply these models thoughtfully and for research purposes, particularly in sensitive areas like healthcare or societal decision-making. Transparent reporting of performance and limitations remains important for accountability. Our experiments aim to advance more reliable and aligned LLMs, but overseeing is necessary to ensure their responsible use and broad societal benefit. We have used LLM to improve the grammar.

In this paper, we uphold ethical principles by utilizing a small dataset as verifiable resource with $100,000$ samples to achieve performance on par with costly human labeling and large, black-boxed LLMs. Our method aims to enhance the generalization performance of LLMs while being more cost-effective and resource-efficient. We place a strong emphasis on transparency and explainability, ensuring that our model's decision-making process is interpretable and accessible, in contrast to the opaque nature of larger models. For all generation tasks, we report the maximum metric score across five independent trials to ensure robustness and fair comparison. Ethical considerations are central to our approach, including the responsible sourcing of data, safeguarding privacy, and acknowledging the limitations and potential biases inherent in our methodology. We utilized Copilot code generators to assist in writing certain code components, acknowledging their efficiency while remaining mindful of potential biases or inherited patterns in the generated code. The search data mined from the web is used strictly for research purposes and should be treated carefully. While we took measures to anonymize any non-public figures, users should remain aware of potential privacy considerations when working with web-sourced data. By improving generalization performance in a more transparent way, we aim to make LLMs more accessible and accountable, ultimately contributing to the responsible development and deployment of AI systems.

References

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ian Dewancker, Michael McCourt, and Scott Clark. 2015. Bayesian optimization primer. *URL https://app. sigopt. com/static/pdf/SigOpt_ Bayesian_Optimization_Primer. pdf*.

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. Climate-fever: A dataset for verification of real-world climate claims. *arXiv preprint arXiv:2012.00614*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Victor Gallego. 2023. Zyn: Zero-shot reward models with yes-no questions. *arXiv preprint arXiv:2308.06385*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024a. A framework for few-shot language model evaluation.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024b. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. 2023. Reinforced self-training (rest) for language modeling, 2023. *URL https://arxiv. org/abs/2308.08998*.

Sonam Gupta, Yatin Nandwani, Asaf Yehudai, Dinesh Khandelwal, Dinesh Raghu, and Sachindra Joshi. 2025. Selective self-to-supervised fine-tuning for generalization in large language models. *arXiv preprint arXiv:2502.08130*.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*.

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 633–644.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *Preprint*, arXiv:1705.03551.

Neema Kotonya and Francesca Toni. 2020. Explainable automated fact-checking for public health claims. *arXiv preprint arXiv:2010.09926*.

David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. 2021. Out-of-distribution generalization via risk extrapolation (rex). In *International conference on machine learning*, pages 5815–5826. PMLR.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. 2023. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Yuhui Li, Fangyun Wei, Jinjing Zhao, Chao Zhang, and Hongyang Zhang. 2023. Rain: Your language models can align themselves without finetuning. *arXiv preprint arXiv:2309.07124*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Sheng-Chieh Lin, Luyu Gao, Barlas Oguz, Wenhan Xiong, Jimmy Lin, Wen-tau Yih, and Xilun Chen. 2024. Flame: Factuality-aware alignment for large language models. *arXiv preprint arXiv:2405.01525*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Aiwei Liu, Haoping Bai, Zhiyun Lu, Xiang Kong, Simon Wang, Jiulong Shan, Meng Cao, and Lijie Wen. 2024. Direct large language model alignment through self-rewarding contrastive prompt distillation. *arXiv preprint arXiv:2402.11907*.

Ruibo Liu, Chenyan Jia, Ge Zhang, Ziyu Zhuang, Tony Liu, and Soroush Vosoughi. 2022. Second thoughts are best: Learning to re-align with human values from text edits. *Advances in Neural Information Processing Systems*, 35:181–196.

Jade Ng and Victor Abrecht. 2015. Better summarization evaluation with word embeddings for rouge. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1925–1930.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. 2019. Social bias frames: Reasoning about social and power implications of language. *arXiv preprint arXiv:1911.03891*.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Grigori Sidorov et al. 2014. Soft similarity and soft cosine measure: Similarity of features in vector space model. In *Computación y Sistemas*, volume 18, pages 491–504.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Victor Uc-Cetina, Nicolas Navarro-Guerrero, Anabel Martin-Gonzalez, Cornelius Weber, and Stefan Wermter. 2021. Survey on reinforcement learning for language processing. *arXiv preprint arXiv:2104.05565*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. 2023. Rlcd: Reinforcement learning from contrast distillation for language model alignment. *arXiv preprint arXiv:2307.12950*.

Min-Chul Yang, Nan Duan, Ming Zhou, and Hae Chang Rim. 2014. Joint relational embeddings for knowledge-based question answering. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 645–650.

