

Beyond Reward Modeling: General Preference Modeling with Preference Representations

Yifan Zhang*, Ge Zhang*, Yue Wu*, Kangping Xu, Quanquan Gu

IIS, Tsinghua University & UCLA



Samueli
School of Engineering

Outline

- Introduction to RLHF & Reward Modeling
- General Preference Modeling
- General Preference Optimization
- Concluding Remarks

Yifan Zhang*, Ge Zhang*, Yue Wu*, Kangping Xu, Quanquan Gu

IIS, Tsinghua University & UCLA

+

•

○

Outline

- **Introduction to RLHF & Reward Modeling**
- General Preference Modeling
- General Preference Optimization
- Concluding Remarks

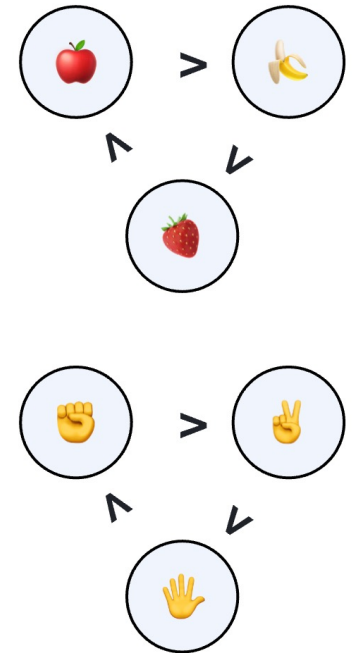
Yifan Zhang*, Ge Zhang*, Yue Wu*, Kangping Xu, Quanquan Gu

IIS, Tsinghua University & UCLA





Motivation

- **Aligning Language Model with Human Values**
 - Essential for safe and effective AI interaction.
 - Traditional models struggle with complex human preferences.
- **Limitations of Existing Method**
 - **Bradley-Terry (BT) Reward Model:**
 - Assumes transitive (and additive) preferences.
 - Cannot handle intransitive (cyclic) preferences.

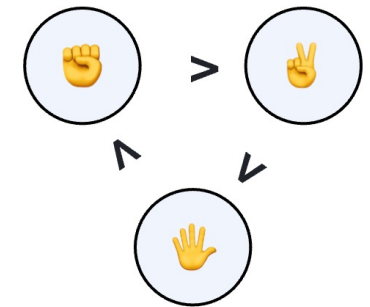


Reward Modeling & Preference Modeling

- Using an dataset of completions, we collect **pairwise feedback** from humans:
 - $x \rightarrow y, y' \sim \mu(x)$, where x represents instructions and y, y' are responses.
 - $\Rightarrow y_w, y_l \sim P(y_w \succ y_l)$
 - y win:  y lose:  (human preferences)
- , where y_w is the winning response and y_l is the losing response, as determined on **human preferences**.
- An important special case is when human preferences are actually following a Bradley-Terry (BT) reward model: $BT(y_w, y_l | x) = P(y \succ y') = \sigma(r(y_w | x) - r(y_l | x))$

Limitations of Bradley-Terry Reward Model

- BT models **cannot model non-transitivity** (Gardner, 1970) or non-additivity (Bertrand et al 2023).
 - Rock-paper-scissors game
 - In BT model, it will give
 - $\text{Reward}(\text{Rock}) = \text{Reward}(\text{Scissor}) = \text{Reward}(\text{Paper})$
 - Such that
 - $P(y > y') = 0.5$
 - *This is not correct!*



Challenges in Preference Modeling

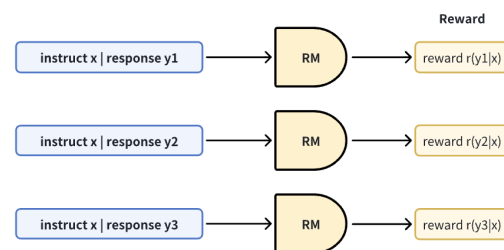
- **Intransitive Preferences in Real World**

- Human preferences are not always transitive, and can be cyclic (intransitive) e.g., A preferred over B, B over C, but C over A.
- Traditional models such as BT assume transitivity, which fails to capture such preferences.

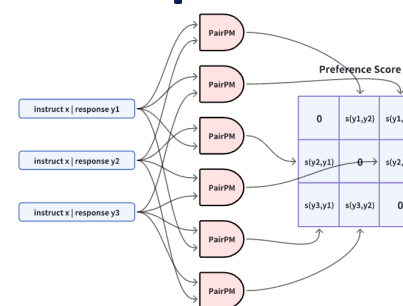
- **Computational Efficiency**

- Pairwise comparisons (PairPM / PairRM) becomes computationally expensive with $O(K^2)$ complexity for K responses.

Need for models that are both expressive and computationally efficient.



(a) Bradley-Terry (BT) reward model



(b) PairPM

Challenges in Preference Modeling

- **Is there a principled way to model general preference?**
- Can we perform general preference modeling that are both expressive and computationally efficient?

We answer this question affirmatively by proposing
Preference Representation Learning.

General Preference Modeling With Preference Representations

- **Preference Representation Learning**

- Embeds (contextual) responses into a latent space.
- Our General Preference representation Model (GPM) can capture complex and intransitive preference structures.
- Achieves linear computational complexity $O(K)$ when comparing K responses.

- **General Preference Optimization (GPO)**

- Utilizes preference scores calculated using preference representations (embeddings) instead of scalar rewards.
- Generalizes reward-based reinforcement learning from human feedback.

Zhang et al., General Preference Modeling with Preference Representations for Aligning Language Models (arxiv.org/abs/2410.02197)

Outline

- Introduction to RLHF & Reward Modeling
- **General Preference Modeling**
- General Preference Optimization
- Concluding Remarks

Yifan Zhang*, Ge Zhang*, Yue Wu*, Kangping Xu, Quanquan Gu

IIS, Tsinghua University & UCLA



Background & Notations

In this section, we present preliminaries on reward modeling, preference modeling, and reinforcement learning from human feedback (RLHF) for language model alignment. We consider an autoregressive language model that generates responses to the given prompts. Let $\mathbf{x} = [x_1, x_2, \dots]$ denote a prompt, a sequence of tokens. The language model π generates a response $\mathbf{y} = [y_1, y_2, \dots, y_N]$ based on the conditional probability distribution: $\pi(\mathbf{y} \mid \mathbf{x}) = \prod_{i=1}^N \pi(y_i \mid \mathbf{x}, \mathbf{y}_{<i})$, where $\mathbf{y}_{<i}$ represents the sequence of tokens generated before position i . In this paper, we assume a general-preference oracle. Given two responses \mathbf{y} and \mathbf{y}' to the same prompt \mathbf{x} , the oracle provides the feedback indicating which response is preferred.

$$\mathbb{P}(\mathbf{y} \succ \mathbf{y}' \mid \mathbf{x}) := \mathbb{E}[o(\mathbf{y} \succ \mathbf{y}' \mid \mathbf{x})].$$

Reward-based Reinforcement Learning from Human Feedback (RLHF)

The most prevalent approach to aligning language models with human preferences is to consider a scalar reward function $r(\mathbf{y}; \mathbf{x})$ that assigns a numerical score to each response. The preference between two responses is then determined solely by the reward scores for the two responses. For example, the Bradley-Terry (BT) model ([Bradley & Terry, 1952](#)) is a widely used method for modeling pairwise preferences in this context. However, the BT model can not capture intransitive (e.g. cyclic) preferences effectively ([Bertrand et al., 2023](#)). Under the BT model, the probability that response \mathbf{y} is preferred over \mathbf{y}' is given by:

$$\mathbb{P}(\mathbf{y} \succ \mathbf{y}' \mid \mathbf{x}) = \sigma(r(\mathbf{y}; \mathbf{x}) - r(\mathbf{y}'; \mathbf{x})),$$

where $\sigma(z) = 1/(1 + e^{-z})$ is the logistic (sigmoid) function.

