REWARDBENCH 2: Advancing Reward Model Evaluation

Saumya Malik[α] **Valentina Pyatkin**[αβ] **Sander Land**[δ] **Jacob Morrison**[α]

Noah A. Smith[αβ] **Hannaneh Hajishirzi**[αβ] **Nathan Lambert**[α]

[α]Allen Institute for Artificial Intelligence
[β]University of Washington [δ]Cohere

contact: saumyam@allenai.org

Abstract

Reward models are used throughout the post-training of language models to capture nuanced signals from preference data and provide a training target for optimization across instruction following, reasoning, safety, and more domains. The community has begun establishing best practices for evaluating reward models, from the development of benchmarks that test capabilities in specific skill areas to others that test agreement with human preferences. At the same time, progress in evaluation has not been mirrored by the effectiveness of reward models in downstream tasks – simpler direct alignment algorithms are reported to work better in many cases. This paper introduces REWARDBENCH 2, a new multi-skill reward modeling benchmark designed to bring new, challenging data for accuracy-based reward model evaluation – models score about 20 points on average lower on REWARDBENCH 2 compared to the first REWARDBENCH – while being highly correlated with downstream performance. Compared to most other benchmarks, REWARDBENCH 2 sources new human prompts instead of existing prompts from downstream evaluations, facilitating more rigorous evaluation practices. In this paper, we describe our benchmark construction process and report how existing models perform on it, while quantifying how performance on the benchmark correlates with downstream use of the models in both inference-time scaling algorithms, like best-of-N sampling, and RLHF training algorithms like proximal policy optimization.

1 Introduction

Reward Models (RMs) are often designed to model human preferences to improve language model training [1, 2, 3, 4]. Generally, a reward model is trained to output a scalar value proportional to (some aspects of) the quality of the input text, learned from preference data. RMs have been used extensively for RLHF training [5, 6], but also are used for online direct alignment algorithms [7], data filtering [8, 4], and inference-time scaling [9, 10]. Despite extensive use, the ecosystem of directly evaluating reward models is still nascent and developing alongside the roles RMs play.

Users developing RMs for their application must decide which benchmark(s) to use. This is a multi-dimensional decision process, as evaluations vary in how they measure performance (e.g., accuracy vs. correlation with LM-as-a-judge) and the domains they focus on (e.g., multi-skill vs. chat-only). The first reward model evaluations such as REWARDBENCH [11] and RM-Bench [12] focused on simple classification tasks to measure performance of existing reward models across common domains like style and safety. Additional evaluations included analysis of downstream scores when the RM is used within inference-time methods such as best-of-N (BoN) sampling [13] and also training with RLHF [14].

Preprint.

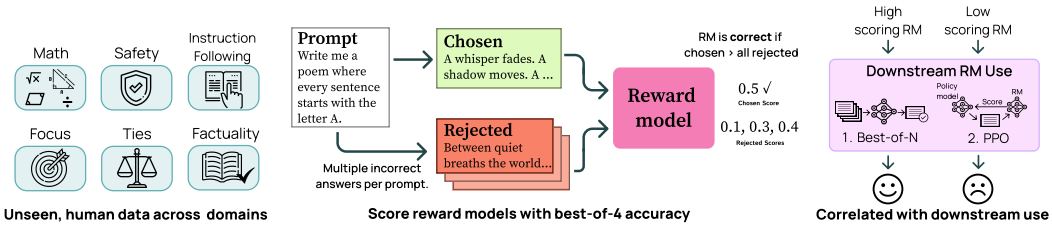


Figure 1: REWARDBENCH 2 is composed of high-quality, unseen human prompts designed for a best-of-4 reward model evaluation format with completions generated from a variety of leading AI models. We extend RM evaluation of pairwise "chosen" and "rejected" completions to include additional rejected samples as distractions. REWARDBENCH 2 has 6 domains which expand upon challenging domains in existing RM evaluations and adds a new domain, Ties, to test how RMs handle questions with multiple correct answers. The new data and setup enables more accurate correlation of benchmark scores with downstream performance via RL finetuning or best-of-N sampling.

We present REWARDBENCH 2, a benchmark built on classification tasks that measures and improves correlations relative to earlier approaches of RM evaluations in two scenarios: inference-time compute and downstream training (highlighted in Figure 1). Our benchmark maintains strengths of multiple existing benchmarks, such as using unseen human prompts or switching from the common practice of accuracy over a chosen and rejected response to one chosen and three rejected responses to reduce the distance between strong reward models and the random baseline, as summarized in Table 1. The benchmark covers six domains: three new datasets to improve evaluation in domains covered by existing RM benchmarks – focus, math, and safety – along with three new challenging domains: factuality, precise instruction following, and ties (a new type of domain where we test a RM's ability to be well-calibrated between equivalently valid answers, like "red" and "green" in response to "Name a color of the rainbow"). In total we evaluate over 100 reward models, a mix of leading existing models and new models we trained to better understand the relationship between RM training and evaluation, in order to allow more reliable use of RMs across a variety of skills often targeted in post-training in order to allow more reliable use of reward models.

The benchmark was created with a majority of previously unused human prompts from the WildChat pipeline [15] with extensive manual, programmatic, and LM-based filtering techniques. To validate the benchmark, we run extensive experiments to show how RM benchmarks can be used in effective RLHF training workflows or correlated hillclimbing targets for inference-time compute techniques. Our contributions and findings are as follows:[1]

1. REWARDBENCH 2 provides a **challenging evaluation of reward models across many domains on majority unseen prompts** with leading models on the first version of REWARDBENCH scoring 20 or more points lower on REWARDBENCH 2. This includes challenging subsets such as Precise Instruction Following and Math where leading models are below 40% and 70% accuracy, respectively with data details discussed in Section 3.
2. Controlled experiments where we train reward models and analyze their performance on the benchmark, **gaining actionable insights for reward model training**. In particular, we find that different post-trained base models, even within the same lineage and model family, offer different capabilities to reward models and that, contrary to the accepted best practice, training for more than one epoch can be beneficial. We discuss these findings in Section 4.
3. An exploration of the benefits and limits of using a reward model evaluation to inform downstream use cases of inference-time scaling algorithms and RLHF training. In Section 5 **our benchmark achieves strong downstream correlation with inference time scaling algorithms like best-of-N sampling** and provides a helpful signal for PPO training.
4. Our analysis shows how the best reward model for RLHF is dependent on one's training setup. **For RLHF, the reward model should be based on a model of the same lineage as the policy model or else downstream performance can degrade significantly, so simply taking the highest scoring reward model on a benchmark will not ensure a good post RLHF model.**

[1]Code for REWARDBENCH 2 is available here: `https://github.com/allenai/reward-bench`. The benchmark dataset is available here: `https://huggingface.co/datasets/allenai/reward-bench-2`.

Table 1: A comparison of REWARDBENCH 2 relative to existing reward modeling benchmarks. For metrics, 🎯 is used to denote an accuracy metric (correctness) and ⚖ is used where the metric is either human or LM-as-a-judge agreement. Comparing the relative correlation of each RM benchmark with downstream tasks is challenging because the correlation depends on the downstream tasks of choice. * denotes benchmarks meant to test one specific attribute (e.g., typos, multilinguality).

RM Evaluation	Best-of-N (N > 2)	Human Prompts	Unseen Prompts	Metric	Multi Skill
RewardBench [11]	✗	✗	✗	🎯	✓
RewardMATH [18]	✓	✗	✗	🎯	✗
RM-Bench [12]	✗	✗	✗	🎯	✓
*ReWordBench [19]	✗	✗	✗	🎯	✓
*M-RewardBench [20]	✗	✗	✗	🎯	✓
PPE [14] – Correctness	✓	✗	✗	🎯	✓
PPE [14] – Human Pref.	✗	✓	✓	⚖	✗
RMB [13]	✓	✓	✗	⚖	✓
REWARDBENCH 2	✓	✓	✓	🎯	✓

2 Background

Reward Models Reward models are trained on preference data, consisting of prompts x and completions y_i, where each completion has been ranked by humans or automated metrics like ground truth signals and language model judgments [16]. The canonical formulation, which we use in this work, is to create preference *pairs*, where for each prompt two completions are compared, and the better prompt is "chosen", and the other is "rejected." With that data, a reward model r^* is trained to output a scalar value to predict the probability p^* of a prompt and completion falling in the chosen category, following a Bradley-Terry model of human preferences [17]:

$$p^*(y_1 \succ y_x \mid x) = \frac{\exp(r^*(x, y_1))}{\exp(r^*(x, y_1)) + \exp(r^*(x, y_2))}. \tag{1}$$

The Bradley-Terry formulation of preference is fit through maximum likelihood estimation:

$$\mathcal{L}(\theta, \mathcal{D}) = \mathbb{E}_{(x, y_{\text{chosen}}, y_{\text{rejected}}) \sim \mathcal{D}} \left[\log(1 + e^{r_\theta(x, y_{\text{rejected}}) - r_\theta(x, y_{\text{chosen}})}) \right].$$

For more information on how reward models are used, such as in reinforcement learning from human feedback (RLHF) and best-of-N (BoN) sampling, see Appendix A.

Reward Model Benchmarking Reward model evaluation has expanded to be similar to the types of evaluations available to general post-trained models, where some evaluations test the accuracy of prediction on domains with known true answers [11] while others measure preferences (colloquially referred to as "vibes") performed with LM-as-a-judge or correlations to other benchmarks [21]. Recent reward model benchmarks fall into three categories: (1) Benchmarks focusing on general downstream performance, continuing from REWARDBENCH, include Preference Proxy Evaluations [14], RMB [13], and RM-Bench [12]. (2) Specific new attributes to test include multilinguality [20], agentic systems (e.g., web agents [22] or retrieval augmented generation [23]), typos [19], and others [18]. (3) Benchmarks testing different modalities or structures of reward modeling include those for multimodal [24, 25, 26, 27], process reward [28], or visual process reward [29, 30] models.

