REWARD MODELING IN LLM ALIGNMENT

Shuo Liu
Computer Science
Northeastern University
shuo.liu2@northeastern.edu

ABSTRACT

Constructing rewards is crucial yet challenging to achieve RL objectives. This post explores how to guide the LLM optimization via a proper RLHF reward model.

1 REWARD MODELS IN RLHF

As LLMs scale, their raw outputs (optimized primarily for next-token prediction) often diverge from expected traits. To enable RL fine-tuning from human feedbacks (RLHF), reward models are introduced as trainable proxies for human preference. Once trained, it can generalize preference signals to unseen inputs, making alignment more scalable by reducing reliance on slow and costly human annotations. It also allows flexible fine-tuning toward different objectives, such as helpfulness, truthfulness, or safety.

A typical alignment pipeline consists of 3 stages: supervised fine-tuning (SFT), reward modeling, and RL. After an initial SFT based on base transformer with curated human-labeled data, a reward model is constructed to predict human preferences over model-generated responses. This model is then used to guide further optimization by encouraging outputs that maximize the predicted reward. For example, `ChatGPT` employs reward models trained on ranked annotations to guide its generation toward preferred outputs Achiam et al. (2024); `DeepSeek` and `LLaMA 2` include explicit reward modeling components in their alignment pipelines, using pairwise preferences to train reward models that inform subsequent learning Shao et al. (2024); Touvron et al. (2023).

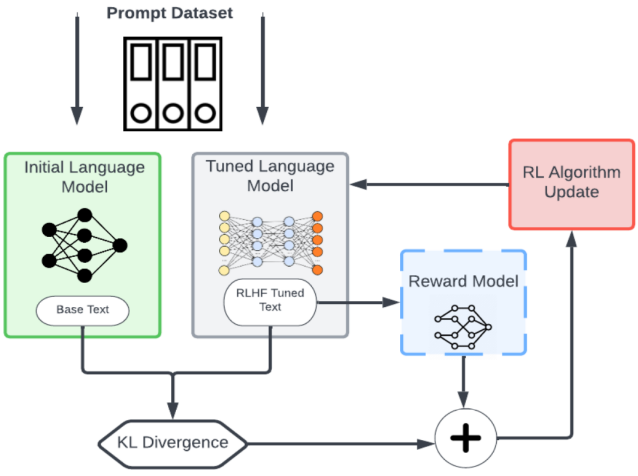

Figure 1: Role of the reward models in RLHF (reward model in blue).

2 ANTI-SYMMETRIC PREFERENCE MODELING

In this section, we introduce the mainstream methods that model rewards of LLM responses through preference comparison.

2.1 BT MODEL AND ITS RANKING EXTENSION

The original Bradley-Terry (BT) model posits that, given a pair of options i and j drawn from some population, the probability of selecting i is,

$$\Pr(i \succ j) = \frac{u_i}{u_i + u_j} = \frac{\exp(r(i))}{\exp(r(i)) + \exp(r(j))}, \tag{1}$$

where u_i and u_j are the respective utility or preference of options i and j, which commonly parameterized in the exponential form via rewards $r(i)$ and $r(j)$. It can be further extended to rank N options, known as Plackett-Luce (PL) model,

$$\Pr(i_1 \succ i_2 \succ \cdots \succ i_N) = \prod_{k=1}^{N} \frac{\exp(r(i_k))}{\sum_{j=k}^{N} \exp(r(i_j))}. \tag{2}$$

BT model is anti-symmetric, which means the preference between two responses depends only on the difference in their reward values. It satisfies $\Pr(y_i \succ y_j) = 1 - \Pr(y_j \succ y_i)$, and the log-odds of preference is anti-symmetric: $\log\left(\frac{\Pr(y_i \succ y_j)}{\Pr(y_j \succ y_i)}\right) = r(x, y_i) - r(x, y_j)$. This structure ensures consistent and transitive pairwise comparisons, making BT suitable for preference modeling. This is initially used to rank sports teams and players, i.e., Elo rating system.