In practice, the reward function $r(\mathbf{y}; \mathbf{x})$ is learned by maximizing the likelihood of the observed preference data. Once the reward function is established, policy optimization techniques, such as Proximal Policy Optimization (PPO) ([Schulman et al., 2017](#)), can be applied to adjust the language model to generate responses that maximize expected rewards. The optimization problem can be formulated as:

$$\max_{\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}, \mathbf{y} \sim \pi_{\theta}(\cdot \mid \mathbf{x})} [r(\mathbf{y}; \mathbf{x})] - \beta \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} [\text{KL}(\pi_{\theta}(\cdot \mid \mathbf{x}) \parallel \pi_{\text{ref}}(\cdot \mid \mathbf{x}))], \quad (3.1)$$

where θ are the parameters of the policy π_{θ} , π_{ref} is a reference policy (often the pre-trained or supervised-fine-tuned language model), β is a scaling parameter that controls the strength of regularization, and KL denotes the Kullback-Leibler divergence.

General Preference Modeling

We consider the scenario where given a prompt \mathbf{x} , a set of responses $\{\mathbf{y}_i\}$ is generated, and human preferences over these responses are represented as pairwise probabilities $\mathbb{P}(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) \in (0, 1)$, indicating the likelihood that response \mathbf{y}_i is preferred over \mathbf{y}_j given the prompt \mathbf{x} .

To model these preferences, we define a (pairwise) preference score function:

$$s(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) := \log \frac{\mathbb{P}(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x})}{1 - \mathbb{P}(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x})}, \quad (3.2)$$

which represents the log-odds of \mathbf{y}_i being preferred over \mathbf{y}_j . This score function allows us to express the preference probability as:

$$\mathbb{P}(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) = \sigma(s(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x})), \quad (3.3)$$

where $\sigma(z) = 1/(1 + e^{-z})$ is the logistic function. One can see that the BT model is a special case: $s(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) = r(\mathbf{y}_i; \mathbf{x}) - r(\mathbf{y}_j; \mathbf{x})$.

Supervised Pair Preference Models (PairPM & PairRM)

Existing approaches often involve concatenating the prompt and responses with a template and training an LLM-based sequential classifier in a supervised learning manner. For example, [Jiang et al. \(2023\)](#) simply concatenate the three segments $(\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$ sequentially and form a single input sequence with special tokens as separators:

```
'<s> <source> x </s> <candidate1> y1 </s> <candidate2> y2 </s>'
```

Then a sequential classification head on the last token is trained to predict the preference. Another example is [Munos et al. \(2023\)](#), which uses the following template for text summarization:

```
'You are an expert summary rater. Given a piece of text and two of  
its possible summaries, output 1 or 2 to indicate which summary  
is better.'
```

```
Text - <text>, Summary 1 - <summary1>, Summary 2 - <summary2>.
```

```
Preferred Summary -'
```

Then use the last logit for an arbitrarily chosen token as $s(\mathbf{y}_1 \succ \mathbf{y}_2 | \mathbf{x})$ for training.

However, due to the language model's position encoding ([Press et al., 2021](#); [Su et al., 2024](#)) and the causal attention ([Radford et al., 2018, 2019](#)) mechanism not being symmetric, the candidate's order in the concatenation will affect the final prediction results. It is mitigated by randomly shuffling the two responses in the training dataset but the output is still highly asymmetric. Another limitation is that how to represent the preference score can be highly ad-hoc. The two examples above already use different templates and different linear heads (sequential classification v.s. language modeling).

General Preference Modeling with Preference Representations

- **Goal:** Develop a model that captures complex human preferences efficiently
- **Key Idea:** Represent responses with preference representations (embeddings) and model preferences through their interactions.

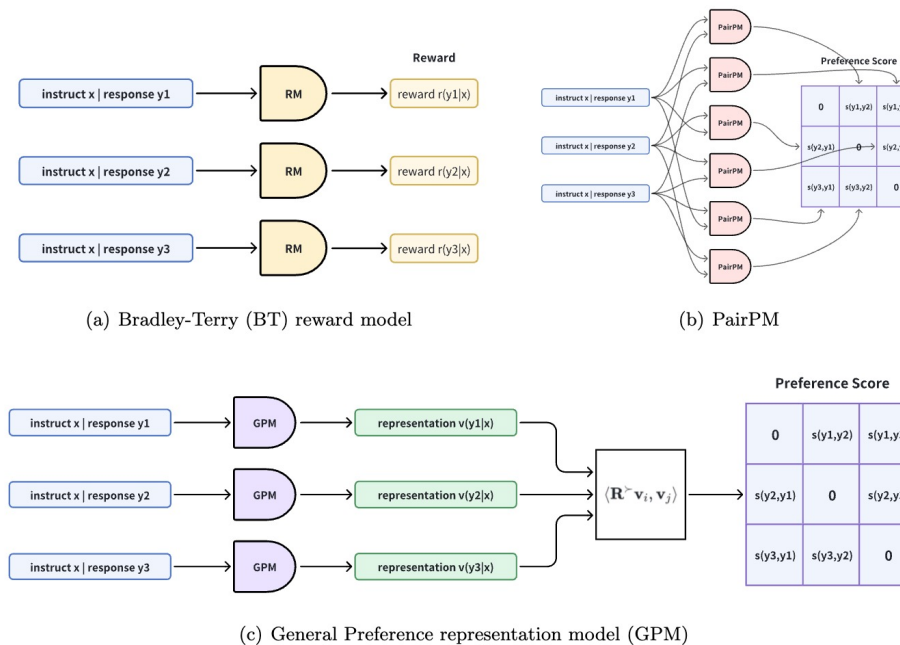


Figure 2: Illustration of (a) Bradley Terry (BT) reward model, (b) supervised pair preference model (PairPM) (Jiang et al., 2023; Dong et al., 2024), and (c) our General Preference representation model (GPM).

Preference Representations

- **Preference Representations:** Each response is embedded as a vector in a latent space, and the preferences are modeled through interactions between these representations (embeddings) using a skew-symmetric operator.

Definition 4.1 (Preference Representations). Given a prompt \mathbf{x} , we assign to each response \mathbf{y} a preference representation vector $\mathbf{v}_{\mathbf{y}|\mathbf{x}} \in \mathbb{R}^{2k}$. These representations are designed to capture the features relevant to human preferences beyond what can be represented by scalar rewards.

- Embedding captures latent features relevant to preferences.
- Enables modeling preferences beyond scalar rewards (BT model).

Preference Representations

- **Skew-Symmetric Preference Operator**

- Captures the anti-symmetric nature of preferences.
- Ensures that $s(y_i \succ y_j) = -s(y_j \succ y_i)$.

Definition 4.2 (Skew-symmetric Preference Operator). To capture the directional nature of preferences, we define a skew-symmetric (anti-symmetric) preference operator $\mathbf{R}^\succ \in \mathbb{R}^{2k \times 2k}$. Specifically, \mathbf{R}^\succ is a block-diagonal matrix consisting of k skew-symmetric blocks of the form (for more discussion, please see Appendix A):

$$\mathbf{R}_l = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad l = 1, \dots, k. \quad (4.1)$$

An example of \mathbf{R}^\succ for $k = 2$ is:

$$\mathbf{R}^\succ = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Preference Score Function

Definition 4.3 (Preference Score). The preference score between two responses \mathbf{y}_i and \mathbf{y}_j using preference representations is defined as:

$$s(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) = \langle \mathbf{R}^\succ \mathbf{v}_{\mathbf{y}_i \mid \mathbf{x}}, \mathbf{v}_{\mathbf{y}_j \mid \mathbf{x}} \rangle, \quad (4.2)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in \mathbb{R}^{2k} . This score captures the anti-symmetric relationship between responses induced by human preferences.

We model the preference probability using the logistic function as defined in Equation (3.3). Our general preference representation model (GPM) exhibits two desirable properties:

1. **Skew-symmetry.** The preference score function is skew-symmetric, satisfying:

$$s(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) = -s(\mathbf{y}_j \succ \mathbf{y}_i \mid \mathbf{x}).$$

This reflects the fact that the preference relation is naturally skew-symmetric: if \mathbf{y}_i is preferred over \mathbf{y}_j with probability $p_{i,j}$, then \mathbf{y}_j is preferred over \mathbf{y}_i with probability $1 - p_{i,j}$.

Specifically,

$$s(\mathbf{y} \succ \mathbf{y} \mid \mathbf{x}) = \langle \mathbf{R}^\succ \mathbf{v}_{\mathbf{y} \mid \mathbf{x}}, \mathbf{v}_{\mathbf{y} \mid \mathbf{x}} \rangle = 0.$$

This means that a response is neither superior nor inferior to itself.