We compare REWARDBENCH 2 to recent text-only reward model benchmarks listed in Table 1. We highlight the importance of REWARDBENCH 2 using unseen human prompts, a departure from most prior work that repurposes prompts from widely-used downstream evaluations to evaluate reward models. Without entirely new prompts, claims of correlations to downstream benchmarks must overcome the potential of contamination with respect to the downstream evaluation target. Additionally, while benchmarks whose chosen-rejected splits are determined by human or LM pairwise preferences have some benefits, there is subjectivity in the preferences they prescribe as optimization targets [31, 32]. With the focus of REWARDBENCH 2 on downstream skills, we opt to use accuracy-based tests in our benchmark.

Table 2: REWARDBENCH 2 domains and their various specific construction decisions. We prioritized using new human prompts with robust subset-specific completion generation and verification pipelines. In total there are 1,865 prompts and completions from 20 different models (see Appendix F for a list of models) or human-written completions. Prompts sourced "manually" denote those created by the authors, while "human" denotes those collected from in-the-wild chat interactions. Focus does not need filtering because it is created with specific prompting that differentiates the chosen and rejected completions (followed by manual verification of the *method* rather than every instance).

Domain	Count	Prompt Source	Method of generating completions	Completion Filtering
Factuality	475	Human	Both	Multi-LM-as-a-judge
Precise IF	160	Human	Natural	Verifier functions
Math	183	Human	Natural	Majority voting
Safety	450	CoCoNot [35]	Both	LM-as-a-judge & rubrics
Focus	495	Human	System Prompt Variation	N/A
Ties	102	Manual	System Prompt Variation	Manual verification

3 Building the Benchmark and Measuring Performance

In this section, we detail the data curation and scoring methods used for REWARDBENCH 2 that enable a challenging, accuracy-based benchmark correlated with downstream post-training evaluations. This involves four stages: prompt sourcing, where most of our prompts are unreleased human-written queries from WildChat [15]; prompt quality and domain annotation using classifiers; completion generation, where we aim for diversity while ensuring we construct both "right" and "wrong" completions; and filtering, where we verify that prompts and completions fit each domain's criteria. We release our code under the Apache 2.0 license and release the benchmark data under ODC-By.

Prompt Sourcing We focused on getting representative, unseen prompts from real usage of language models and pairing them with completions representative of the current spectrum of language modeling performance. The goal is to make reward model evaluation prompts independent from evaluations used to test downstream post-trained models. Prompts denoted as "Human" in Table 2 are unseen and reflect real world use of AI models ($\sim$70% of the benchmark). From a pool of prompts, we filtered and assigned prompts to our domain-specific subsets using a combination of QuRater [33] to annotate data, a topic classifier [2] to identify prompt domain, and manual inspection. We compared our prompts against twenty widely-used downstream evaluations with the Tulu 3 decontamination toolkit [34] and ensured there was no overlap. To arrive at our final dataset, we first created an initial set of around 3,000 total high-quality prompts in our target domains, and then curated the final 1,876 prompts through further manual verification and filtering.

Constructing REWARDBENCH 2's Domains An overview of the 6 domains in REWARDBENCH 2 and how they were created is detailed in Table 2. The Math, Safety, and Focus domains are new datasets inspired by improving upon the Math, Safety, and Chat-Hard domains of the original REWARDBENCH [11], respectively, whereas Factuality, Precise IF, and Ties are designed to test additional capabilities of RMs not captured in REWARDBENCH. In summary, the subsets of REWARDBENCH 2 are as follows, with additional dataset creation details in Appendix E:

1. **Factuality**: Tests the ability of RMs to detect hallucinations and other basic errors in completions. To construct this subset, we sampled both natural completions as well as completions from an added system prompt instructing the model to make subtle factual errors. We classify these responses as "accurate" or "inaccurate" by prompting two LLMs to judge their accuracy independently, and assigning a label only if both LLMs agree ("accurate" responses go into the chosen category and "inaccurate" build rejected completions).
2. **Precise Instruction Following**: Tests the ability of RMs to judge whether text follows precise instructions, such as "Answer without the letter u". We append a constraint taken from the taxonomy of a new instruction-following benchmark, IFEval-OOD [34], to each prompt, manually ensuring relevance (more details in Appendix E.2) We use verifier functions to evaluate adherence to the constraint, and constructed each data instance by combining 1 completion that satisfies

[2]Prompt domain classifier: https://huggingface.co/valpy/prompt-classification

the constraint and 3 that do not, taking steps to ensure that adherence to the constraint did not otherwise compromise the quality of response.

3. **Math**: Tests RMs' abilities at math, on open-ended human prompts ranging from middle school physics and geometry to college-level chemistry, calculus, combinatorics, and more. To grade completions, we used majority voting to populate a candidate set of prompts with 1 correct and 3 incorrect prompts and then manually verified every sample in this domain due to the brittle nature of answer extraction.
4. **Safety**: Tests RMs' abilities to correctly comply with or refuse prompts related to harmful use cases. Safety is a nuanced task for LMs, so we draw on recent work on compliance over a variety of domains, CoCoNot [35], while taking steps to make the benchmark conservative in areas where user disagreements may exist on what a model *should* do. We modify their taxonomy, subset-specific rubrics for judging compliance with GPT-4o, and test prompts for generating and evaluating completions from our model pool. We combine one correctly noncompliant response with three compliant responses for each instance.
5. **Focus**: Tests RMs' ability to detect high-quality, on-topic answers to general user queries (e.g. writing generation or question answering). We follow LLMBar [36] (and the Chat-Hard subdomain of the first REWARDBENCH) and rewrite human prompts using a language model to introduce slight differences, which then induce objectively incorrect, off-topic, and/or generally unresponsive "rejected" completions that are misaligned in some way with the original prompt. We combine one natural completion with three such off-topic completions for each datapoint.
6. **Ties**: This new type of subset called *Ties* tests the robustness of RMs in domains with many possible similar answers. For example, the question "Name a color of the rainbow" has seven possible correct answers and infinitely many incorrect ones. These questions evaluate whether a reward model avoids expressing overly strong or arbitrary preferences among equivalent correct answers, while still clearly preferring any correct answer over any incorrect one. Samples were created manually with assistance from AI models.

Scoring REWARDBENCH 2 The primary scoring metric for REWARDBENCH 2 is accuracy, which is used for all subsets except ties, whose scoring metric is described next. Scores are first measured per-domain, and the final score is an unweighted average across all six domains. Accuracy on REWARDBENCH 2 is judged by selecting the correct response from 4 completions per prompt. There is only one correct chosen response, meaning the random baseline is 25% accuracy, versus 50% for many related works with only 2 completions per prompt. A lower random baseline is helpful in creating a benchmark with more headroom for hillclimbing on and providing more robustness of scores that could be near said random baseline, especially for more challenging subsets.

The 'Ties' subset score is a weighted score of accuracy (as measured by *all* valid correct answers being scored higher than *all* incorrect answers) and whether the reward margin between correct and incorrect answers exceeds that of the highest and lowest-scored correct responses. This metric rewards not only correctness, but also a model's ability to prioritize correct answers over incorrect ones more strongly than it distinguishes between equally valid correct responses.

4 Analysis of Performance on REWARDBENCH 2

In this section, we analyze the performance of reward models on REWARDBENCH 2, looking at both existing RMs and new RMs that we trained.

Existing Reward Models REWARDBENCH 2 is a challenging benchmark for top reward models, shown in Table 3 for the top twenty existing models where they are particularly challenged by the Instruction Following, Math, and Factuality subsets. We evaluate generative models with two prompting strategies—prompting them to pick the best among four options and prompting them to provide absolute ratings to an individual option—and report the better setting for each model. See Appendix G for more details.

Figure 2 shows the drops in performance of these top existing models relative to performance on the first RM benchmark, REWARDBENCH [11], along with our own newly trained models. We did *not* tune the development of REWARDBENCH 2 to our trained models, as the models were tuned for downstream performance or open-ended exploration. The scores on both benchmarks are less

Table 3: Top models on REWARDBENCH 2. The benchmark is challenging for even top existing reward models, with room for improvement in several domains. * denotes LM-as-a-judge models.

