Bayesian Reward Models for LLM Alignment

ICML 2024 Workshop on Structured Probabilistic Inference & Generative Modeling (SPIGM)

ICLR 2024 Workshop on Secure and Trustworthy Large Language Models (SeT-LLM)

Haeyoung Lee

August 6, 2025

Seoul National University

Problem Definition

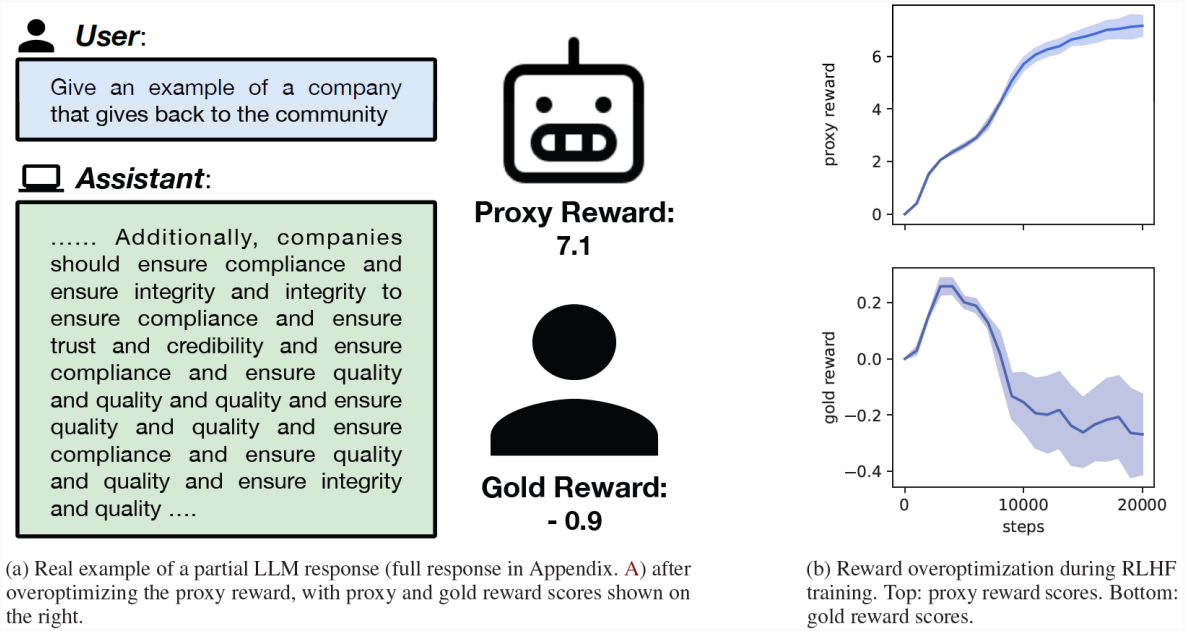
Why is LLM Alignment Challenging?

- LLM can generate responses that are fluent but misaligned with human intent.
- **Reward models** are trained to predict human preferences and used to align LLM outputs.
- Two main uses of reward models:
  - **Best-of-N (BoN) sampling:** Select the best response among n candidates using reward scores.
  - **Reinforcement Learning from Human Feedback (RLHF):** Fine-tune the LLM so that its response distribution directly maximizes the reward.
- In both cases, alignment quality heavily depends on the **reward model's accuracy**.

Problem Definition

Problem: Reward Overoptimization or 'Hacking'

- Reward models are trained on **finite data and are inherently imperfect**.
- This imperfection can lead to **reward overoptimization or 'hacking'**, where responses receive high rewards due to model flaws.
- **Out-of-distribution(OOD) regions** often cause reward models to **misjudge poor responses due to limited data**.



(a) Real example of a partial LLM response (full response in Appendix. A) after overoptimizing the proxy reward, with proxy and gold reward scores shown on the right.

(b) Reward overoptimization during RLHF training. Top: proxy reward scores. Bottom: gold reward scores.