2. **Magnitude preserving.** The skew-symmetric preference operator does not change the representation vector's magnitude, which makes this operation stable for training and inference.

$$\langle \mathbf{R}^\succ \mathbf{v}_{\mathbf{y} \mid \mathbf{x}}, \mathbf{R}^\succ \mathbf{v}_{\mathbf{y} \mid \mathbf{x}} \rangle = \langle \mathbf{v}_{\mathbf{y} \mid \mathbf{x}}, \mathbf{v}_{\mathbf{y} \mid \mathbf{x}} \rangle.$$

Preference Score Generalizes Beyond BT Reward

Relation to Bradley-Terry Model. If we set $k = 1$, $\mathbf{v}_y = [r(\mathbf{y} | \mathbf{x}), c]^\top$, where c is a constant and $c \neq 0$ (e.g., $c = 1$), and $\mathbf{R}^\succ = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$, then the preference score reduces to:

$$s(\mathbf{y}_i \succ \mathbf{y}_j | \mathbf{x}) = c(r(\mathbf{y}_i | \mathbf{x}) - r(\mathbf{y}_j | \mathbf{x})),$$

and the preference probability becomes:

$$\mathbb{P}(\mathbf{y}_i \succ \mathbf{y}_j | \mathbf{x}) = \sigma[c(r(\mathbf{y}_i | \mathbf{x}) - r(\mathbf{y}_j | \mathbf{x}))],$$

- We could find that BT reward model is a special case of our general preference model.
 - When embeddings are scalar rewards $[r(\mathbf{y} | \mathbf{x}), c]$, we recover the BT reward.

Example Case

- $P(y_w > y_l) = \sigma(v_w^\top R^\top v_l / \tau)$, where example case when $k = 1$:

- We have $v_i \in \mathbb{S}^2$, $\|v_i\|_2 = 1$, $R^\top = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$, where $v_i = (v_i(1), v_i(2))^\top$.

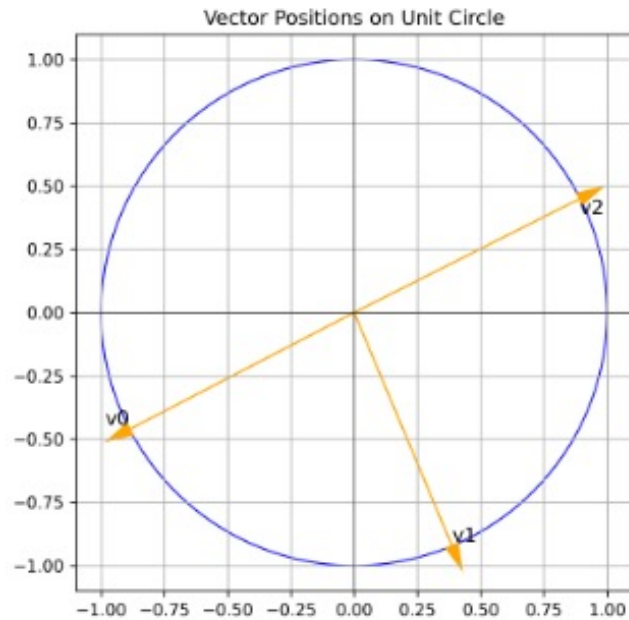
Then $s'(y_i \succ y_j) = \langle R^\top v_i, v_j \rangle = v_1(1)v_2(2) - v_1(2)v_2(1)$.

$$P(y_i \succ y_j) = \sigma(s'(y_i \succ y_j)).$$

We have the following training objective:

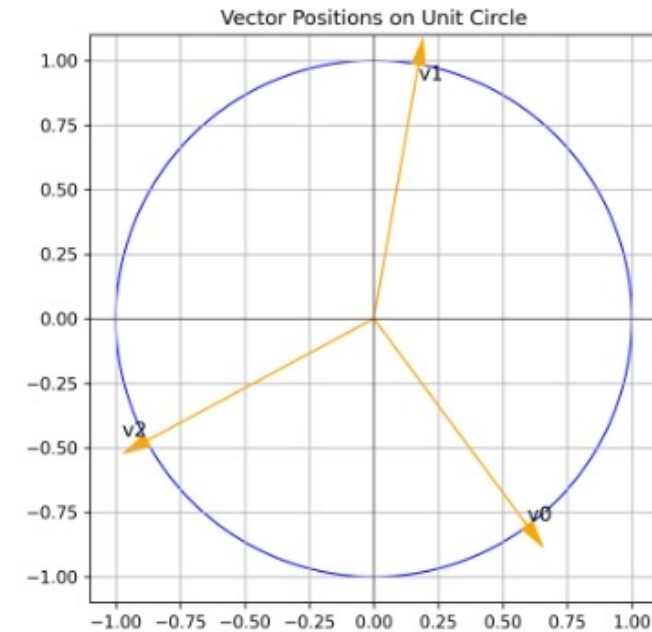
$$\mathcal{L}'_{\text{GP}} = \mathbb{E}_{\{y_1 \succ y_2\} \sim D} [-\log \sigma(s'(y_1 \succ y_2) / \tau)], \text{ one can let } \tau = 0.1.$$

GPM can handle transitive and (intransitive) cyclic tasks



Preference embedding for Transitive Game:

$$3 > 2 > 1$$



Preference embedding for Rock-Paper-Scissor Game:

$$A > B > C > A > B > C > A > B > C$$

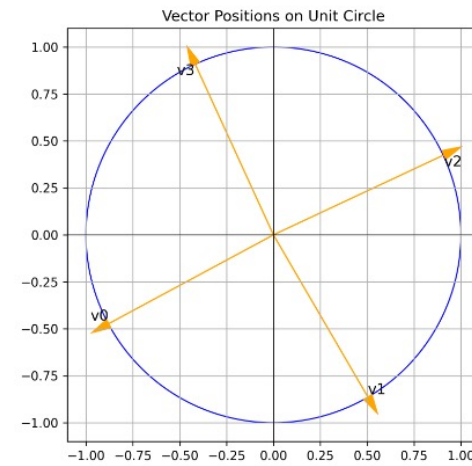
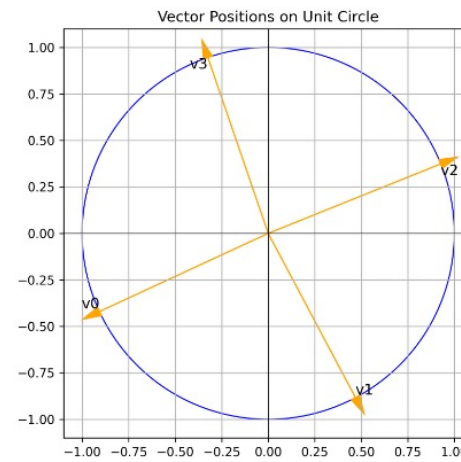
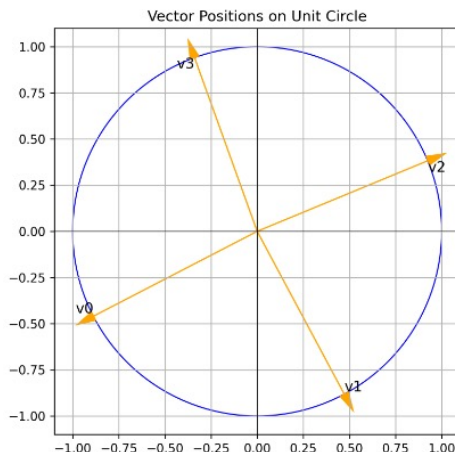
Experiments on synthetic data

Experiments on synthetic data:

$A > B > C > D$

Perfectly fitted.

```
{ "instruction": "Which character do you prefer? Choose one from [A, B, C, D].", "chosen": "A", "reject": "B", "label": 0 }
{ "instruction": "Which character do you prefer? Choose one from [A, B, C, D].", "chosen": "B", "reject": "C", "label": 1 }
{ "instruction": "Which character do you prefer? Choose one from [A, B, C, D].", "chosen": "C", "reject": "D", "label": 2 }
{ "instruction": "Which character do you prefer? Choose one from [A, B, C, D].", "chosen": "D", "reject": "A", "label": 3 }
{ "instruction": "Which character do you prefer? Choose one from [E, F, G, H].", "chosen": "E", "reject": "F", "label": 0 }
{ "instruction": "Which character do you prefer? Choose one from [E, F, G, H].", "chosen": "F", "reject": "G", "label": 1 }
{ "instruction": "Which character do you prefer? Choose one from [E, F, G, H].", "chosen": "G", "reject": "H", "label": 2 }
{ "instruction": "Which character do you prefer? Choose one from [E, F, G, H].", "chosen": "H", "reject": "E", "label": 3 }
{ "instruction": "Which character do you prefer? Choose one from [I, J, K, L].", "chosen": "I", "reject": "J", "label": 0 }
{ "instruction": "Which character do you prefer? Choose one from [I, J, K, L].", "chosen": "J", "reject": "K", "label": 1 }
{ "instruction": "Which character do you prefer? Choose one from [I, J, K, L].", "chosen": "K", "reject": "L", "label": 2 }
{ "instruction": "Which character do you prefer? Choose one from [I, J, K, L].", "chosen": "L", "reject": "I", "label": 3 }
{ "instruction": "Which character do you prefer? Choose one from [M, N, O, P].", "chosen": "M", "reject": "N", "label": 0 }
{ "instruction": "Which character do you prefer? Choose one from [M, N, O, P].", "chosen": "N", "reject": "O", "label": 1 }
{ "instruction": "Which character do you prefer? Choose one from [M, N, O, P].", "chosen": "O", "reject": "P", "label": 2 }
{ "instruction": "Which character do you prefer? Choose one from [M, N, O, P].", "chosen": "P", "reject": "M", "label": 3 }
{ "instruction": "Which character do you prefer? Choose one from [Q, R, S, T].", "chosen": "Q", "reject": "R", "label": 0 }
{ "instruction": "Which character do you prefer? Choose one from [Q, R, S, T].", "chosen": "R", "reject": "S", "label": 1 }
{ "instruction": "Which character do you prefer? Choose one from [Q, R, S, T].", "chosen": "S", "reject": "T", "label": 2 }
{ "instruction": "Which character do you prefer? Choose one from [Q, R, S, T].", "chosen": "T", "reject": "Q", "label": 3 }
{ "instruction": "Which character do you prefer? Choose one from [U, V, W, X].", "chosen": "U", "reject": "V", "label": 0 }
{ "instruction": "Which character do you prefer? Choose one from [U, V, W, X].", "chosen": "V", "reject": "W", "label": 1 }
{ "instruction": "Which character do you prefer? Choose one from [U, V, W, X].", "chosen": "W", "reject": "X", "label": 2 }
{ "instruction": "Which character do you prefer? Choose one from [U, V, W, X].", "chosen": "X", "reject": "U", "label": 3 }
```



Expressiveness of the Model

Theorem 4.4 (Expressiveness of Preference Representation Model). *Let $\mathbf{P} \in \mathbb{R}^{k \times k}$ be a real skew-symmetric matrix (i.e., $\mathbf{P} = -\mathbf{P}^\top$). Then there exist vectors $\{\mathbf{v}_i\}_{i=1}^k \subset \mathbb{R}^{2k}$ and a block-diagonal skew-symmetric matrix $\mathbf{R}^\succ \in \mathbb{R}^{2k \times 2k}$, with \mathbf{R}^\succ consisting of k blocks of the form:*

$$\mathbf{R}_l = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad l = 1, \dots, k,$$

such that:

$$P_{ij} = \mathbf{v}_i^\top \mathbf{R}^\succ \mathbf{v}_j, \quad \forall i, j.$$

Moreover, the vectors $\{\mathbf{v}_i\}$ can be constructed explicitly from \mathbf{P} .

Theorem 4.4 suggests that our preference representation framework can theoretically model arbitrary complex and potentially intransitive (e.g., cyclic) preference structures (see Appendix A.4 for proofs).

Canonical form of the Preference Operator

Proposition A.1. For any two vectors $\mathbf{v}_i \in \mathbb{R}^{2k}$ and $\mathbf{v}_j \in \mathbb{R}^{2k}$, if $\mathbf{R} \in \mathbb{R}^{2k \times 2k}$ satisfies the following two properties:

1. Skew-symmetry: $\langle \mathbf{R}\mathbf{v}_i, \mathbf{v}_j \rangle = -\langle \mathbf{R}\mathbf{v}_j, \mathbf{v}_i \rangle$.
2. Magnitude preserving: $\langle \mathbf{R}\mathbf{v}_i, \mathbf{R}\mathbf{v}_i \rangle = \langle \mathbf{v}_i, \mathbf{v}_i \rangle$.

Then \mathbf{R} must be in the form $\mathbf{R} = \mathbf{U}\mathbf{J}\mathbf{U}^\top$, where $\mathbf{U} \in \mathbb{R}^{2k \times 2k}$ is an orthonormal matrix (e.g. identity matrix \mathbf{I}_{2k}) and \mathbf{J} is a block-diagonal matrix consisting of k skew-symmetric blocks of the form:

$$\mathbf{J}_l = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad l = 1, \dots, k.$$

- Canonical form of the skew-symmetric (anti-symmetric) preference operator.

Complex Representation Interpretation

Our model can also be interpreted using complex representations. By representing the representations as complex vectors $\mathbf{v}_y \in \mathbb{C}^k$, we can express the preference score as:

$$s(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) = \text{Im} (\langle \mathbf{v}_{\mathbf{y}_i}, \mathbf{v}_{\mathbf{y}_j} \rangle),$$

where $\text{Im}(\cdot)$ denotes the imaginary part, and $\langle \cdot, \cdot \rangle$ is the Hermitian inner product. This formulation captures cyclic and intransitive preferences through the angular relationships between complex presentations.

Theorem A.2 (Expressiveness of Complex Preference Representations). Let $\mathbf{P} \in \mathbb{R}^{k \times k}$ be a real skew-symmetric matrix (i.e., $\mathbf{P} = -\mathbf{P}^\top$). Then, there exist complex vectors $\{\mathbf{v}_i\}_{i=1}^k \subset \mathbb{C}^k$ such that:

$$P_{ij} = \text{Im} (\langle \mathbf{v}_i, \mathbf{v}_j \rangle), \quad \forall i, j.$$

Example. For $k = 1$, let $\mathbf{v}_y = e^{i\theta_y}$, then:

$$s(\mathbf{y}_i \succ \mathbf{y}_j \mid \mathbf{x}) = \sin(\theta_{\mathbf{y}_i} - \theta_{\mathbf{y}_j}).$$

Implementing GPM

When the preference score matrix \mathbf{P} has an even dimension, i.e., $\mathbf{P} \in \mathbb{R}^{2k \times 2k}$, we have a more interesting interpretation based on spectral decomposition.

Theorem A.3 (Expressiveness of Preference Representation Model). Let $\mathbf{P} \in \mathbb{R}^{2k \times 2k}$ be a real skew-symmetric matrix (i.e., $\mathbf{P} = -\mathbf{P}^\top$). Then there exist representations (embeddings) $\{\mathbf{v}_i\}_{i=1}^{2k} \subset \mathbb{R}^{2k}$ and a block-diagonal skew-symmetric matrix $\mathbf{R}^\succ \in \mathbb{R}^{2k \times 2k}$, with \mathbf{R}^\succ consisting of k blocks of the form:

$$\mathbf{R}_l = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad l = 1, \dots, k,$$

such that:

$$P_{ij} = \mathbf{v}_i^\top \mathbf{R}^\succ \mathbf{v}_j, \quad \forall i, j.$$

Moreover, the representations $\{\mathbf{v}_i\}$ can be constructed from the orthogonal matrix \mathbf{U} in the decomposition of \mathbf{P} , scaled by the square roots of the positive eigenvalues of \mathbf{P} .

Implementing GPM

Eigenvalue Scale Gate. The eigenvalue scale gate \mathcal{G}_λ computes context-dependent scaling factors $\{\lambda_l(\mathbf{x})\}$, where $\lambda_l(\mathbf{x}) \geq 0$, based solely on the prompt \mathbf{x} :

$$\{\lambda_l(\mathbf{x})\} = \mathcal{G}_\lambda(\mathbf{x}).$$

This component models how different preference dimensions are weighted in the context of the given prompt, effectively adjusting the importance of various aspects such as helpfulness, instruction-following, and creativity.