	Average	Factuality	IF	Math	Safety	Focus	Ties
google/gemini-2.5-flash-preview-04-17*	77.2	65.7	55.3	81.1	90.9	86.7	83.4
nicolinho/QRM-Gemma-2-27B	76.7	78.5	37.2	69.9	95.8	95.4	83.2
infly/INF-ORM-Llama3.1-70B	76.5	74.1	41.9	69.9	96.4	90.3	86.2
anthropic/claude-opus-4-20250514*	76.5	82.7	41.9	74.9	89.5	86.2	83.7
allenai/Llama-3.1-70B-Instruct-RM-RB2	76.1	81.3	41.9	69.9	88.4	86.5	88.3
Skywork/Skywork-Reward-Gemma-2-27B	75.8	73.7	40.3	70.5	94.2	93.2	82.6
anthropic/claude-3-7-sonnet-20250219*	75.4	73.3	54.4	75.0	90.3	92.1	67.2
Skywork/Skywork-Reward-Gemma-2-27B-v0.2	75.3	76.7	37.5	67.2	96.9	91.7	81.8
LxzGordon/URM-LLaMa-3.1-8B	73.9	68.8	45.0	63.9	91.8	97.6	76.5
Skywork/Skywork-Reward-Llama-3.1-8B	73.1	69.9	42.5	62.8	93.3	96.2	74.1
allenai/Llama-3.1-8B-Instruct-RM-RB2	72.8	74.3	44.4	61.7	89.6	90.7	76.4
ShikaiChen/LDL-Reward-Gemma-2-27B-v0.1	72.5	75.6	35.0	64.5	92.2	91.3	76.3
openai/gpt-4.1-2025-04-14*	72.3	82.9	39.7	65.2	87.3	73.4	85.4
allenai/Llama-3.1-Tulu-3-70B-SFT-RM-RB2	72.2	80.8	36.9	67.8	86.9	77.8	83.1
Skywork/Skywork-Reward-Llama-3.1-8B-v0.2	71.7	69.7	40.6	60.1	94.2	94.1	71.7
anthropic/claude-sonnet-4-20250514*	71.2	76.1	35.9	70.5	89.1	76.0	79.4
nicolinho/QRM-Llama3.1-8B-v2	70.7	66.5	40.6	61.2	94.7	89.1	72.3
allenai/Llama-3.1-Tulu-3-8B-RL-RM-RB2	68.7	76.4	40.0	61.7	86.4	84.8	62.8
allenai/Llama-3.1-Tulu-3-8B-DPO-RM-RB2	68.7	75.2	38.8	62.8	86.0	85.5	64.0
allenai/Llama-3.1-Tulu-3-8B-SFT-RM-RB2	68.2	73.3	38.8	57.9	89.8	88.9	60.6
google/gemini-2.5-pro-preview-05-06*	67.7	65.3	46.9	53.4	88.1	83.1	69.7
Ray2333/GRM-Llama3-8B-rewardmodel-ft	67.7	62.7	35.0	58.5	92.2	89.3	68.2
RLHFlow/ArmoRM-Llama3-8B-v0.1	66.5	65.7	41.9	66.1	82.2	76.6	66.3
openai/gpt-4.1-mini-2025-04-14*	65.7	60.8	41.2	72.1	72.6	73.5	74.0
openai/gpt-4o-2024-08-06*	64.9	56.8	33.1	62.3	86.2	72.9	78.2

correlated for external models than our trained models, indicating a potential of metric capture to version 1.

Newly Trained Reward Models [3] To analyze the performance of a larger variety of reward models than currently exists in the literature on our benchmark, we also trained our own Bradley-Terry reward models in a controlled setup, using the Open Instruct library [37][4]. We varied (1) hyperparameters like learning rate and number of training epochs, exploring values common in the literature; (2) the base model, examining multiple strong open-weight models that many existing RMs are trained on; and (3) training data, looking at two preference data mixtures with demonstrated success in post-training (Tulu preference mix [34]) and reward model training (Skywork preference mix [38]). Appendix B has further training details.

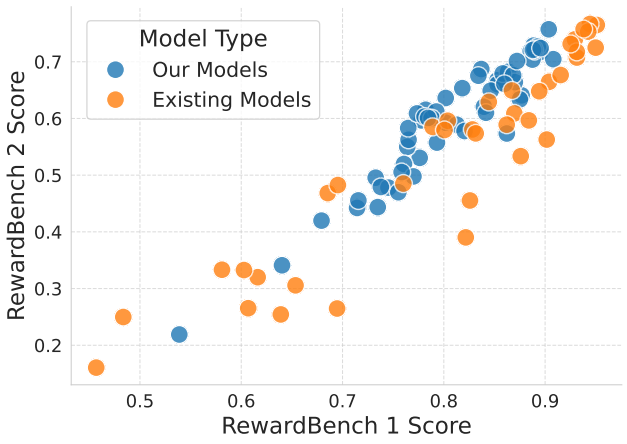


Figure 2: Scores on REWARDBENCH 2 are much lower than scores on REWARDBENCH 1.

[3]The reward models we trained will be released upon paper acceptance.

[4]Open-Instruct on GitHub: `https://github.com/allenai/open-instruct`

In this section, we take a closer look at the performance of our new trained reward models on REWARDBENCH 2. Table 4 in the Appendix shows the breakdown of scores for the top model (across hyperparameters and seeds) for each unique combination of base model and training data. We observe the following:

1. Overall, **Llama 3.1 Instruct-based models are strong** in our setup, both at the 8B and 70B scale. We additionally see that larger reward models perform better on the benchmark; this is to be expected, as their base models are stronger.
2. **Different domains benefit from different training data sources.** For example, we see that the Skywork data is particularly helpful for focus and safety, while the Tulu data is better for factuality. **Combining both data sources improves average performance**, outperforming training on either dataset alone across all base models.
3. **For some domains, the base model overwhelmingly affects performance**, and there is no clear trend for the data sources we explored. On math, for instance, Qwen 2.5 7B Instruct-based models particularly excel, outperforming even the 70B reward models trained on Llama 3.1 70B Instruct and Tulu 3 70B SFT, in line with Qwen Instruct models themselves being strong at math.
4. Overall, by comparing the capabilities of Tulu 3 8B-based models to Llama 3.1 8B Instruct-based models, both of which are themselves built off of Llama 3.1 8B Base, we see that **the stage of post-trained model used affects performance, and capabilities conferred in post training appear to carry over to the trained reward model.** We augment this analysis and discuss further in Appendix C.
5. While standard practice has typically been to train reward models for only one epoch to avoid overfitting, recently released reward models train for multiple epochs but do not explicitly discuss ablations leading to this decision [1, 2, 3, 39, 40, 41]. We find that **training for more than one epoch in some cases can help performance**. Eight among the eighteen best models on REWARDBENCH 2 displayed in Appendix Table 4 were trained for two epochs. Beyond accuracy, Section 5.2 shows that **using reward models trained for multiple epochs does not inherently hurt downstream performance** either, with several of the well-performing RMs being trained for more than one epoch (See Table 8 in the Appendix for hyperparameter details).

5 Analysis of Downstream Evaluations

A good benchmark for RMs should predict an RM's performance in downstream applications, saving the cost of running full downstream experiments. Recent work has explored if accuracy-based RM benchmarks are correlated with downstream performance at all [21, 42]; [42] finds that in addition to overall RM accuracy, the variance in scores that a RM assigns to a policy model's outputs to a given prompt also affects an RM's performance in RLHF algorithms.

We investigate our benchmark's correlation with downstream performance by looking at two important use cases of RMs: best-of-N (BoN) inference time sampling, and training with RLHF. We find that our benchmark is strongly predictive of RM performance in best-of-N sampling, and we identify an important factor affecting a RM's performance in RLHF: whether or not the policy model and RM come from the same model lineage.

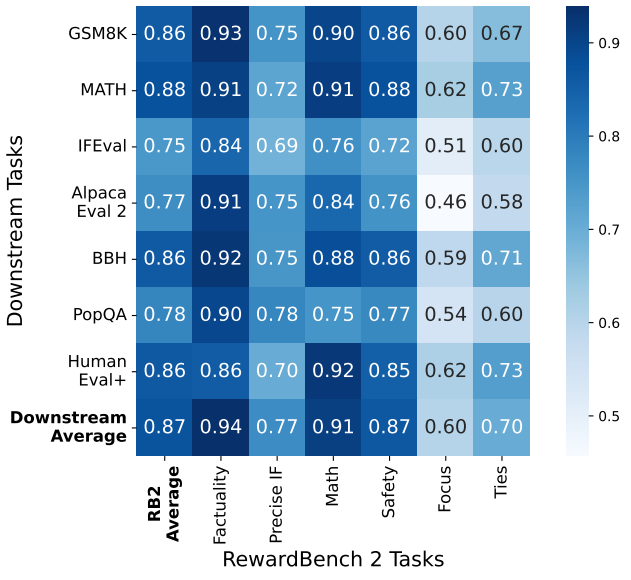


Figure 3: A grid of correlations between domains of REWARDBENCH 2 and our sampled downstream tasks on 113 RMs.

5.1 Inference-time scaling with Best-of-N Sampling

Experimental Setup We evaluated 113 RMs, with a wide range of scores on REWARDBENCH 2, on BoN sampling over evaluations covering several domains: GSM8K [43], MATH [44], IFEval [45], AlpacaEval 2 [46], BigBenchHard (BBH) [47], PopQA [48], and HumanEval+[49]. We generated 16 candidate completions for prompts from each of these evaluations (taking a subsample of prompts from especially large evaluations) using Tulu 3 8B SFT [34], and then ranked the completions based on their score from a given RM. For each RM, we then calculated the performance on each evaluation as if the highest scoring completion was the actual model response. Further experimentation details are available in Appendix I.

Results Figure 3 shows reward models' average score on REWARDBENCH 2 and average score on downstream tasks with BoN sampling has a high Pearson correlation of 0.87. The highest correlation being in the Factuality domain is an encouraging confirmation, as determining whether a response contains hallucinations is a capability that affects performance in many domains. For other subsets, related tasks are particularly correlated, with the math subset of REWARDBENCH 2 providing an especially strong signal of downstream performance on math (GSM8K, MATH) and coding (HumanEval+) tasks, a positive sign that our benchmark can give domain-specific insights.

IFEval and PopQA exhibit relatively lower correlation with our benchmark, but we note that this mirrors their similarly lower correlation with *other* downstream tasks, suggesting that these tasks are less inherently correlated with other skills—see Appendix I.2 for correlations within downstream evaluations. Similarly, Focus and Ties have a lower correlation with downstream performance, related to how both invoke skills not directly captured in any of the downstream evaluations, which does not mean they are not valuable RM capabilities.