2.2 REWARD MODELING WITH PAIRWISE PREFERENCES (BT MODEL)

Inferring a reward model using the BT framework can be formulated as a parameter estimation problem, where the objective is to recover latent reward values for candidate responses based on observed pairwise comparisons.

Suppose a prompt x is associated with N candidate responses $\{y_1, \cdots, y_N\}$, and human annotators provide preference labels between some pairs. Ideally, given sufficiently many comparisons under deterministic preference, the true reward values can be accurately inferred.[1] In practice, however, this inference is challenged by stochastic human behavior and sparse annotation.

Modeling Assumptions To make BT applicable to reward modeling under the practical conditions, we adopt the following assumptions Sun et al. (2025):

1. `Deterministic responses and rewards`: For a given prompt x, a response y is deterministically generated by a model. The oracle reward $r(x, y)$ associated with each prompt-response pair is fixed.
2. `Deterministic annotators with bounded bias`: When a an annotator A compares responses, their preferences are deterministically depending on the comparison of their biased evaluation of the reward,

$$\mathbb{1}(\underbrace{y_i \succ y_j}_{\text{decision}} | x, A) = \mathbb{1}(\underbrace{r(x, y_i) + b(x, y_i; A) > r(x, y_j) + b(x, y_j; A)}_{\text{biased preference}}). \tag{3}$$

3. `Order-preserving shaping`: To address sparsity in comparisons, a known embedding function Ψ is assumed to map each (x, y) pair into a feature space, with the constraint that Ψ is order-preserving and does not affect optimization. High-dimensional embeddings can be beneficial because they increase the likelihood that semantically similar prompt-response pairs are mapped to nearby regions in the embedding space, thereby enabling comparisons to generalize more effectively. Though, it also introduces potential issues such as *reward hacking* (Section 5).
4. `Imperfect human annotations`: The annotator function $h(x_1, x_2, y_1, y_2)$ provides feedback that possibly aligns with the oracle reward $r(x, y)$. And it is harder to assign correctly when the reward difference between two pairs is small,

$$\mathbb{P}\left(h(x_1, x_2, y_1, y_2)(r(x_1, y_1) - r(x_2, y_2)) > 0 \,|\, \Delta r\right) = \xi(\Delta r), \tag{4}$$

where $\Delta r := |r(x_1, y_1) - r(x_2, y_2)|$ is the reward difference, and ξ is any monotonic increasing function to $[0.5, 1]$.

[1] $O(N \log N)$ comparisons are sufficient for modeling rewards of N responses.

Likelihood and Estimation Let each observed preference be denoted as a pair $(i \succ j)$, indicating that y_i is preferred over y_j for prompt x. Under the BT model, the probability of this outcome is,

$$\Pr(y_i \succ y_j | x) = \frac{\exp(r(x, y_i))}{\exp(r(x, y_i)) + \exp(r(x, y_j))}. \tag{5}$$

Given a dataset of M annotated comparisons $\mathcal{C} = \{(i_m, j_m)\}_{m=1}^{M}$, the likelihood of observing these preferences is,

$$\mathcal{L}(r) = \prod_{m=1}^{M} \frac{\exp(r(x, y_{i_m}))}{\exp(r(x, y_{i_m})) + \exp(r(x, y_{j_m}))}, \tag{6}$$

and the corresponding log-likelihood is,

$$\log \mathcal{L}(r) = \sum_{m=1}^{M} \left[r(x, y_{i_m}) - \log \left(\exp(r(x, y_{i_m})) + \exp(r(x, y_{j_m})) \right) \right]. \tag{7}$$

The optimal reward model is then obtained by maximizing the log-likelihood (MLE),

$$r^* = \operatorname*{argmax}_r \log \mathcal{L}(r). \tag{8}$$

The solution is identifiable up to an additive constant and converges to the ground-truth rewards under the assumptions above.