Eigenvector Embedding Head. The eigenvector embedding head \mathcal{E}_v generates embeddings $\mathbf{v}_{y|\mathbf{x}}$ for each response \mathbf{y} in the context of the prompt \mathbf{x} :

$$\mathbf{v}_{y|\mathbf{x}} = \mathcal{E}_v(\mathbf{x}, \mathbf{y}).$$

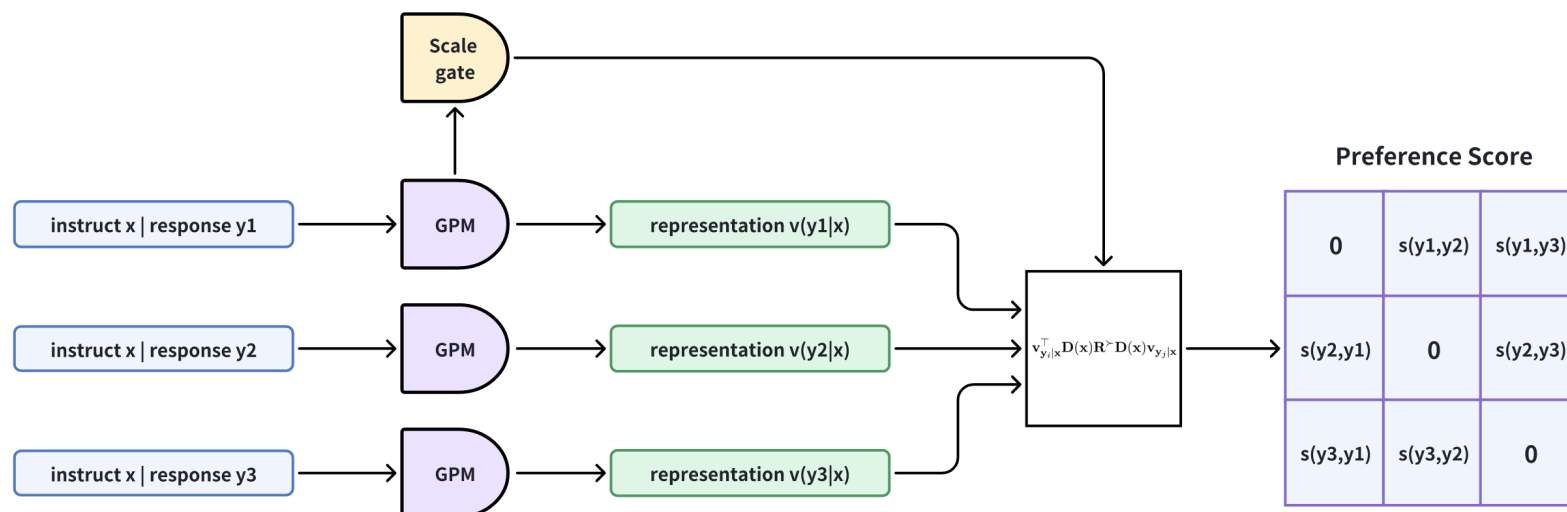
These embeddings capture the nuanced characteristics of the responses relevant to human preferences.

Preference Score. The preference score between two responses is computed as:

$$s(\mathbf{y}_i \succ \mathbf{y}_j | \mathbf{x}) = \mathbf{v}_{\mathbf{y}_i|\mathbf{x}}^\top \mathbf{D}(\mathbf{x}) \mathbf{R}^\succ \mathbf{D}(\mathbf{x}) \mathbf{v}_{\mathbf{y}_j|\mathbf{x}}.$$

where $\mathbf{D}(\mathbf{x})$ is a block-diagonal matrix with blocks $\sqrt{\lambda_l(\mathbf{x})} \mathbf{I}_2$, and \mathbf{R}^\succ is the skew-symmetric operator defined previously. We normalize the embeddings \mathbf{v}_y to have unit length to ensure stability in training.

Implementing GPM as MoE



Automatic Subspace Discovery. The use of multiple dimensions in the embeddings allows the model to discover different subspaces corresponding to various preference dimensions automatically. Each pair of dimensions can capture distinct aspects of preferences, such as helpfulness, correctness, or stylistic elements. The context-dependent eigenvalues $\lambda_l(\mathbf{x})$ modulate the contributions of these subspaces based on the prompt, enabling the model to adapt to varying user preferences dynamically.

This implementation provides a scalable and flexible way to model complex human preferences, making it suitable for large-scale applications in language modeling and other domains where alignment with nuanced human values is essential.

More on Generative Reward Model (GenRM) & PairRM

3 Language Models as Zero-shot Verifiers

$$\text{LM-Score}(\cdot) = \frac{\exp(\text{logit}(\text{'YES'}))}{\exp(\text{logit}(\text{'YES'})) + \exp(\text{logit}(\text{'NO'}))}.$$

```
“<system>
You are ChatGPT, equipped with extensive expertise in
mathematics and coding, and skilled in complex rea-
soning and problem-solving. In the following task, I
will present a text excerpt from a website. Your role
is to evaluate whether this text exhibits mathematical
intelligence and if it is suitable for educational purposes
in mathematics. Please respond with only YES or NO
</system>

User: {
  “url”: “{url}”,
  “text”: “{text}”
}
1. Does the text exhibit elements of mathematical intel-
ligence? Respond with YES or NO
2. Is the text suitable for educational purposes for
YOURSELF in the field of mathematics? Respond with
YES or NO
Assistant: 1.”
```

Figure 3: Illustration of a zero-shot meta-prompt designed for the AutoDS method.

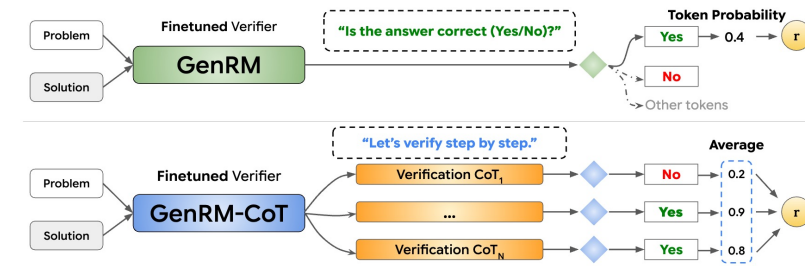


Figure 3 | An illustration of generative verifiers, namely GenRM and GenRM-CoT. Given a question and a candidate solution, GenRM directly finetunes an LLM to answer the question 'Is the answer correct (Yes/No)?' via SFT on the next-token response corresponding to either 'Yes' or 'No'. During inference, the verifier score is obtained by extracting the probability of the 'Yes' token (4). In comparison, GenRM-CoT finetunes a LLM to produce verification chain-of-thought (CoT) rationale before yielding the final Yes/No token. At test-time, we sample multiple CoT rationales and use majority voting to compute the average probability of 'Yes', enabling GenRM-CoT to utilize additional inference-compute for better verification.