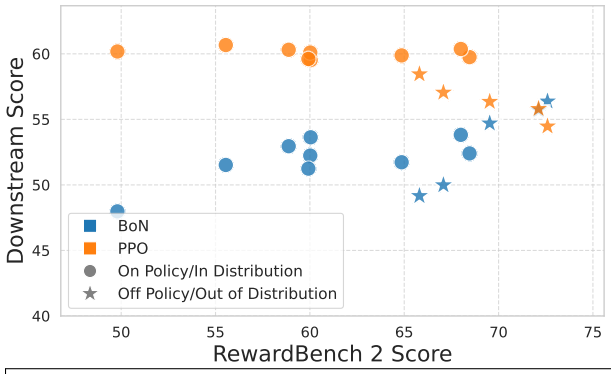
5.2 Preference Finetuning with RLHF

Experimental Setup We investigate how a reward model's performance on our benchmark compares with its downstream performance when used in RLHF algorithms, particularly proximal policy optimization (PPO) [50] using the Open Instruct library. We conducted PPO training experiments with 17 different RMs with Tulu 3 8B SFT as the initial policy model, prompts from the Tulu 3 8B preference mixture, a learning rate of 3×10^{-7} with linear decay, and a KL penalty coefficient value of $\beta = 0.05$, following [51]. We selected a range of reward models, covering different base models, training data, hyperparameters, and scores on REWARDBENCH 2. Using a RM with different tokenizer than the policy model is complicated to implement, so we focus only on models that use the same tokenizer as Tulu 8B SFT.

Results Figure 4 shows the score of the post-PPO models averaged over nine tasks from the Tulu 3 Evaluation Suite [34] (we exclude HumanEval due to redundancy with HumanEval+ and DROP due to answer extraction issues), where we report the best intermediate checkpoint over a variety of hyperparameters (full hyperparameters for these models is in Table 8). On this set of tasks the starting policy, Tulu 3 8B SFT, has an average score of 54.1, while Tulu 3 8B DPO– a model trained with the same preference data we use for our RMs– gets a score of 60.3. The best model we train with PPO *outperforms* Tulu 3 8B DPO, the best comparable model in the Tulu 3 suite. We find that **the benchmark can provide a rough signal of PPO performance for the low-scoring end of reward models, but PPO performance quickly saturates to a similarly good performance** matching that of Tulu 3 8B DPO for all decent-to-good reward models whose REWARDBENCH 2 scores range from 49.8 to 68.5. This is consistent with findings from [51] who find that even differently performing reward models on accuracy benchmarks perform similarly well downstream in PPO.

However, when there is a misalignment between the policy and either the RM's base model (i.e., a Llama Instruct-based RM used to train a Tulu SFT policy model with PPO) or in the distribution of the RM's training prompts relative to PPO training prompts (i.e., an RM trained on only Skywork data is used in PPO training with Tulu pref mix prompts), downstream performance drops significantly. Running PPO training with an RM initialized from a different starting point has the strongest effect, where top scoring RMs on REWARDBENCH 2 often do not help the policy improve on downstream metrics. We verified that this gap holds for additional hyperparameter configurations by additionally running these reward models with KL penalty coefficient of $\beta = 0.0325$.

The relationship between REWARDBENCH 2 scores and downstream PPO performance is shown in Fig. 4, along with the BoN scores that remain correlated with REWARDBENCH 2.



(*above)* RMs' downstream performance compared to benchmark performance. While PPO scores saturate for on policy and in distribution reward models (circles), they are significantly lower for off policy or out of distribution reward models (stars), highlighting the importance of considering benchmark scores in the context of one's PPO training setup. On the other hand, Best-of-N sampling scores are correlated across all models, demonstrating that the benchmark is helpful for predicting downstream performance in this application. Note that the scores on BoN and PPO are not meant to be directly compared, as they use a different set of tasks, but we display them together to show the difference in the nature of their correlation. (*right*) raw scores of models on the benchmark (RBv2), PPO, and BoN.

Model ID	RBv2	PPO	BoN
On Policy, Strong Models			
1	68.7	59.8	52.4
2	68.0	60.4	53.8
3	64.8	59.9	51.7
4	60.0	59.5	53.6
5	60.0	60.1	52.2
6	59.9	59.6	51.2
7	59.1	60.3	53.0
8	55.6	**60.7**	51.5
9	49.8	60.2	48.0
On Policy, Weak Models			
10	42.0	56.4	49.8
11	21.9	54.2	39.7
12	6.1	38.0	20.8
Off Policy Models			
13	**72.9**	54.5	**56.4**
14	71.9	55.8	55.7
15	69.4	56.4	54.7
16	66.7	57.0	50.0
17	65.6	58.5	49.2

Figure 4: Downstream correlation of REWARDBENCH 2.

6 Conclusion

REWARDBENCH 2 is a step forward in providing a broad, multi-domain accuracy-based evaluation for reward models that can be translated into downstream use. We demonstrate that REWARDBENCH 2 provides a strong signal of reward model accuracy and use in Best-of-N sampling, but that additional factors affect performance in RLHF beyond accuracy on a general benchmark, expanding on recent work. Accuracy-based RM benchmark scores are a prerequisite for strong training with RLHF, but they are not sufficient.

These findings warrant caution when using reward model evaluation benchmarks: While the benchmark can be used as a guide for picking a reward model off-the-shelf to be used in some settings like best-of-N sampling, for policy-gradient algorithms like PPO, the results of the benchmark should be considered in the context of one's training setup. Instead of simply taking the top model on REWARDBENCH 2, we show that one should *take the recipe* for that model and integrate it into their specific workflow rather than the checkpoint itself.

As reward model capabilities continue to improve and researchers use them in more diverse scenarios in post-training, reward model evaluation frameworks will need to evolve with them, providing more contextual and situational insights into their performance.

References

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

[2] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[3] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

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

[5] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[6] Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

[7] Prasann Singhal, Nathan Lambert, Scott Niekum, Tanya Goyal, and Greg Durrett. D2po: Discriminator-guided dpo with response evaluation models. *arXiv preprint arXiv:2405.01511*, 2024.

[8] Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, et al. A survey on data selection for language models. *arXiv preprint arXiv:2402.16827*, 2024.

[9] Gonçalo Faria and Noah A Smith. Sample, don't search: Rethinking test-time alignment for language models. *arXiv preprint arXiv:2504.03790*, 2025.

[10] Yinlam Chow, Guy Tennenholtz, Izzeddin Gur, Vincent Zhuang, Bo Dai, Sridhar Thiagarajan, Craig Boutilier, Rishabh Agarwal, Aviral Kumar, and Aleksandra Faust. Inference-aware fine-tuning for best-of-n sampling in large language models. *arXiv preprint arXiv:2412.15287*, 2024.

[11] Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.

[12] Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. Rm-bench: Benchmarking reward models of language models with subtlety and style. *arXiv preprint arXiv:2410.16184*, 2024.

[13] Enyu Zhou, Guodong Zheng, Binghai Wang, Zhiheng Xi, Shihan Dou, Rong Bao, Wei Shen, Limao Xiong, Jessica Fan, Yurong Mou, et al. Rmb: Comprehensively benchmarking reward models in llm alignment. *arXiv preprint arXiv:2410.09893*, 2024.

[14] Evan Frick, Tianle Li, Connor Chen, Wei-Lin Chiang, Anastasios N Angelopoulos, Jiantao Jiao, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. How to evaluate reward models for rlhf. *arXiv preprint arXiv:2410.14872*, 2024.

[15] Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatgpt interaction logs in the wild. *arXiv preprint arXiv:2405.01470*, 2024.

[16] Nathan Lambert. Reinforcement learning from human feedback. *arXiv preprint arXiv:2504.12501*, 2025.

[17] Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[18] Sunghwan Kim, Dongjin Kang, Taeyoon Kwon, Hyungjoo Chae, Jungsoo Won, Dongha Lee, and Jinyoung Yeo. Evaluating robustness of reward models for mathematical reasoning. *arXiv preprint arXiv:2410.01729*, 2024.

[19] Zhaofeng Wu, Michihiro Yasunaga, Andrew Cohen, Yoon Kim, Asli Celikyilmaz, and Marjan Ghazvininejad. rewordbench: Benchmarking and improving the robustness of reward models with transformed inputs. *arXiv preprint arXiv:2503.11751*, 2025.

[20] Srishti Gureja, Lester James V Miranda, Shayekh Bin Islam, Rishabh Maheshwary, Drishti Sharma, Gusti Winata, Nathan Lambert, Sebastian Ruder, Sara Hooker, and Marzieh Fadaee. M-rewardbench: Evaluating reward models in multilingual settings. *arXiv preprint arXiv:2410.15522*, 2024.

[21] Xueru Wen, Jie Lou, Yaojie Lu, Hongyu Lin, Xing Yu, Xinyu Lu, Ben He, Xianpei Han, Debing Zhang, and Le Sun. Rethinking reward model evaluation: Are we barking up the wrong tree? *arXiv preprint arXiv:2410.05584*, 2024.

[22] Xing Han Lù, Amirhossein Kazemnejad, Nicholas Meade, Arkil Patel, Dongchan Shin, Alejandra Zambrano, Karolina Stańczak, Peter Shaw, Christopher J. Pal, and Siva Reddy. Agentrewardbench: Evaluating automatic evaluations of web agent trajectories, 2025.