2.3 Reward Modeling with Ranked Preferences (PL Model)

While the BT model provides a principled way to infer rewards from pairwise preferences, real-world systems can collect richer feedback in the form of ranked lists. Human annotators are presented with multiple responses to the same prompt and asked to provide an overall ranking. The reward model is then trained to assign scores that agree with these rankings.

PL model Equation. 2 is used in a similar way for MLE, just like BT model. For a ranking of alternatives $(y_{i_1} \succ y_{i_2} \succ \ldots \succ y_{i_N})$ for prompt x, the probability under the PL model is,

$$\Pr(y_{i_1} \succ y_{i_2} \succ \ldots \succ y_{i_N} | x) = \prod_{k=1}^{N-1} \frac{\exp(r(x, y_{i_k}))}{\sum_{j=k}^{N} \exp(r(x, y_{i_j}))}. \tag{9}$$

Given a dataset of M rankings $\mathcal{C} = \{(y_{i_1^m}, y_{i_2^m}, \ldots, y_{i_{N_m}^m})\}_{m=1}^{M}$, the likelihood is,

$$\mathcal{L}(r) = \prod_{m=1}^{M} \prod_{k=1}^{N_m - 1} \frac{\exp(r(x, y_{i_k^m}))}{\sum_{j=k}^{N_m} \exp(r(x, y_{i_j^m}))}. \tag{10}$$

And the corresponding log-likelihood is,

$$\log \mathcal{L}(r) = \sum_{m=1}^{M} \sum_{k=1}^{N_m - 1} \left[r(x, y_{i_k^m}) - \log \left(\sum_{j=k}^{N_m} \exp(r(x, y_{i_j^m})) \right) \right]. \tag{11}$$

The optimal reward model is then obtained by MLE,

$$r^* = \arg\max_r \log \mathcal{L}(r). \tag{12}$$

Preference modeling with the PL model is also anti-symmetric, swapping the order of two responses in the ranking inverts the relative score difference in the likelihood expression.

3 SYMMETRIC REWARD MODELING

As clich as it sounds, modeling rewards of LLM responses through symmetric preference signals is indeed feasible. Symmetric models predict the reward for each prompt-response pair independently, without reference to alternative responses.

3.1 REGRESSION-BASED REWARD MODEL

A common symmetric approach is to train a regression-based reward model on scalar human ratings. Given a dataset $\{(x_n, y_n, s_n)\}_{n=1}^N$, where $s_n \in \mathbb{R}$ denotes the score assigned to response y_n for prompt x_n, the reward model $r(x, y)$ is trained to minimize a squared loss:

$$\mathcal{L}_{\text{reg}} = \sum_{n=1}^{N} \left(r(x_n, y_n) - s_n\right)^2. \tag{13}$$

Such scalar-labeled data is available in open datasets like `Anthropic HH`, `OpenAssistant`, and `MT-Bench`, where annotators assign numeric quality scores to individual responses without ranking alternatives.

3.2 CLASSIFICATION-BASED REWARD MODEL

Alternatively, a reward model can be trained as a binary classifier to predict whether a response satisfies certain alignment criteria (e.g., helpfulness or harmlessness). In this case, binary labels $s_n \in \{0, 1\}$ indicate whether a response is considered acceptable, and the reward model is trained using a standard cross-entropy loss with sigmoid σ activation,

$$\mathcal{L}_{\text{cls}} = -\sum_{n=1}^{N} \left[s_n \log \sigma(r(x_n, y_n)) + (1 - s_n) \log(1 - \sigma(r(x_n, y_n)))\right]. \tag{14}$$