More on Generative Reward Model (GenRM) & PairRM

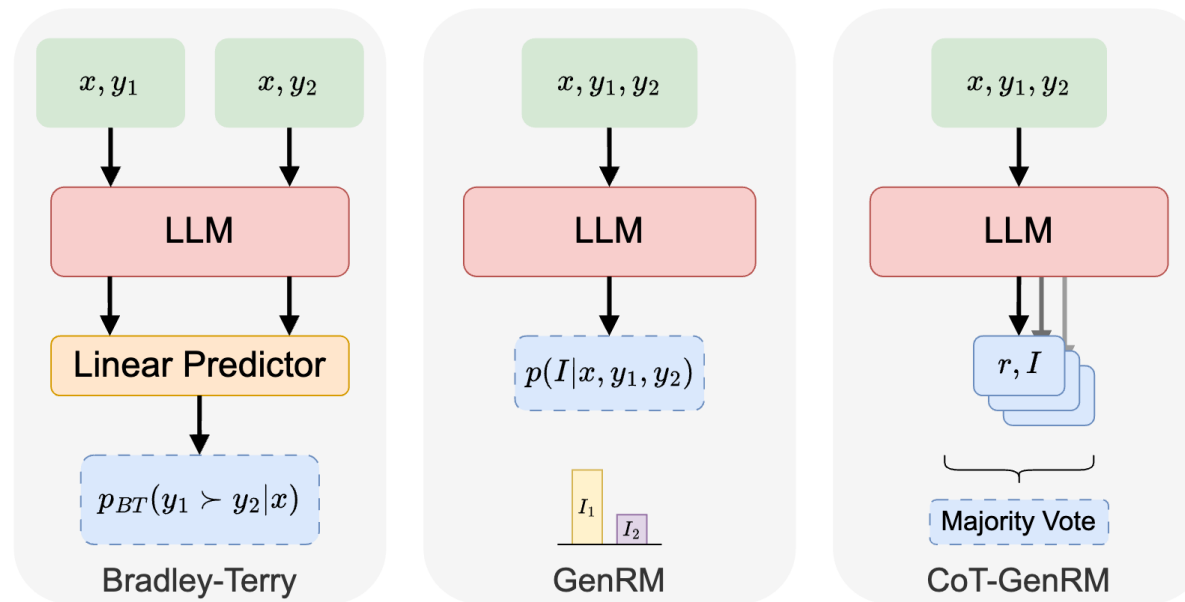


Figure 1: **Methods overview.** *Bradley-Terry* methods directly output the probability of y_1 being preferred over y_2 , while *GenRM* compares the LLMs next-token probabilities of answer indicator tokens (I_1, I_2). *CoT-GenRM* samples reasoning traces (r) followed by the answer indicator token.

Mahan et al., Generative Reward Models. arxiv.org/abs/2410.12832

Outline

- Introduction to RLHF & Reward Modeling
- General Preference Modeling
- **General Preference Optimization**
- Concluding Remarks

Yifan Zhang*, Ge Zhang*, Yue Wu*, Kangping Xu, Quanquan Gu

IIS, Tsinghua University & UCLA



Background on Preference-based RLHF

To address the potential intransitive human preference, the preference-based LLM alignment algorithms ([Munos et al., 2023](#); [Azar et al., 2023](#); [Wu et al., 2024b](#); [Rosset et al., 2024](#)) have been proposed to directly work on the preference pairs instead of assuming a reward function.

Given a preference oracle $\mathbb{P}(\mathbf{y} \succ \mathbf{y}' \mid \mathbf{x})$. The objective is to find a policy π that performs well against another competing policy π' in terms of these preference probabilities. For example, [Azar et al. \(2023\)](#) consider competing with another fixed policy μ (\mathcal{X} denotes the distribution over prompts):

$$\max_{\pi} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} [\mathbb{E}_{\mathbf{y} \sim \pi(\cdot \mid \mathbf{x}), \mathbf{y}' \sim \mu(\cdot \mid \mathbf{x})} [\mathbb{P}(\mathbf{y} \succ \mathbf{y}' \mid \mathbf{x})] - \beta \text{KL}(\pi \parallel \pi_{\text{ref}})], \quad (5.1)$$

Other works ([Munos et al., 2023](#); [Wu et al., 2024b](#); [Rosset et al., 2024](#)) consider solving the two-player constant-sum game:

$$\max_{\pi} \min_{\pi'} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} [\mathbb{E}_{\mathbf{y} \sim \pi(\cdot \mid \mathbf{x}), \mathbf{y}' \sim \pi'(\cdot \mid \mathbf{x})} [\mathbb{P}(\mathbf{y} \succ \mathbf{y}' \mid \mathbf{x})]]. \quad (5.2)$$

To simplify notation, we define the winning probability of a policy π over another policy π' as:

$$\mathbb{P}(\pi \succ \pi' \mid \mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim \pi(\cdot \mid \mathbf{x}), \mathbf{y}' \sim \pi'(\cdot \mid \mathbf{x})} [\mathbb{P}(\mathbf{y} \succ \mathbf{y}' \mid \mathbf{x})]. \quad (5.3)$$

The optimization problem then becomes:

$$\max_{\pi} \min_{\pi'} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} [\mathbb{P}(\pi \succ \pi' \mid \mathbf{x})]. \quad (5.4)$$

Seek a policy that performs well against any opponent in terms of preference probabilities.

The von Neumann winner

The von Neumann winner is a concept drawn from social choice theory ([Sen, 1986](#)) and has been studied in preference-based RL ([Owen, 2013](#); [Dudík et al., 2015](#)). It is the Nash equilibrium of the two-player symmetric game (Equation 5.4). It represents a mixed strategy—a probability distribution over possible responses—that performs optimally in the worst-case scenario against any opponent.

More formally speaking, a distribution π^* is called a von Neumann winner if it satisfies:

$$\min_{\pi' \in \Delta} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} [\mathbb{P}(\pi^* \succ \pi' \mid \mathbf{x})] \geq 1/2.$$

This condition ensures that, on average, the von Neumann winner π^* is at least as likely to be preferred than any other policy π' . The von Neumann winner always exists due to the symmetric nature of the two-player game (Equation 5.4).

- Represents an optimal mixed strategy.
- Guarantees a performance at least equal to any opponent.

Efficient Preference Optimization with General Preference

- Computation Advantage of GPM:
 - Traditional Pairwise models: $O(K^2)$ complexity.
 - GPM computes embeddings once: $O(K)$ complexity.
 - Firstly, we compute v_i for each response y_i .
 - Then, we calculate preference scores using: $s(y_i \succ y_j) = \langle R^\top v_i, v_j \rangle$.
- Result:
 - Efficiently obtain all pairwise preference scores.
 - Suitable for applications requiring scalability.

Policy Optimization with General Preference

Policy Optimization with Preference Score. Once we have a general preference model that outputs the preference score $s(\mathbf{y}_i \succ \mathbf{y}_j | \mathbf{x})$ at hand, we aim to find a policy π that performs well against an opponent policy μ in terms of expected preference scores. The optimization problem is formulated as:

$$\max_{\theta} \mathbb{E}_{\mathbf{x}} \left[\mathbb{E}_{\mathbf{y} \sim \pi_{\theta}(\cdot | \mathbf{x}), \mathbf{y}' \sim \mu(\cdot | \mathbf{x})} [s(\mathbf{y} \succ \mathbf{y}' | \mathbf{x})] \right] - \beta \mathbb{E}_{\mathbf{x}} [\text{KL}(\pi_{\theta}(\cdot | \mathbf{x}) \| \pi_{\text{ref}}(\cdot | \mathbf{x}))], \quad (5.5)$$

where π_{ref} is a reference policy (e.g., the initial language model), μ is the opponent policy (usually the same as π_{ref}), and $\beta > 0$ is a regularization parameter controlling the divergence from the reference policy. We would like to point out that this formulation is different from the many previous works ([Wu et al., 2024b](#); [Swamy et al., 2024](#); [Rosset et al., 2024](#); [Munos et al., 2023](#); [Azar et al., 2023](#)) as they consider maximizing the win rate $\mathbb{P}(\mathbf{y} \succ \mathbf{y}' | \mathbf{x})$, while our formulation is to maximize $s(\mathbf{y} \succ \mathbf{y}' | \mathbf{x}) = \log \frac{\mathbb{P}(\mathbf{y} \succ \mathbf{y}' | \mathbf{x})}{\mathbb{P}(\mathbf{y} \prec \mathbf{y}' | \mathbf{x})}$. Note that $\mathbb{P}(\mathbf{y} \succ \mathbf{y}' | \mathbf{x})$ only varies between 0 and 1, while $s(\mathbf{y} \succ \mathbf{y}' | \mathbf{x})$, similar to the reward $r(\mathbf{y}; \mathbf{x})$ in RLHF or DPO, can take arbitrary values. The flexibility in its value range might benefit fine-tuning.

Generalize reward-based RLHF with PPO seamlessly.