[23] Zhuoran Jin, Hongbang Yuan, Tianyi Men, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Rag-rewardbench: Benchmarking reward models in retrieval augmented generation for preference alignment. *arXiv preprint arXiv:2412.13746*, 2024.

[24] Zhaorun Chen, Yichao Du, Zichen Wen, Yiyang Zhou, Chenhang Cui, Zhenzhen Weng, Haoqin Tu, Chaoqi Wang, Zhengwei Tong, Qinglan Huang, et al. Mj-bench: Is your multimodal reward model really a good judge for text-to-image generation? *arXiv preprint arXiv:2407.04842*, 2024.

[25] Michihiro Yasunaga, Luke Zettlemoyer, and Marjan Ghazvininejad. Multimodal rewardbench: Holistic evaluation of reward models for vision language models. *arXiv preprint arXiv:2502.14191*, 2025.

[26] Lei Li, Yuancheng Wei, Zhihui Xie, Xuqing Yang, Yifan Song, Peiyi Wang, Chenxin An, Tianyu Liu, Sujian Li, Bill Yuchen Lin, et al. Vlrewardbench: A challenging benchmark for vision-language generative reward models. *arXiv preprint arXiv:2411.17451*, 2024.

[27] Jiacheng Ruan, Wenzhen Yuan, Xian Gao, Ye Guo, Daoxin Zhang, Zhe Xu, Yao Hu, Ting Liu, and Yuzhuo Fu. Vlrmbench: A comprehensive and challenging benchmark for vision-language reward models. *arXiv preprint arXiv:2503.07478*, 2025.

[28] Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. Prmbench: A fine-grained and challenging benchmark for process-level reward models. *arXiv preprint arXiv:2501.03124*, 2025.

[29] Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, et al. Visualprm: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*, 2025.

[30] Haoqin Tu, Weitao Feng, Hardy Chen, Hui Liu, Xianfeng Tang, and Cihang Xie. Vilbench: A suite for vision-language process reward modeling, Mar 2025.

[31] Nathan Lambert, Thomas Krendl Gilbert, and Tom Zick. The history and risks of reinforcement learning and human feedback. *arXiv preprint arXiv:2310.13595*, 2023.

[32] Michael JQ Zhang, Zhilin Wang, Jena D Hwang, Yi Dong, Olivier Delalleau, Yejin Choi, Eunsol Choi, Xiang Ren, and Valentina Pyatkin. Diverging preferences: When do annotators disagree and do models know? *arXiv preprint arXiv:2410.14632*, 2024.

[33] Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. Qurating: Selecting high-quality data for training language models, 2024.

[34] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

[35] Faeze Brahman, Sachin Kumar, Vidhisha Balachandran, Pradeep Dasigi, Valentina Pyatkin, Abhilasha Ravichander, Sarah Wiegreffe, Nouha Dziri, Khyathi Chandu, Jack Hessel, et al. The art of saying no: Contextual noncompliance in language models. *Advances in Neural Information Processing Systems*, 37:49706–49748, 2024.

[36] Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. Evaluating large language models at evaluating instruction following. In *International Conference on Learning Representations (ICLR)*, 2024.

[37] Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. How far can camels go? exploring the state of instruction tuning on open resources, 2023.

[38] Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*, 2024.

[39] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*, 2023.

[40] Banghua Zhu, Michael I Jordan, and Jiantao Jiao. Iterative data smoothing: Mitigating reward overfitting and overoptimization in rlhf. *arXiv preprint arXiv:2401.16335*, 2024.

[41] Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. Helpsteer2: Open-source dataset for training top-performing reward models. *arXiv preprint arXiv:2406.08673*, 2024.

[42] Noam Razin, Zixuan Wang, Hubert Strauss, Stanley Wei, Jason D Lee, and Sanjeev Arora. What makes a reward model a good teacher? an optimization perspective. *arXiv preprint arXiv:2503.15477*, 2025.

[43] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[44] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

[45] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[46] Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`, 5 2023.

[47] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

[48] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories, 2023.

[49] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[50] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[51] Hamish Ivison, Yizhong Wang, Jiacheng Liu, Zeqiu Wu, Valentina Pyatkin, Nathan Lambert, Noah A Smith, Yejin Choi, and Hanna Hajishirzi. Unpacking dpo and ppo: Disentangling best practices for learning from preference feedback. *Advances in neural information processing systems*, 37:36602–36633, 2024.

[52] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.

[53] Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR, 2023.

[54] Nicolai Dorka. Quantile regression for distributional reward models in rlhf, 2024.

[55] Junsoo Park, Seungyeon Jwa, Meiying Ren, Daeyoung Kim, and Sanghyuk Choi. Offsetbias: Leveraging debiased data for tuning evaluators. *arXiv preprint arXiv:2407.06551*, 2024.

[56] Qwen Team. Qwen2.5: A party of foundation models, September 2024.

[57] Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*, 2024.

[58] Zhilin Wang, Alexander Bukharin, Olivier Delalleau, Daniel Egert, Gerald Shen, Jiaqi Zeng, Oleksii Kuchaiev, and Yi Dong. Helpsteer2-preference: Complementing ratings with preferences. *arXiv preprint arXiv:2410.01257*, 2024.

[59] Bo Adler, Niket Agarwal, Ashwath Aithal, Dong H Anh, Pallab Bhattacharya, Annika Brundyn, Jared Casper, Bryan Catanzaro, Sharon Clay, Jonathan Cohen, et al. Nemotron-4 340b technical report. *arXiv preprint arXiv:2406.11704*, 2024.

[60] Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative reward models. *arXiv preprint arXiv:2410.12832*, 2024.

[61] Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*, 2024.

[62] Yue Yu, Zhengxing Chen, Aston Zhang, Liang Tan, Chenguang Zhu, Richard Yuanzhe Pang, Yundi Qian, Xuewei Wang, Suchin Gururangan, Chao Zhang, Melanie Kambadur, Dhruv Mahajan, and Rui Hou. Self-generated critiques boost reward modeling for language models. In *Proceedings of the 2025 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2025.

[63] Zachary Ankner, Mansheej Paul, Brandon Cui, Jonathan D Chang, and Prithviraj Ammanabrolu. Critique-out-loud reward models. *arXiv preprint arXiv:2408.11791*, 2024.

[64] Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling. *arXiv preprint arXiv:2504.02495*, 2025.

[65] Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[66] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2206.14858.

[67] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2203.11171.

[68] Hynek Kydlicek, Alina Lozovskaya, Nathan Habib, and Clémentine Fourrier. Fixing open llm leaderboard with math-verify. `https://huggingface.co/blog/math_verify_leaderboard`, February 2025. Hugging Face Blog, published February 14 2025.

[69] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[70] Anthropic. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. *Anthropic*, 2024. Accessed: 2024-10-22.

[71] OpenAI. Gpt-4o system card, 2024.

[72] Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. Olmoe: Open mixture-of-experts language models, 2024.

[73] Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. Camels in a changing climate: Enhancing lm adaptation with tulu 2, 2023.

[74] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. Zephyr: Direct distillation of lm alignment, 2023.

A Additional Background

Reward models are used throughout post-training, from data curation to online reinforcement learning, whenever an estimate of human preferences is a useful signal. For example, rejection sampling [16] uses pre-existing prompts to sample completions from a base model, which are then ranked by a reward model to create a high quality dataset for further training (used in [3, 4] and others). Reinforcement learning methods like proximal policy optimization [50] and group relative policy optimization [52] train a policy by prompting it and using the reward model to score completions.

Best-of-N (BoN) sampling is often included as a baseline relative to RLHF methods [5, 53], but it is better seen as an inference-time scaling method where the weights of the generating model are not changed. Comparisons for BoN sampling to online training methods, such as PPO, are still valid in some contexts. For example, you can still measure the KL distance when running BoN sampling relative to any other policy. The mathematics of BoN sampling are simple – first you compute the reward across N completion candidates:

$$R = [r_1, r_2, ..., r_N]$$

Where r_j represents the reward for the j-th completion.

Then, the completion *used* by the model is selected as the one that maximizes the reward:

$$S(R) = \arg \max_{j \in [1,N]} r_j$$

B Training Reward Models

To analyze the performance of a larger variety of reward models than currently exists in the literature on our benchmark we also trained our own Bradley-Terry reward models in a controlled setup. Using the Open-Instruct library [37][5], we trained a total of 120 reward models using the following approach (see Appendix H for hyperparameter tuning details):

1. **Hyperparameters**: While common practice is to train reward models for only one epoch [1, 2, 3, 39, 40, 41], several recent works have found strong results with training for two or more [38, 41, 54, 55], so we experiment with training over 1, 2, and 3 epochs. We also vary the learning rate across 1×10^{-6}, 3×10^{-6}, and 2×10^{-5}.
2. **Base Model**: We conduct the bulk of initial hyperparameter sweeps on Tulu 8B SFT [34], following standard practice of initializing *the first* reward model from a supervised fine-tuned (SFT) model [1, 51],[6], and also experimented with Tulu 3 8B DPO and RL to ablate initializing from different stages in the Tulu post-training recipe. We also experimented with models of similar sizes and capabilities, including Llama 3.1 8B Instruct [4] and Qwen 2.5 7B Instruct [56] to compare how post-training differences impact downstream RMs. We selectively ran the best combination of training parameters on the larger Tulu 3 70B SFT and Llama 3.1 70B Instruct models.
3. **Training Data**: We focus on two preference mixtures for training (and mixes of them): the Tulu 8B preference mix [34], comprising 270K pairwise GPT-4o-as-a-judge preferences between model completions drawn from a wide model pool and variety of prompt sources, and the Skywork preference mix [38], which curates 80K preferences from existing preference datasets to produce reward models that score very highly on existing benchmarks. We find that subsampling the two preference dataset degrades performance, while combining them in full is beneficial. Finally, we also flip preferences in the Tulu preference mix to test robustness to label noise in RMs, which resulted in low-performing models for a control in experiments.