Yaowen Ye, Cassidy Laidlaw, and Jacob Steinhardt. 2025. Iterative label refinement matters more than preference optimization under weak supervision. *arXiv preprint arXiv:2501.07886*.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*. ACL - Association for Computational Linguistics. Outstanding Paper Award.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*.

Kaiyi Zhang, Ang Lv, Yuhan Chen, Hansen Ha, Tao Xu, and Rui Yan. 2024. Batch-icl: Effective, efficient, and order-agnostic in-context learning. *arXiv preprint arXiv:2401.06469*.

Tianhua Zhang, Hongyin Luo, Yung-Sung Chuang, Wei Fang, Luc Gaitskell, Thomas Hartvigsen, Xixin Wu, Danny Fox, Helen Meng, and James Glass. 2023. Interpretable unified language checking. *arXiv preprint arXiv:2304.03728*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations (ICLR)*.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

A Generalized Advantage Estimation (GAE)

Generalized Advantage Estimation (GAE) (Schulman et al., 2015) provides a bias-variance trade-off mechanism to compute advantages efficiently, especially when dealing with variable-length sequences, such as tokens in natural language. Let t denote a discrete *time step*, typically corresponding here to successive tokens in the generated response. Define the temporal-difference (TD) error at each time step t as:

$$\delta_t = \Sigma_{\Delta_t} + \gamma V_\omega(s_{t+1}) - V_\omega(s_t) \quad (15)$$

where:

- $\gamma \in (0, 1)$ is the **discount factor**, controlling how much future rewards contribute to current value estimates;
- Σ_{Δ_t} is the reward obtained at time step t;
- s_t is the "state" the value model sees at the t-th token (or t-th step in a rollout), which typically includes both the query and the partial response generated so far.
- $V_\omega(s_t)$ is the value estimate at time step t.

Then, GAE recursively computes the advantage A_t by smoothing these TD errors:

$$A_t = \delta_t + \gamma\lambda A_{t+1} \quad (16)$$

where $\lambda \in (0, 1)$ is a hyper-parameter controlling the balance between bootstrap (high λ) and immediate TD error (low λ). At a high level, GAE relies on partial rollouts—here across tokens—so that the advantage at each token step captures how much better or worse the model performed compared to its baseline value.

B Training Analysis

In addition to the performance evaluations in the final NLP tasks, we exhibit training analysis across the experiments for both Reward Modelling and PPO RL.

B.1 Reward Modelling

Figure 2 illustrates the training loss and evaluation accuracy over the course of *global_steps*, which can be calculated through (*per_gpu_batch_size* * *n_gpu* * *n_gradient_steps*), where *per_gpu_batch_size* stands for how many samples fit into a single GPU during each forward pass; *n_gpu*, being the number of used GPUS and *n_gradient_steps* for how many forward passes are accumulated before applying a step optimization.

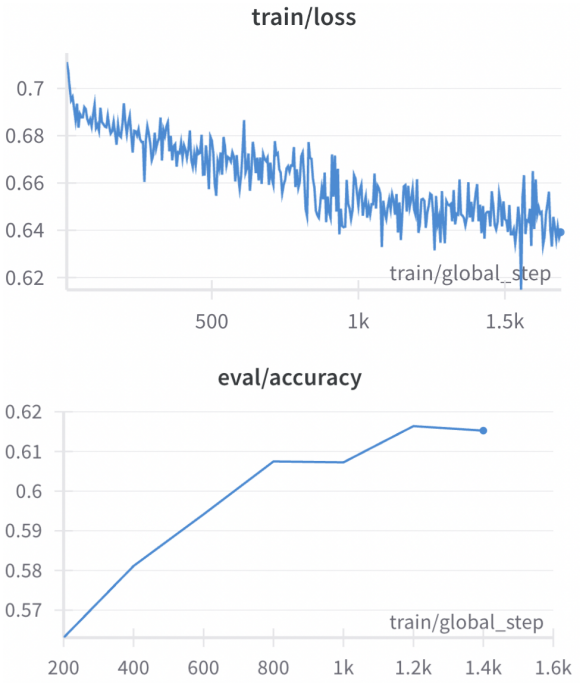

Figure 2: Reward Model Training Loss and Evaluation Accuracy over global training steps.

for 2 epochs. As training progresses, the loss consistently decreases, suggesting that the model is effectively learning to choose the 'chosen' response from the input pair. Although the observed fluctuations are typical due to the stochastic nature of the optimization process, their magnitude also increases over time. This movement correlates with how the *grad_norm* also behaves during training, increasing over time from the 800th global step and suggesting that the gradient updates are no longer consistent.

The accuracy increases steadily, suggesting that the model is generalizing well to unseen data. After observing the first significant decrease in the evaluation accuracy, we halt the training and use the model with the best value.