(Ouyang et al. Training language models to follow instructions with human feedback)

General Preference Optimization with Policy Gradients

General Preference Optimization. We consider the SPPO loss used by [Wu et al. \(2024b\)](#) for iterative preference optimization, except that we use preference score instead of preference probability in the loss form. SPPO used K responses for each prompt \mathbf{x} and calculated the empirical win rate of each response \mathbf{y}_k . Instead, we calculate $\hat{s}(\mathbf{y}_i \succ \mu \mid \mathbf{x})$ to estimate the empirical win rate over the distribution μ as below:

$$\hat{s}(\mathbf{y}_i \succ \mu \mid \mathbf{x}) = \frac{1}{K} \sum_{k=1}^K s(\mathbf{y}_i \succ \mathbf{y}_k \mid \mathbf{x}), \forall i \in [K], \quad (5.6)$$

At each iteration t , GPO has the following learning objective:

$$\theta_{t+1} = \arg \min_{\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}, \mathbf{y} \sim \pi_{\theta_t}(\cdot \mid \mathbf{x})} \left[\left(\log \left(\frac{\pi_{\theta}(\mathbf{y} \mid \mathbf{x})}{\pi_{\theta_t}(\mathbf{y} \mid \mathbf{x})} \right) - \frac{1}{\beta} \left(\hat{s}(\mathbf{y} \succ \pi_{\theta_t} \mid \mathbf{x}) - \log Z_{\pi_{\theta_t}}(\mathbf{x}) \right) \right)^2 \right], \quad (5.7)$$

where the normalizing factor $Z_{\pi_{\theta_t}}(\mathbf{x}) := \sum_{\mathbf{y}} \pi_{\theta_t}(\mathbf{y} \mid \mathbf{x}) \exp(\hat{s}(\mathbf{y} \succ \pi_{\theta_t} \mid \mathbf{x}))$.

In practice, we directly replace $\log Z_{\pi_{\theta_t}}(\mathbf{x})$ with 0^1 . Intuitively, if a response \mathbf{y} receives a high average score, GPO will increase its log probability. We report the empirical performance of GPO in [Section 6.3](#).

General Preference Optimization with Policy Gradients

Connection to Policy Gradient. Applying policy gradient theorem on Equation (5.5) gives:

$$\begin{aligned} & \nabla_{\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}, \mathbf{y} \sim \pi_{\theta}} \left[\widehat{s}(\mathbf{y} \succ \pi_{\theta_t}) - \beta \log \frac{\pi_{\theta}(\mathbf{y}|\mathbf{x})}{\pi_{\theta_t}(\mathbf{y}|\mathbf{x})} \right] \\ &= \mathbb{E}_{\mathbf{x} \sim \mathcal{X}, \mathbf{y} \sim \pi_{\theta}} \left[\left(\widehat{s}(\mathbf{y} \succ \pi_{\theta_t}) - \beta \log \frac{\pi_{\theta}(\mathbf{y}|\mathbf{x})}{\pi_{\theta_t}(\mathbf{y}|\mathbf{x})} \right) \nabla_{\theta} \log \pi_{\theta}(\mathbf{y}|\mathbf{x}) \right] \\ &= \mathbb{E}_{\mathbf{x} \sim \mathcal{X}, \mathbf{y} \sim \pi_{\theta}} \left[- \nabla_{\theta} \left(\widehat{s}(\mathbf{y} \succ \pi_{\theta_t}) - \beta \log \frac{\pi_{\theta}(\mathbf{y}|\mathbf{x})}{\pi_{\theta_t}(\mathbf{y}|\mathbf{x})} \right)^2 \right]. \end{aligned}$$

So Equation (5.7) can also be seen as an offline policy gradient method for the optimization problem (5.5).

Remark 5.2. Note that the general preference score given by our GPM in Equation (5.5) can also be integrated as preference (reward) signal for any off-the-shelf RLHF and preference optimization methods, including (iterative) DPO ([Rafailov et al., 2024](#)), IPO ([Azar et al., 2023](#)), NLHF ([Munos et al., 2023](#)), SPPO ([Wu et al., 2024b](#)) and REBEL ([Gao et al., 2024](#)), as well as PPO-based methods ([Ouyang et al., 2022](#)) by directly optimizing Equation (5.5).

Experimental Results

We conducted extensive experiments to evaluate the effectiveness of the proposed General Preference representation model (GPM) in comparison to traditional reward-based models, particularly focusing on its ability to model cyclic preferences and improve language model alignment. Our experiments are designed to address the following key questions:

- **Q1:** Can the GPM effectively capture and model cyclic and intransitive preferences, where traditional models like the Bradley-Terry (BT) reward model struggle?
- **Q2:** How does the GPM perform on standard preference modeling benchmarks (RewardBench) compared to the BT model?
- **Q3:** How does using the GPM for downstream policy optimization impact language model performance on real-world tasks compared to reward-based approaches?

Cyclic Preference Modeling

To address **Q1**, we evaluate the ability of the GPM to capture intransitive, cyclic preferences that traditional transitive models (like the BT model) struggle to represent.

Cyclic Preference Dataset. We constructed a dataset by inducing cyclic preferences from the Ultrafeedback dataset [Cui et al. \(2024\)](#). The dataset includes responses evaluated across four key metrics: *instruction following*, *honesty*, *truthfulness*, and *helpfulness*. We created preference cycles such as: `instruction following` \succ `honesty` \succ `truthfulness` \succ `helpfulness` \succ `instruction following`, ensuring the presence of intransitive cycles. We further generated four sub-datasets by omitting one metric from each cycle, resulting in datasets of varying complexity with 216 to 363 instances.

Table 1: Comparison of Bradley-Terry (BT) reward model and General Preference representation models (GPM) on cyclic preference datasets.

Model	Dataset	Acc. (%)
Random Guess		50.0
BT RM	No <code>instruction following</code>	62.4
GPM	No <code>instruction following</code>	100.0 (+37.6)
BT RM	No <code>honesty</code>	61.6
GPM	No <code>honesty</code>	100.0 (+38.4)
BT RM	No <code>truthfulness</code>	50.0
GPM	No <code>truthfulness</code>	100.0 (+50.0)
BT RM	No <code>helpfulness</code>	62.9
GPM	No <code>helpfulness</code>	100.0 (+37.1)

RewardBench Evaluation

- RewardBench covers diverse tasks: Chat, Chat-Hard, Safety, Reasoning.

Table 2: Comparison between the Bradley-Terry (BT) models and the General Preference representation models (GPM) with varying embedding head dimensions on RewardBench. The highest scores are in bold and the second highest are underlined.

Model	Embed Dim.	Chat	Chat-Hard	Safety	Reasoning	Average
Base Model: Gemma-2B-it						
BT RM	1	58.4	62.9	82.9	71.5	68.9
GPM	2	<u>68.4</u>	66.7	70.7	<u>80.3</u>	71.5
	4	<u>63.1</u>	<u>68.4</u>	72.4	81.0	71.2
	6	68.2	66.4	80.6	76.7	<u>73.0</u>
	8	71.5	69.7	<u>81.3</u>	75.6	74.5 (+5.6)
Base Model: Llama-3.1-8B-Instruct						
BT RM	1	89.7	<u>87.3</u>	91.0	<u>96.2</u>	91.1
GPM	2	92.7	<u>87.3</u>	90.5	96.1	91.7
	4	93.6	87.1	90.4	96.0	<u>91.8</u>
	6	<u>93.3</u>	88.6	<u>90.6</u>	96.0	92.1 (+1.0)
	8	92.7	86.4	90.4	96.7	91.6

Language Model Alignment

- Evaluation Benchmarks:
 - **AlpacaEval 2.0**: Measures alignment with human preferences.
 - **MT-Bench**: Evaluates model performance on multi-turn dialogues.
- Applied GPO & SPPO to align language models using preference scores from GPM.

Language Model Alignment

Table 3: AlpacaEval 2.0 evaluation results. Base model: LLama3-8B-it, Evaluator: GPT-4o-mini. The results are grouped by the size and type of the RM or PM, and the number of iterations.