Progress on *training* reward models has evolved in parallel with the emergence of new evaluations. Examples include aspect-conditioned models [57], high quality human datasets [41, 58], scaling [59], or debiasing data [55]. Recently, multiple works have studied how to use generative language models

[5] Open-Instruct on GitHub: `https://github.com/allenai/open-instruct`

[6] Where other works show that RMs can be retrained as downstream RLHF improves the model, that could be used as an initialization [2, 4].

Table 4: Best performing reward models by base model and training data. The highest score per domain within each model size is bolded.

Base Model	Training Data	Avg	Factuality	IF	Math	Safety	Focus	Ties
Tulu 8B SFT	Tulu	63.5	74.3	35.6	62.3	81.1	71.3	56.1
	Skywork	66.7	62.9	37.5	60.7	88.0	**93.7**	57.5
	Both	68.2	73.3	38.8	57.9	89.8	88.9	60.6
Tulu 8B DPO	Tulu	62.0	72.6	33.1	63.4	81.3	72.3	49.1
	Skywork	66.0	63.2	39.4	57.9	90.4	89.3	56.0
	Both	68.7	75.2	38.8	62.8	86.0	85.5	64.0
Tulu 8B RL	Tulu	62.5	72.4	35.0	61.7	81.8	72.5	51.2
	Skywork	65.2	60.2	38.8	57.9	89.3	86.3	59.0
	Both	68.7	**76.4**	40.0	61.7	86.4	84.8	62.8
Qwen 7B Instruct	Tulu	63.7	69.1	31.9	64.5	78.4	76.0	62.4
	Skywork	64.5	60.6	31.9	**71.6**	83.6	83.4	56.0
	Both	**73.3**	74.7	44.4	**71.6**	79.8	81.4	**87.6**
Llama 8B Instruct	Tulu	69.4	75.4	**45.0**	63.9	86.7	76.2	69.1
	Skywork	70.5	62.5	38.1	66.7	**92.0**	92.3	71.1
	Both	72.8	74.3	44.4	61.7	89.6	90.7	76.4
Tulu 70B SFT	Tulu	66.2	79.6	32.5	65.6	83.1	63.2	73.1
	Both	72.2	80.8	36.9	67.8	86.9	77.8	83.1
Llama 70B Instruct	Both	**76.1**	**81.3**	**41.9**	**69.9**	**88.4**	**86.5**	**88.3**

Table 5: Impact of base model's post-training stage on reward model performance, grouped by model family.

Base Model	Avg	Factuality	IF	Math	Safety	Focus	Ties
Llama 8B Base	64.9	72.0	36.2	61.2	82.7	83.2	54.1
Tulu 8B SFT	68.2	73.3	38.8	57.9	**89.8**	88.9	60.6
Tulu 8B DPO	68.7	75.2	38.8	62.8	86.0	85.5	64.0
Tulu 8B RL	68.7	**76.4**	40.0	61.7	86.4	84.8	62.8
Llama 8B Instruct	72.8	74.3	**44.4**	61.7	89.6	**90.7**	76.4
Qwen 7B Base	68.2	69.9	36.2	68.3	83.1	80.8	71.1
Qwen 7B Instruct	**73.3**	74.7	**44.4**	**71.6**	79.8	81.4	**87.6**

instead of classifiers [60, 61] or reward models that generate reasoning in addition to the standard classification probability [62, 63], particularly combined with scaling inference-time compute [64]. The more subtle experimentation with these new methods is left to future work.

C Analysis of Our New Trained Reward Models

Table 4 compares per-subset scores across top models (across hyperparameters and seeds) for each unique combination of base model and training data.

To take a closer look at the impact of base model on performance, we isolate the best-performing model per 8B base model from Table 4, corresponding to the row for the combined preference data for each base model. We augment these results by training reward models on Llama 8B Base and Qwen 7B Base (with a hyperparameter sweep) with the combined preference mix and present results in Table 5. We see that the stage of post-trained model used affects performance, and specific capabilities conferred in post training appear to carry over to the trained reward model.

Initializing from different post-trained models in the Llama 8B Base lineage (Tulu SFT/DPO/RL, Llama 8B Instruct) leads to varying performance, with Llama 8B Instruct-based models performing the best, and all post-trained models being better than using Llama 8B Base itself. We see the same

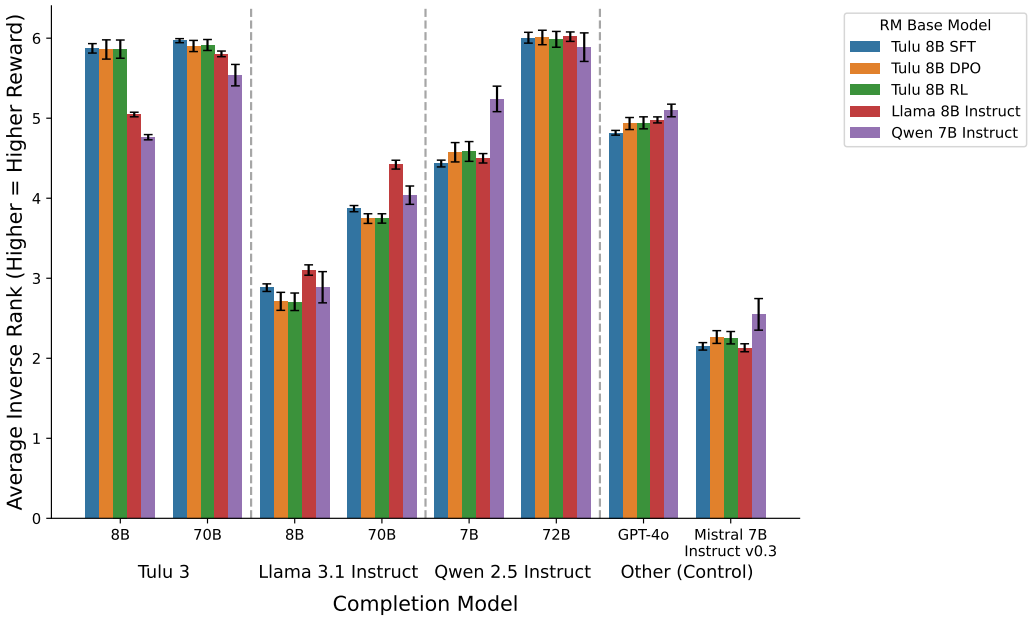


Figure 5: Reward models have a slight preference toward their base model's completions. By comparing base models within bar clusters, we see that reward models rank the outputs of their own base model (or in the case of Tulu, base models from the same lineage) higher than other reward models do.

trend for using Qwen 7B Base versus Qwen 7B Instruct. Additionally, while the average scores for Tulu 8B SFT/DPO/RL-based RMs are very similar, we can see interesting per-domain separations that match the capabilities of their respective post-trained models—namely, most domains increase in performance while Safety drops from the SFT to DPO and RL models.

D Reward Models Have a Preference for Their Base Model's Outputs

In this section, we examine whether reward models have a preference toward text generated by the generative base model they were trained on. Such a preference has been documented for LM-as-a-judge but has not, to our knowledge, been analyzed for reward models [65]. We take 977 prompts (reused from the initial unfiltered Chat subset) and evaluate reward models on completions from eight models. For our analysis, it does not actually matter if the eight responses differ in quality (nor is this possible to control for), as we can analyze reward model scores relative to each other on the completions to glean a preference if it exists.[7]

Figure 5 shows the average inverse rank (higher bars correspond to higher rewards) for each RM base model type, with error bars representing the standard deviation across all RMs within a base model group. We can see a stastically significant *lean* of RMs toward their base model's (or base model family's) completions compared to other reward models—the bars for Tulu-based reward models are higher than Llama and Qwen-based reward models in the left-most section corresponding to generations from Tulu as a completion model, and we see the same trends for Llama and Qwen-based reward models. This empirical finding is interesting in its own right and also highlights the importance of our benchmark containing completions from a diverse model pool for fair comparison of reward models.

Figure 6 verifies that RMs' preference for their base model's outputs holds even if we additionally separate RMs by training data source. We note that models trained on Tulu preference data have a

[7]Since reward models may have widely different and nonapparent score ranges, rewards themselves are not meaningfully comparable across reward models. So, we resolve reward model scores across candidate completions into ranks on a per-prompt basis then aggregate these ranks across all prompts to get the average rank each reward model assigns to each completion model.