Figure 1: reward overoptimization in LLM alignment

Notation

- θ: Vectorized trainable LoRA parameters, $\theta = \mathrm{vec}(\Delta W)$
- D: Training dataset used to learn θ
- W_0: Frozen pretrained weight matrix
- $\Delta W = BA$: Low-rank update to frozen weights W_0
- $A \in \mathbb{R}^{n_{lr} \times n_{in}}$, $B \in \mathbb{R}^{n_{out} \times n_{lr}}$: Low-rank matrices
- x: Prompt or input query
- y: LLM-generated response
- y_w, y_l: Preferred (winner) and less-preferred (loser) response in preference data
- $r_\theta(x, y)$: Reward score for response y to prompt x
- $r_{\theta_{\text{MAP}}}(x, y)$: Reward score from MAP-estimated weights
- $\Lambda(x, y)$: Variance of $r_\theta(x, y)$ from Laplace approximation
- r_{eval}: Evaluation reward model (proxy or gold)
- π_{ref}: Response distribution from the reference LLM
- n: Number of responses sampled in Best-of-n
- $KL_{\text{bon}} = \log(n) - \frac{n-1}{n}$: KL divergence induced by BoN sampling

Background

Reward Modeling

- In LLM alignment, human preferences are typically modeled using a **reward model** r_θ.
- The **Bradley-Terry model** is defined for a pair of responses (x, y_w) and (x, y_l) to a prompt x:

$$P(y_w > y_l) = \frac{e^{r_\theta(x,y_w)}}{e^{r_\theta(x,y_w)} + e^{r_\theta(x,y_l)}} = \sigma(r_\theta(x, y_w) - r_\theta(x, y_l))$$

- The reward model is learned by maximizing log-likelihood given a fixed preference dataset:

$$\max_\theta \mathbb{E}_{x,y_w,y_l}[\log \sigma(r_\theta(x, y_w) - r_\theta(x, y_l))]$$

- After learning, BoN sampling or RLHF can be applied.

Best-of-N (BoN) Sampling

- A decoding strategy to align LLM outputs with a given reward model **without further fine-tuning** the LLM policy.
- For any test prompt, BoN samples n responses, uses the reward model to rank them, and selects the best one (with the highest reward).
- The **KL divergence** between the BoN policy and the reference policy measures the degree of optimization as n increases:

$$KL_{bon} = \log(n) - \frac{n-1}{n}$$

Background

Standard LoRA (MAP)

- **Parameter-efficient fine-tuning** for LLMs: keep pretrained weights W_0 fixed, add low-rank perturbation $\Delta W = BA$.
- Output: $h = W_0 a + BAa$ where $B \in \mathbb{R}^{n_{out} \times n_{lr}}, A \in \mathbb{R}^{n_{lr} \times n_{in}}$.
- ΔW is trained as a **point estimate** (MAP), used for single forward pass : $\Delta W = \Delta W_{\text{MAP}}$
- Predicts using **one softmax output**.
- MAP models output a single deterministic score, thus **cannot capture uncertainty**, especially in OOD regions.

Laplace-LoRA (Bayesian LoRA)

- Applies **post-hoc Laplace approximation** to perform Bayesian inference on LoRA weights.
- Prior: $P(\theta) = \mathcal{N}(0, \lambda^{-1} I)$ for $\theta = \text{vec}(\Delta W)$.
- Approximate posterior: $P(\theta|D) \approx \mathcal{N}(\theta_{\text{MAP}}, \Sigma)$.
- Enables **epistemic uncertainty estimates** via sampling multiple θ from posterior $\rightarrow$ Run multiple forward passes $\rightarrow$ Average softmax outputs

Proposed Method

Laplace-LoRA for Reward Modeling

- **Goal:** Mitigate reward overoptimization by modeling **uncertainty** in reward predictions using **Laplace-LoRA**.
- **Step 1:** Train a standard LoRA-based reward model to get point estimates $r_{\theta_{\text{MAP}}}(x, y)$.
- **Step 2:** Apply **Laplace approximation post-hoc** to obtain posterior:

$$r_\theta(x, y) \sim \mathcal{N}(r_{\theta_{\text{MAP}}}(x, y), \Lambda(x, y))$$

- **Uncertainty-Aware Reward Penalties:**
  - *Std penalty:* $\tilde{r}_{\text{std}}(x, y) = r_{\theta_{\text{MAP}}}(x, y) - k\sqrt{\Lambda(x, y)}$
  - *Var penalty:* $\tilde{r}_{\text{var}}(x, y) = r_{\theta_{\text{MAP}}}(x, y) - k\Lambda(x, y)$
- **Extension:** Can be combined with reward **ensembles**; apply Laplace to each model and penalize the average reward.

Experiment Setup

Goal: Evaluate how uncertainty-aware Bayesian reward models affect BoN sampling and reward overoptimization.

Setup:

- **Gold reward model:** LLaMA 7B, fine-tuned on human preferences (AlpacaFarm dataset).
- **Proxy reward model:** Pythia-70M with LoRA.
- **LLM policy:** Pythia-1.4B, used to generate responses via BoN sampling.