Size	Type	Iter	SPPO		GPO	
			Win Rate	Avg. Len	Win Rate	Avg. Len
		base	32.26	1959	32.26	1959
2B	BT RM	1	44.48	1843	49.55	1837
		2	59.89	2028	57.19	2048
		3	71.11	2244	67.81	2245
	GPM	1	52.64 (+8.16)	2102	58.49 (+8.94)	2145
		2	66.01 (+6.12)	2318	72.22 (+15.03)	2404
		3	72.97 (+1.86)	2490	77.11 (+9.30)	2613
8B	BT RM	1	40.96	1802	42.56	1764
		2	52.43	1945	57.19	2080
		3	55.30	1832	62.79	3445
	GPM	1	45.80 (+4.84)	1878	49.55 (+6.99)	1877
		2	56.04 (+3.61)	2020	57.54 (+0.35)	2126
		3	60.73 (+5.43)	1994	66.23 (+3.44)	2157

Table 4: AlpacaEval 2.0 evaluation results. Base model: LLama3-8B-it, Evaluator: GPT-4-turbo. The results are grouped by the size and type of the RM or PM, and the number of iterations.

Size	Type	Iter	SPPO			GPO		
			LC. WR	WR	Avg. Len	LC. WR	WR	Avg. Len
		base	23.07	23.34	1959	23.07	23.34	1959
2B	BT RM	1	33.71	30.88	1843	36.28	33.17	1837
		2	37.38	37.88	2028	37.87	38.89	2048
		3	38.86	42.98	2244	38.79	42.80	2245
	GPM	1	32.64	35.12	2102	35.23	38.10	2145
		2	35.16	41.40	2318	36.04	44.47	2404
		3	35.30	45.44	2490	38.51	48.30	2613
8B	BT RM	1	33.38	29.90	1802	35.96	31.71	1764
		2	38.83	37.82	1945	41.48	41.73	2080
		3	40.55	37.09	1832	42.16	43.33	3445
	GPM	1	33.39	31.49	1878	37.45	35.17	1877
		2	37.35	37.60	2020	39.07	39.94	2126
		3	39.72	39.38	1994	41.19	43.38	2157

Language Model Alignment

Table 5: AlpacaEval 2.0 evaluation results. Base model: LLama3-8B-it, Evaluator: DeepSeek-V2. The results are grouped by the size and type of the RM or PM, and the number of iterations.

Size	Type	Iter	SPPO		GPO	
			Win Rate	Avg. Len	Win Rate	Avg. Len
		base	30.20	1959	30.20	1959
2B	BT	1	43.00	1843	43.78	1837
		2	51.78	2028	52.80	2048
		3	60.99	2244	58.21	2245
	GP	1	48.41 (+5.41)	2102	52.57 (+8.79)	2145
		2	58.12 (+6.34)	2318	62.52 (+9.72)	2404
		3	65.26 (+4.27)	2490	68.28 (+10.07)	2613
8B	BT	1	41.73	1802	43.36	1764
		2	50.24	1945	52.90	2080
		3	50.76	1832	56.59	3445
	GP	1	45.42 (+3.69)	1878	46.89 (+3.53)	1877
		2	52.77 (+2.53)	2020	53.03 (+0.13)	2126
		3	54.73 (+3.97)	1994	59.43 (+2.84)	2157

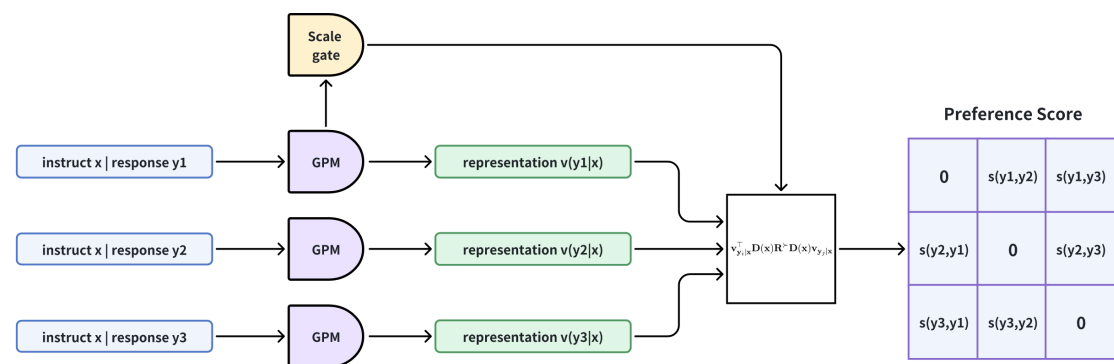
Table 6: MT-Bench evaluation results. Base model: LLama3-8B-it, Evaluator: GPT-4.

Size	Type	Iter	SPPO			GPO		
			1st	2nd	Avg.	1st	2nd	Avg.
		base	8.27	7.38	7.82	8.27	7.38	7.82
2B	BT RM	1	8.11	7.44	7.77	8.15	7.37	7.77
		2	8.37	7.59	7.98	8.20	7.69	7.95
		3	8.26	7.36	7.81	8.59	7.54	8.06
	GPM	1	8.26	7.66	7.96	8.49	7.50	7.99
		2	8.26	7.66	7.96	8.38	7.55	7.97
		3	8.31	7.53	7.92	8.63	7.89	8.26
8B	BT RM	1	8.53	7.85	8.19	8.54	8.05	8.29
		2	8.33	7.91	8.12	8.16	7.55	7.85
		3	8.05	7.66	7.86	8.12	7.76	7.94
	GPM	1	8.27	7.60	7.93	8.30	7.70	8.00
		2	8.46	7.71	8.08	8.11	7.65	7.88
		3	8.14	7.59	7.87	8.22	7.63	7.93

Ablation Studies on GPM Implementations

Table 7: Impact of the embedding head and the scale gate on the GPM’s performance on Reward-Bench. Dim. represents the dimension of the embedding head. The highest average scores for each base model are in bold and the second highest are underlined.

Embedding Type	Dim.	Chat	Chat-Hard	Safety	Reasoning	Average
Base Model: Gemma-2B-it						
w. scale gate w. l2	2	68.4	66.7	70.7	80.3	71.5
w. scale gate w.o. l2	2	70.7	66.4	72.8	71.4	<u>70.3</u>
w. o. scale gate w. l2	2	68.4	62.9	70.3	72.5	68.5
w. o. scale gate w.o. l2	2	67.0	69.5	70.2	68.1	68.7
w. scale gate w. l2	4	63.1	68.4	72.4	81.0	71.2
w. scale gate w.o. l2	4	69.3	71.7	67.1	73.8	<u>70.5</u>
w. o. scale gate w. l2	4	71.2	60.7	66.6	76.6	68.8
w. o. scale gate w.o. l2	4	66.8	67.8	71.7	69.1	68.9
w. scale gate w. l2	6	68.2	66.4	80.6	76.7	73.0
w. scale gate w.o. l2	6	66.8	62.9	68.5	62.8	65.2
w. o. scale gate w. l2	6	65.4	65.1	66.9	63.6	65.3
w. o. scale gate w.o. l2	6	69.8	71.1	73.3	74.9	<u>72.3</u>
w. scale gate w. l2	8	71.5	69.7	81.3	75.6	74.5
w. scale gate w.o. l2	8	69.8	66.9	78.9	75.9	<u>72.9</u>
w. o. scale gate w. l2	8	67.3	66.2	62.9	70.8	66.8
w. o. scale gate w.o. l2	8	69.6	67.8	76.1	70.4	71.0



Outline

- Introduction to RLHF & Reward Modeling
- General Preference Modeling
- General Preference Optimization
- **Concluding Remarks**

Yifan Zhang*, Ge Zhang*, Yue Wu*, Kangping Xu, Quanquan Gu

IIS, Tsinghua University & UCLA



Summary & Takeaways

- **General Preference Representation Learning (GPM):**
 - We introduced a novel framework for modeling general preferences.
 - Bridged the gap between expressiveness and efficiency.
 - Showed that GPM outperforms traditional reward models such as BT model.
- **General Preference Optimization (GPO):**
 - We generalized reward-based RLHF to preference scores based RLHF seamlessly.
 - We proposed a new (iterative & self-play) preference optimization method GPO using preference scores.
 - Demonstrated its effectiveness in aligning language models.

Implications & Outlook

- **Enhanced Alignment:**

- Our method improves the alignment of language models with nuanced human values.

- **Scalability:**

- Efficient computation enables application to large-scale models and datasets.

- **Future Work:**

- Investigate integration with other alignment techniques (PPO, DPO, IPO, etc.).
- Explore extensions to multi-modal preference learning.

Thank you for your attention!

Q & A

Thank you for your attention!