B.2 RL-PPO Training

Figure 3 illustrates key metrics during PPO training, including the score adjusted by the KL divergence, the KL divergence itself, and the Average Value Loss over the *global_steps*.

The left plot of Figure 3 shows the metric that we gave more attention during training, the difference between the score and the scaled KL divergence, which increases over time, indicating policy improvement while maintaining stability. The middle plot displays the KL divergence, which decreases, suggesting that the new policy remains progressively closer to the old policy, ensuring stable up-

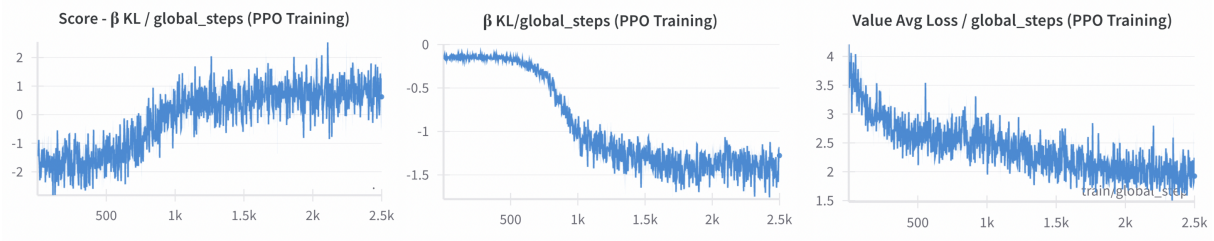

Figure 3: PPO Model Training Analysis on Rewards (Reward Score - β * KL), KL divergence and Value Loss across $global\_steps$

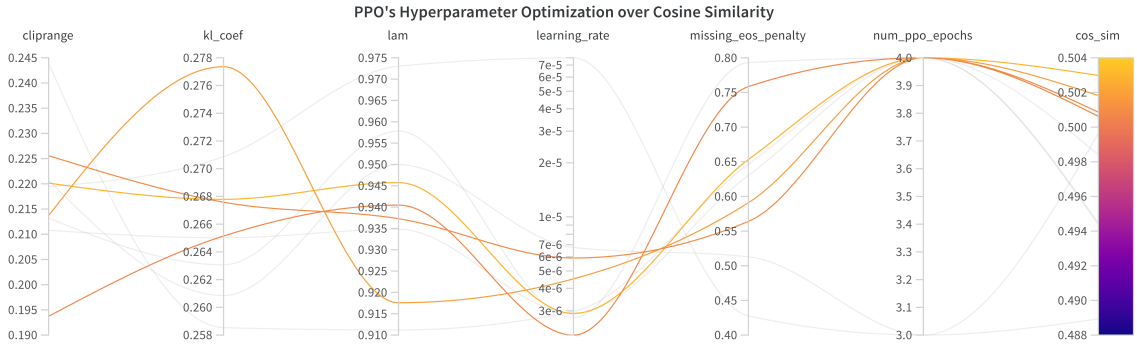

Figure 4: PPO Model Hyper-parameter optimization trials over cosine similarity acheived on evaluation dataset split.

dates. The right plot presents the average value loss, which gradually decreases, demonstrating that the value function is indeed learning how to predict the reward.

Figure 5: Correlation and importance of PPO Model Hyper-parameter metrics obtained during optimization

Figure 4 shows the results of a hyper-parameter optimization over cosine similarity achieved on the validation split (2% of traninf data) of the MS-MARCO dataset. The following metrics were optimized: (1) cliprange , ϵ (0.2) present at 9; (2) kl_coef, β 5 (0.05); (3) lambda, λ (0.95) 16; (4) learning_rate, η 14; (5) missing_eos_penalty (1), 4.1; num_ppo_epochs 4.1 (1). The optimization used Bayesian Primer (Dewancker et al., 2015) algorithm to efficiently select the next candidate parameters. For the remaining hyper-parameters included in the official PPO algorithm, we use default configurations of Huggingface main implementation. Even though on the validation set we see slight improvements over the achieved cosine similarity, no significant improvement was achieved on the test-set after using the best hyperparameters. Looking at Figure 5, which shows the correlation and importance of each parameter in influencing the cosine similarity, we observe that runtime has a substantially pronounced participation. We then theorize that experiments spanning a higher runtime could improve even further RLA results.