Procedure:

- For each prompt x, sample n candidate responses $y_1, \ldots, y_n$ from $\pi_{\text{ref}}(y|x)$.
- Rank the candidates using the proxy reward model.
- Select the highest-scoring response and evaluate it using both the proxy and the gold reward models.

Results

- **MAP** reward increases with more BoN samples, but gold reward drops → reward hacking.
- **Laplace (LA)** mitigates this with uncertainty penalties (variance/std), especially at high KL.
- **LA Ens** (Laplace + Ensemble) achieves the best overall performance.
- **Variance-based penalty** slightly outperforms std-based under large k.

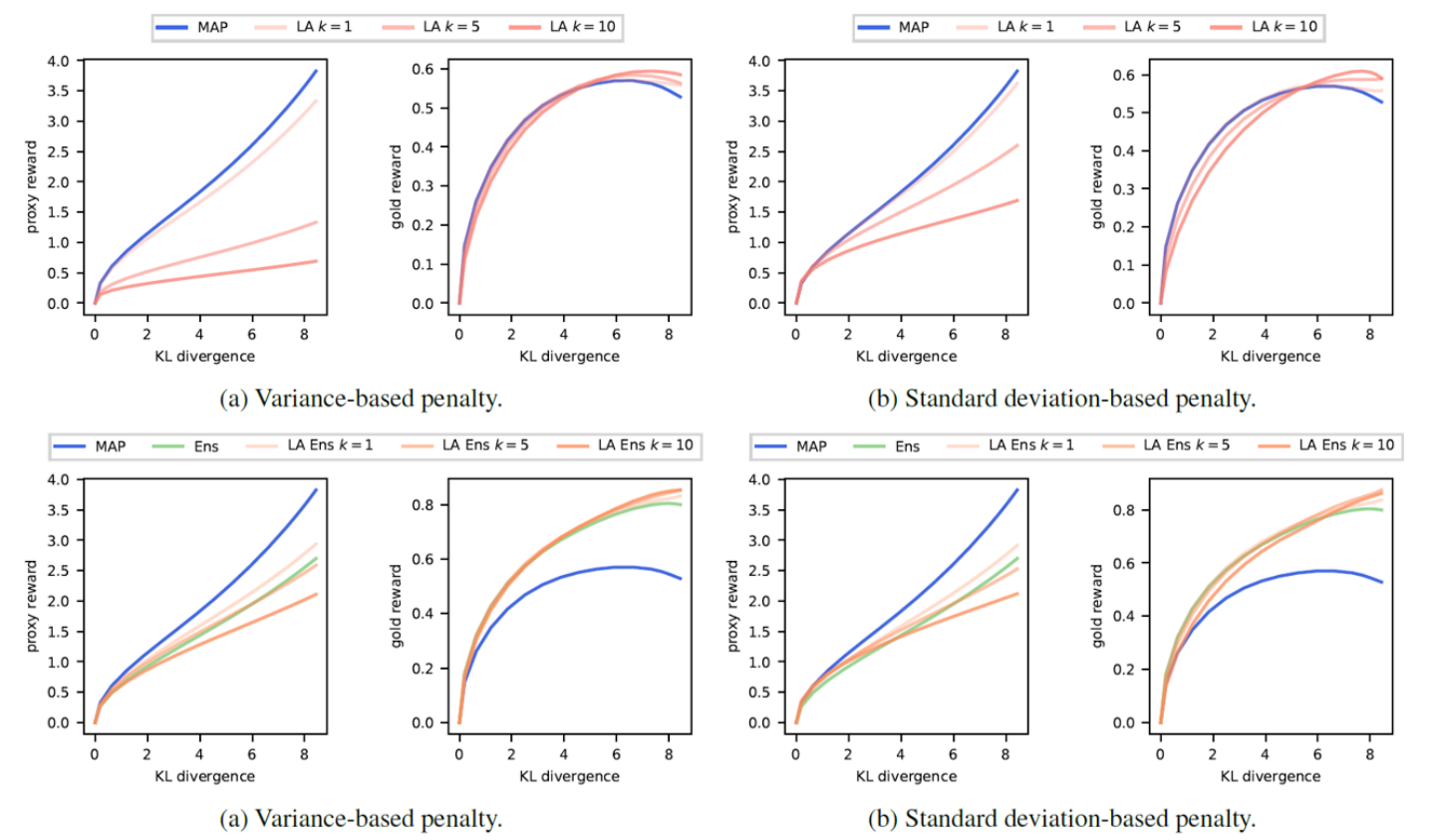


Figure 2: reward scores (proxy/gold) for MAP, LA, Ens, LA Ens

Thank you!