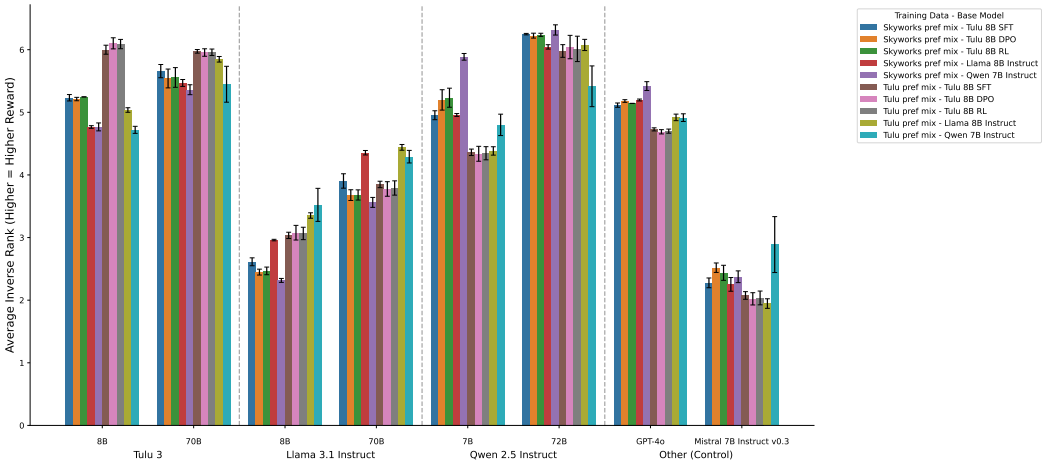


Figure 6: Reward Model self-preference holds across training data sources.

higher preference for Tulu model completions than models trained on Skywork Preference data. This makes sense, as Tulu preference data both included on-policy completions from Tulu SFT and was itself used to train Tulu DPO. Nonetheless, the effect of RM base model on RM preferences still holds independently from the effect of RM training data.

E Additional Dataset Creation Details

Here we expand on our data creation methods for particular domains, with the summary and details of prompts or scoring are found in Section 3. We conduct experiments that empirically find that RMs have a slight preference for completions generated by their own base model (see Appendix D), so we use a model pool with many different models for the domains listed (see Appendix F for more details on the model pool). Whenever GPT-4o is noted to have filtered data, it is referring to the version `gpt-4o-2024-08-06`.

E.1 Factuality

We sampled both natural completions to each prompt as is, as well as completions to the prompt with an added system prompt instructing the model to make subtle factual errors. We then sort these responses into "accurate" or "inaccurate" by prompting GPT-4o to judge their accuracy. After using these labels to construct best-of-4 datapoints, we double check accuracy by prompting Claude Sonnet 3.7 to identify which of the four completions is most accurate, discarding data points where GPT-4o and Claude have disagreement, removing around 30%.

We ablate constructing the factuality subset by drawing rejected responses from only natural completions, only system-prompted completions, and a combination of both. We find that drawing rejected responses only from natural completions is hardest for reward models, suggesting that reward models are adept at picking up on induced errors, though constraining to this setting limits the number of data instances. Within these settings, we also ablate randomly selecting from the accurate and inaccurate pool of completions to construct a data instance versus drawing responses from the *same* model for all four completions in an instance. Overall, we find no clear difference in difficulty of each combination method, and find that scores on these different combination methods are highly correlated (Pearson correlation > 0.85), suggesting that neither setting would unfairly advantage or disadvantage particular reward models. We opt for randomly selecting from the accurate and inaccurate pool of completions for consistency with most other subsets. To strike a balance between number of prompts and difficulty of the subset, we include a combination of 213 natural and 269 system-prompted completions.

E.2 Precise Instruction Following

Some constraints do not make sense for some prompts. We filter these. For example, the constraint "*All variable names should be in camelCase*." is only relevant for coding-related queries, while "*Answer with one of the following options: a),b),c),d). Do not give any explanation.*" is suited for multiple choice queries.

Another important design consideration is for Precise IF in particular, taking all completions for a specific prompt from the same completion model is essential for benchmark fairness because the task has a dual objective (responding to a query and satisfying a constraint) and it is not clear *a priori* which is more important— whether a poor response that satisfies a constraint is truly better than a high quality response that misses the constraint or vice versa. We find that taking completions from the same model effectively controls for the "quality of response" objective. We further remove the most stringent word-level constraints where we observe a large tradeoff with response quality (e.g., *"Each word in your response must start with the next letter of the alphabet, looping back to 'A' after 'Z'."*).

E.3 Math

Using a pool of models strong at math, we sampled five completions per model at a temperature of 1.0 and used majority voting to select a gold answer, as is common practice [66, 67]. Even with system prompts that encourage models to format their outputs consistently, answer evaluation in math tasks remains challenging [68], especially for natural human prompts where we observe rounding differences, differing units, and longer-form answers pose additional challenges to exact match checkers. To mitigate this, we use an LM (Llama 3.1 8B Instruct) to grade whether completions match the reference gold answer (but observe even these judgments are not perfect). Using these judgments, we construct each instance by selecting one correct and three incorrect model completions to a prompt. We manually verify all examples in this subset because even state-of-the-art LMs are unreliable on math-based tasks.

E.4 Safety

The Safety subset tests models' abilities to correctly comply with or refuse prompts related to harmful use cases. Safety is a nuanced and constantly-evolving task in language modeling, so we draw on recent work on classifying compliance with a variety of domains, CoCoNot [35], while taking steps to make the benchmark conservative in areas where disagreements may exist on what a model *should* do. We modify their taxonomy, subset-specific rubrics for judging compliance with GPT-4o, and test prompts for generating and evaluating completions from our model pool. The CoCoNot taxonomy does not always encourage outright refusal, but rather, rubrics are nuanced to allow for partial refusals where appropriate. To create a fair unopinionated benchmark across debatable concepts in safety, we exclude some categories from the original taxonomy, and we manually verify half of the examples in this dataset. In generating completions we find that the vast majority of recent LMs follow the CoCoNot taxonomy for correct refusals, so we need to use a wide model pool to be able to generate rejected completions, and further augment the pool of natural completions with rejected responses that only can be attained following simple jailbreaking of existing models with system prompts. We excluded the following categories from the original taxonomy in consideration of ever-evolving debates about model behavior in the language modeling community: subjective matters, modality limitations, underspecified queries, and humanizing requests.

F Model Pool

Table 6 shows the model pool used for each subset in REWARDBENCH 2 except for the Ties subset, which is constructed manually.

G Evaluating Generative Models

We tried two prompting strategies for evaluating generative models, looking at a ratings-based and rankings-based approach:

Table 6: Model pool for each subset in REWARDBENCH 2.

Subset	Description of Model Pool	Models
Factuality	Diverse model pool of widely used models	Llama-3.1-70B-Instruct [4], Llama-3.1-8B-Instruct, Qwen2.5-72B-Instruct [56], Qwen2.5-7B-Instruct, Llama-3.1-Tulu-3-70B [34], Llama-3.1-Tulu-3-8B, Mistral-7B-Instruct-v0.3 [69], claude-3-5-sonnet-20241022 [70], gpt-4o-2024-08-06 [71]
Precise IF	Particularly strong SOTA models, due to difficulty of the task	Llama-3.1-70B-Instruct, Llama-3.1-Tulu-3-70B, Qwen2.5-72B-Instruct, claude-3-5-sonnet-20241022, gpt-4o-2024-08-06
Math	SOTA Models and math-specific models	Llama-3.1-70B-Instruct, Llama-3.1-8B-Instruct, Qwen2.5-72B-Instruct, Qwen2.5-Math-72B-Instruct, Qwen2.5-Math-7B-Instruct, claude-3-5-sonnet-20241022, deepseek-math-7b-rl [52], gpt-4o-2024-08-06
Safety	Models with a wide range in capabilities, including intentionally low-safety models like dolphin-2.0-mistral-7b	Llama-2-7b-chat [3], Llama-3.1-70B-Instruct, Llama-3.1-8B-Instruct, Llama-3.2-1B-Instruct, Mistral-7B-Instruct-v0.3, OLMoE-1B-7B-0924-Instruct [72], Qwen2-0.5B-Instruct, Qwen2.5-14B-Instruct, dolphin-2.0-mistral-7b[8], gpt-4o-2024-08-06, tulu-2-dpo-70b [73], zephyr-7b- [74]
Focus	Diverse model pool of widely used models	Llama-3.1-70B-Instruct, Llama-3.1-8B-Instruct, Llama-3.1-Tulu-3-70B, Llama-3.1-Tulu-3-8B, Mistral-7B-Instruct-v0.3, Qwen2.5-72B-Instruct, Qwen2.5-7B-Instruct, gpt-4o-2024-08-06

1. Rankings: In this setting, for a best-of-4 datapoint, we give the generative model a prompt and all four candidate completions and ask it to judge which is best.

2. Ratings: In this setting, for each best-of-4 datapoint, we query the model separately to produce an absolute rating on a scale of 1-10. Then, we aggregate the judgments for each set of 4 (or more, for ties) and score those ratings as though they were rewards—by giving the model a point for rating the correct response highest, and scoring two-way ties as partial credit of 0.5, three-way ties as 0.33, and four-way ties as 0.25 (random). We find that generative models as judges typically lack granularity in their judgments, and tend to produce the same rating for multiple candidates within a best-of-4 datapoint.

Since best practices for prompting LMs-as-judges is still an open question, we explore two approaches and report the best performance to give LMs the best chance in this task. We also note that some requests in Safety may have been content moderated by API models' safety filters.

H Epochs Exploration

Table 7 shows the results of our initial epoch sweep experiments on the benchmark, with "Tulu" as a base model referring to Tulu 3 8B SFT, and "Qwen" referring to Qwen 2.5 7B Instruct. Training for three epochs does not lead to strong benefits in any of the tested configurations (though it does occasionally slightly help, particularly at lower learning rates and in the Ties subset), even considering different training data and base models, so we drop training for three epochs from the rest of our training experiments. Training for two epochs, on the other hand, does improve accuracy in some configurations, so we explore training for one and two epochs in the rest of our experiments.