4 OTHER REWARD MODELING TECHNIQUES

4.1 INVERSE REINFORCEMENT LEARNING

Inverse Reinforcement Learning (IRL) aims to recover an underlying reward function that explains the behavior of an expert operating within a MDP. Formally, consider an MDP defined by the tuple $(\mathcal{X}, \mathcal{Y}, T, \gamma)$, where $\mathcal{X}$ is the set of states, $\mathcal{Y}$ the set of actions, $T(x'|x, y)$ the transition dynamics, and γ the discount factor. Given a set of expert trajectories τ_i, where each trajectory $\tau_i = (x_0, y_0, x_1, y_1, \dots)$, the goal of IRL is to infer a reward function $r : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ such that the optimal policy induced by this reward function explains the observed expert behavior. However, this does not match seamlessly for reward modeling in LLM alignment due to,

- `Unstructuredness`: IRL assumes access to expert trajectories in a well-defined MDP, which does not align with the unstructured, high-dimensional nature of language modeling.
- `Incompatible feedback format`: Human feedback for LLMs is typically given as relative preferences between completions rather than as optimal action sequences.
- `Huge computation`: IRL methods are computationally expensive and require solving RL problems repeatedly, which is impractical for large-scale language models.

4.2 BAYESIAN REWARD LEARNING

Bayesian reward learning explicitly represents uncertainty about the reward function by maintaining a posterior distribution $p(\theta|D)$ over reward parameters θ, given observed demonstration data D. Using Bayesian inference, it updates a prior distribution $p(\theta)$ to a posterior, $p(\theta|D) \propto p(D|\theta)p(\theta)$ allowing the derived policy to incorporate uncertainty (e.g., via posterior sampling). However, Bayesian inference in high-dimensional reward spaces is computationally expensive and impractical for large-scale LLMs, since the exact posterior inference is often intractable.

5 TAKEAWAYS IN REWARD DESIGN

Effective reward design is critical in guiding RL agents towards desirable behaviors. Several considerations and specialized techniques help ensure robust and aligned outcomes. Here are some specific takeaways should be considered when designing rewards.

Potential-based Reward Shaping Reward shaping accelerates learning by augmenting the original reward with additional informative signals. Potential-based shaping ensures optimal policy invariance by defining additional rewards through a state-dependent potential function $\Phi(s)$. Specifically, the shaping reward is formulated as $F(s, a, s') = \gamma\Phi(s') - \Phi(s)$, thereby avoiding the introduction of unintended behaviors or reward hacking vulnerabilities.

"*Rubric engineering is the new prompt engineering.*" *– Will Brown.*

Sparse vs. Dense Rewards The frequency and clarity of feedback significantly influence learning efficiency and agent behaviors. Sparse rewards, given infrequently and typically at task completion, mitigate reward hacking risks but pose exploration challenges Weng (2024).[2] Conversely, dense rewards provide frequent feedback that facilitates exploration yet may inadvertently encourage reward exploitation or unintended optimization behaviors.

Human-in-the-loop Reward Design Incorporating human judgments directly into reward signals enables iterative refinement, capturing nuanced and complex objectives not easily formalized. Human-in-the-loop methods mitigate reward hacking by continuously aligning rewards with intended outcomes through active human oversight.

Intrinsic vs. Extrinsic Rewards Intrinsic rewards originate internally, driven by agent behaviors such as curiosity or novelty-seeking, promoting adaptive and exploratory capabilities. Extrinsic rewards come externally, explicitly set by designers. A balanced integration of intrinsic and extrinsic rewards supports robust agent behaviors through diversified motivational signals.

REFERENCES

Josh Achiam, Steven Adler, and Sandhini Agarwal. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Zhihong Shao, Peiyi Wang, and Qihao Zhu. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Hao Sun, Yunyi Shen, and Jean-Francois Ton. Rethinking bradley-terry models in preference-based reward modeling: Foundations, theory, and alternatives, 2025. URL https://arxiv.org/abs/2411.04991.

Hugo Touvron, Louis Martin, and Kevin Stone. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Lilian Weng. Reward hacking in reinforcement learning. *lilianweng.github.io*, Nov 2024. URL https://lilianweng.github.io/posts/2024-11-28-reward-hacking/.

[2]The ambiguous or underspecified rewards design that creates vulnerabilities for agents' exploitation.