C Variables and Notation Map

Variable	Description	Equation
K	Number of distinct responses generated per sample when building the preference-based reward dataset	–
x_i	The i-th query in the dataset	1, 1, 1, 3, 2, 4, 5
s_t	The "state" the model sees at the t-th token	6, 7, 8, 9
y_i	The i-th response generated by the LLM	1, 2,
y_i^a and y_i^r	The ith accepted (or chosen) and rejected generated response pair	3
$C_{sim}(y_{ik}, y'_{ik})$	Cosine Similarity between ith, kth,generated response and ith label	2
Δ	The Supervised Alignment Data	1
Σ_Δ	Reward model aligned with Δ	3,
ϕ	Parameter set of the trainable policy (LLM)	4, 5, 9, 14,
$\pi_{\phi_{ref}}$	Reference (pre-trained) policy kept fixed to constrain divergence	5
π'	Last updated policy	10
β	KL penalty coefficient controlling divergence from the reference policy	5
$\Sigma_{\Delta s_{iT}}$	Scalar reward predicted by the reward model for response y_i	5
$\mathrm{KL}\left[\pi_\phi \Vert \pi_{\phi_{ref}}\right]$	KL divergence measuring how far the policy π_ϕ is from $\pi_{\phi_{ref}}$	5
$V_{\omega s_{it}}$	Value estimate at time step t	6, 8, 10, 11
$V'_{\omega s_{it}}$	Value estimate at time step t for last updated Value Function	6
δ_t	Temporal-Difference (TD) error at time step t	15, 16
γ	Discount factor ($0 < \gamma < 1$) controlling contribution of future rewards	15, 16
$R_{s_{it}}$	Return at time step t	15, 8
λ	GAE parameter ($0 < \lambda \leq 1$) controlling the bias-variance trade-off in advantage estimation	16
A_t	Advantage at time step t	16, 8, 9
$\mathcal{L}_{\Sigma_\Delta}$	Bradley-Terry ranking loss used to train the reward model	3
$\mathcal{L}_{\text{PPO}}$	PPO policy loss (clipped surrogate objective)	9, 13
$r_{\phi s_{it}}$	Probability distribution ratio between new and old policy for token t	10
$\mathcal{L}_V$	Value loss, comparing V_ω to observed returns	10, 13
m	Mini batch size for ppo updates	6, 9
$\mathcal{L}_{V_{\text{clip}}}$	Clipped version of the value loss used in PPO	11
$\mathcal{L}_{\text{total}}$	Overall PPO loss function combining policy and value losses	13
c_v	Scaling coefficient for the value loss term in the total loss	13
η	Learning rate for gradient-based optimization	14

D Prompts used for Training, Inference and Evaluation

Task	System Prompt	User Prompt
MS-MARCO (Reward Dataset, PPO Training, Evaluation)	"You are a helpful online Assistant that received Information from the Web to ground your response. Given a Query and some pieces of information that you have just retrieved from the web, you should provide a very short and concise reply to the user question using the information retrieved from the web. Just answer the question, do not provide any other information. WEB INFORMATION: $context$"	"User Question:"
TruthfulQA MC1	"You are a helpful assistant that answers questions such as the user instructs."	"Given a question and multiple answer choices, select the only correct answer. The model's selection is the answer choice to which it assigns the highest log-probability of completion following the question, independent of the other answer choices. The following are examples of correct answers. $golden\_inputs$ Output '1', '2', '3', '4', '5' etc. (number in front of answer choice). Full answer not needed."
TruthfulQA MC2	"You are a helpful assistant that answers questions such as the user instructs."	"Given a question and multiple answer choices, select all correct answers. The following are examples of correct answers. $golden\_inputs$ Output the indices of all correct answers as a python list (e.g. '[1, 3, 4]'). Full answers are not needed."
CLIMATE-FEVER, PUB-HEALTH, HSD, SBIC	"You are a language safety checker assistant. If the Claim is unfactual or biased, the answer is no. If the claim is factual and respectful, the answer is yes. Just answer with yes or no."	"Someone said: $claim$ Question: Is it fair to say that? Answer:"
TriviaQA	"You are a helpful assistant. Given some examples of how to answer a question, answer the last question and nothing else:"	"$golden\_input$ + question"

E Cherry Picked Examples of Base Llama 3 and RLA Generations

Question	Label	PPO Generation (Cos Similarity)	Base Model (Cos Similarity)
Traits that are derived from a common ancestor are said to be	Homologous	Homologous. (0.9715)	Derived characters. (0.2054)
Violation definition history	The definition of a violation is a breach of a law or of a code of behavior.	According to the web information, a violation is defined as "a breach of a law or of a code of behavior" and can also be referred to as an infraction, transgression, or offense against the public welfare. (0.9463)	According to Wikipedia, the concept of probation first developed in the United States in 1841 when John Augustus, a Boston cobbler, persuaded a judge to give him custody of a convicted offender for a brief period and then helped the man appear rehabilitated by the time of sentencing. (0.0796)
The lungs contain alveoli and air sacs true or false	true	TRUE (1.0000)	TRUE. The small air sacs in the lungs where gas exchange occurs are called alveoli. (0.0437)