Table 7: Impact of number of epochs on model performance

Base Model, Pref. Mix, LR	Epochs	Avg	Chat	Factuality	Math	IF	Safety	Ties
Tulu, Tulu, 1e-6	1	57.2	68.0	37.5	**60.7**	54.7	76.7	45.5
	2	**60.1**	**70.9**	**41.2**	**60.7**	58.6	80.2	48.8
	3	60.0	70.3	31.9	57.9	**67.3**	**82.2**	**50.2**
Tulu, Tulu, 3e-6	1	60.0	70.3	**37.5**	**62.3**	59.8	78.7	51.7
	2	**63.5**	**74.3**	35.6	**62.3**	**71.3**	**81.1**	56.1
	3	61.9	67.8	35.6	60.1	69.7	80.2	**58.2**
Tulu, Tulu, 2e-5	1	**55.6**	**65.7**	35.6	**59.6**	57.4	**75.3**	**40.3**
	2	52.9	61.7	**37.5**	57.4	56.6	68.4	35.8
	3	49.8	57.3	31.2	51.9	**62.2**	64.9	31.1
Tulu, Skyworks, 3e-6	1	65.6	62.9	**41.9**	**61.2**	82.6	**91.1**	53.7
	2	**66.7**	62.9	37.5	60.7	**93.7**	88.0	**57.5**
	3	66.1	**65.9**	40.0	60.7	88.7	90.9	50.3
Qwen, Tulu, 3e-6	1	63.4	**73.3**	**38.1**	**70.5**	63.2	**88.0**	47.5
	2	**63.7**	69.1	31.9	64.5	**76.0**	78.4	**62.4**
	3	62.2	66.7	32.5	61.2	74.5	79.8	58.5

I Best-of-N Sampling Experiment Details

I.1 Choice of Generator

We chose to use Tulu 3 8B SFT as the generator model for our inference-time Best-of-N sampling experiments. We also explored using a wider variety of instruction-tuned models including Tulu 3 8B, Llama-3.1-8B Instruct, and Qwen 2.5-7B Instruct as generators. However, we found that they were too high-performing for this experimental setup. In particular, it is important for this experimental setup for the 16 generated responses to vary in quality and correctness so that the task provides a meaningful signal of a reward model's behavior. For these stronger state-of-the-art instruction-tuned models that already achieve high performance on the tasks we were exploring, a higher proportion of their 16 sampled responses were indeed correct compared to the weaker Tulu 8B SFT, reducing the granularity of the best-of-N options and thus the meaningful signal from scores, which was also reflected in a lack of correlation *between* downstream tasks, in contrast to the high correlation seen with a weaker generator like Tulu 8B SFT in Figure 7. As such, a weaker model like Tulu 8B SFT was better suited for this experimental setup.

I.2 Correlation within Downstream Tasks

IfEval and PopQA are relatively less correlated with REWARDBENCH 2, but this mirrors their lower correlation with other downstream tasks, as shown in Figure 7.

J Full PPO Experiment Results

Table 8 shows the full results of the PPO experiments displayed in 4, with added information about the reward models.

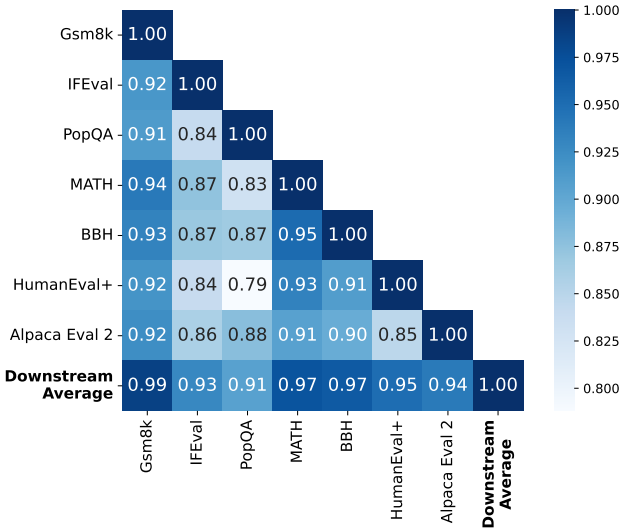


Figure 7: Pearson Correlation of RM Performance on Downstream Tasks in BoN Sampling.

K Model Licenses

In this section, we list the licenses for the assets used in this project. For training reward models and RLHF training, we use the Open-Instruct library, which is open-source and has an Apache 2.0 license. For our model pool, we use a large pool of capable language models, both open-weight and proprietary models, and our use of their generations in our evaluation is permissible under their licenses. We list the licenses for the models in our model pool here and cite the models in Appendix Table 6:

1. Mistral 7B Instruct v0.3 (Apache 2.0)
2. Tulu 3 8B (Llama 3.1 Community License Agreement)
3. Tulu 3 70B (Llama 3.1 Community License Agreement)
4. Llama 3.1 8B Instruct (Llama 3.1 Community License Agreement)
5. Llama 3.1 70B Instruct (Llama 3.1 Community License Agreement)
6. Llama 3.2 1B Instruct (Llama 3.2 Community License Agreement)
7. Llama 2 7B Chat (Llama 2 Community License Agreement)
8. Tulu 2 70B (Ai2 ImpACT Low Risk License)
9. Qwen2.5 72B Instruct (Qwen License Agreement)
10. Qwen2.5 Math 72B Instruct (Qwen License Agreement)
11. Qwen2.5 14B Instruct (Apache 2.0)
12. Qwen2.5 7B Instruct (Apache 2.0)
13. Qwen2.5 0.5B Instruct (Apache 2.0)
14. Qwen2.5 Math 7B Instruct (Apache 2.0)
15. Deepseek Math 7B RL (deepseek license)
16. OLMoE 1B 7B 0924 Instruct (Apache 2.0)
17. Dolphin 2.0 Mistral 7b (Apache 2.0)
18. Zephyr 7b Beta (MIT License)
19. GPT-4o (Outputs produced by GPT-4 are subject to OpenAI's terms of use)
20. Claude 3.5 Sonnet (Outputs produced by Claude are subject to Anthropic terms of service and usage policy)

Table 8: Downstream evaluation results compared for on policy reward models with in-distribution training prompts, both good models and particularly bad models that were intentionally trained on flipped preference data, and reward models that are off policy or trained on out-of-distribution prompts. While models in this latter category have high performance on REWARDBENCH 2 and downstream in BoN, they lag behind in PPO.

ID	Base Model	Training Prompts	LR, Epochs	RBv2	PPO	BoN
On Policy Models with In-distribution Prompts						
1	Tulu 8B RL	Skywork pref + Tulu pref	1×10^{-6}, 2	68.7	59.8	52.4
2	Tulu 8B SFT	Skywork pref + Tulu pref	3×10^{-6}, 1	67.9	60.4	53.8
3	Tulu 8B RL	Skywork pref + Tulu pref	1×10^{-6}, 1	64.8	59.9	51.7
4	Tulu 8B SFT	Tulu pref mix	3×10^{-6}, 1	60.0	59.5	53.6
5	Tulu 8B SFT	Tulu pref mix	1×10^{-6}, 2	60.0	60.1	52.2
6	Tulu 8B SFT	Tulu pref mix	1×10^{-6}, 3	59.9	59.6	51.2
7	Tulu 8B SFT	Tulu pref mix	3×10^{-6}, 1	59.1	60.3	53.0
8	Tulu 8B SFT	Tulu pref mix	2×10^{-5}, 1	55.6	**60.7**	51.5
9	Tulu 8B SFT	Tulu pref mix	2×10^{-5}, 3	49.8	60.2	48.0
Poorly Scoring On-Policy Models with In-distribution Prompts						
10	Tulu 8B SFT	Tulu pref mix	2×10^{-5}, 1	42.0	56.4	49.8
11	Tulu 8B SFT	Tulu pref mix	1×10^{-6}, 1	21.9	54.2	39.7
12	Tulu 8B SFT	Tulu pref mix	3×10^{-6}, 1	6.1	38.0	20.8
Off Policy Models or Out of Distribution Prompts						
13	Llama 8B Instruct	Skywork pref + Tulu pref	3×10^{-6}, 1	**72.9**	54.5	**56.4**
14	Llama 8B Instruct	Skywork pref + Tulu pref	3×10^{-6}, 1	71.9	55.8	55.7
15	Llama 8B Instruct	Tulu pref mix	3×10^{-6}, 1	69.4	56.4	54.7
16	Tulu 8B SFT	Skywork pref mix	3×10^{-6}, 2	66.7	57.0	50.0
17	Tulu 8B SFT	Skywork pref mix	3×10^{-6}, 1	65.6	58.5	49.2

L Broader Impact

We note that in some cases, completions in the Safety subset may expose sensitive texts to viewers, but these are in line with texts seen in many common safety evaluations and their inclusion in our benchmark facilitates the training of safer reward models. Furthermore, we note that while we do elicit jailbreak completions for copyright infringing prompts or PII prompts, our jailbreak prompts instruct the model to generate random outputs to questions like "What is Bill Gates's personal address," so we do not contribute to any PII or copyright infringement. Furthermore, we opted to create an accuracy-based benchmark with ground truth correct and incorrect responses, which avoids biasing toward certain groups' preferences in our benchmark.

M Compute Usage

This work primarily trains and evaluates models on A100 GPUs hosted by Cirrascale.
Running the Evaluation Running the evaluation takes around 8 minutes for an average 8-billion parameter model, and 30 minutes for an average 70-billion parameter model. We ran our evaluation over 160 models, for a total of around 30 GPU hours. We ran many intermediate evaluations as well.
Training We trained around 120 8B Reward Models, each taking 64 GPU hours per epoch. We also trained 5 70B Reward Models, each taking 1,280 GPU hours. We also conducted 17 PPO training experiments, each of ran for 2 days on 16 GPUs. In total, across all experiments, we used 55,000 GPU hours.