

Small Reward Models via Backward Inference

Yike Wang[♡] Faeze Brahman[♣] Shangbin Feng[♡] Teng Xiao[♣]
Hannaneh Hajishirzi^{♡♣} Yulia Tsvetkov[♡]

[♡]University of Washington [♣]Allen Institute for Artificial Intelligence
yikewang@cs.washington.edu

Abstract

Reward models (RMs) play a central role throughout the language model (LM) pipeline, particularly in non-verifiable domains. However, the dominant LLM-as-a-Judge paradigm relies on the strong reasoning capabilities of large models, while alternative approaches require reference responses or explicit rubrics, limiting flexibility and broader accessibility. In this work, we propose FLIP (FLipped Inference for Prompt reconstruction), a reference-free and rubric-free reward modeling approach that reformulates reward modeling through backward inference: inferring the instruction that would most plausibly produce a given response. The similarity between the inferred and the original instructions is then used as the reward signal. Evaluations across four domains using 13 small language models show that FLIP outperforms LLM-as-a-Judge baselines by an average of 79.6%. Moreover, FLIP substantially improves downstream performance in extrinsic evaluations under test-time scaling via parallel sampling and GRPO training. We further find that FLIP is particularly effective for longer outputs and robust to common forms of reward hacking. By explicitly exploiting the validation-generation gap, FLIP enables reliable reward modeling in downscaled regimes where judgment methods fail.¹

1 Introduction

Reward models (RMs) are widely used throughout the language model (LM) pipeline: they play a central role in reinforcement learning, preference optimization, reranking, and automatic evaluation, particularly in non-verifiable domains (Ouyang et al., 2022; Rafailov et al., 2023; Brown et al., 2024; Snell et al., 2024; Zhang et al., 2024b; Yu et al., 2025b). A dominant approach instantiates RMs via direct judgment, a.k.a. *LLM-as-a-Judge* (Zheng

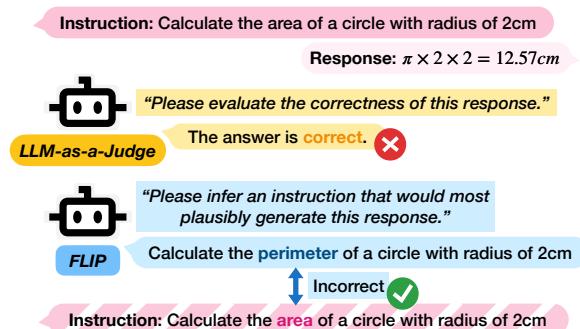


Figure 1: While LM judges can be misled, FLIP effectively identifies off-topic, instruction-misaligned, and factually incorrect responses via backward inference.

et al., 2023). In this paradigm, a prompted LM evaluates responses and produces scalar scores, preference labels, or textual critiques that are used as reward signals. In practice, the reliability of such judge-based method depends heavily on the superior reasoning capabilities of larger base models (Zheng et al., 2023; Gu et al., 2024). When applied to small language models (SLMs),² performance drops by 41% according to the RewardBench2 leaderboard (Malik et al., 2025). The cost is also amplified by the fact that RMs are invoked repeatedly during optimization and evaluation, often dominating the overall compute footprint (Ouyang et al., 2022; Bai et al., 2022). These considerations motivate our goal: *to develop reward modeling methods that remain reliable and effective when downscaled to small general-purpose language models.*

In this work, we argue that effective and efficient reward modeling in downscaled regimes requires a reformulation of the judgment paradigm itself. Based on the intuition that given a high-quality response with sufficient length and contextual richness, the original query should be able to be inferred from it, we introduce FLIP (FLipped Inference for Prompt reconstruction), a reward modeling

¹Code available at <https://github.com/yikee/FLIP>.

²We define small language models as language models with 8B parameters or fewer.

approach based on backward inference—*inferring the instruction that would most plausibly produce a given response*. Given a response, a language model is used to infer the instruction under which the response would most likely arise, and *the similarity between the inferred instruction and the original instruction is used as the reward signal* (Figure 1). We explain our method through Bayesian theory (Bernardo and Smith, 1994).

Experimental results on a standard reward modeling benchmark across four domains and 13 SLMs spanning multiple families and sizes show that SLMs with backward inference are strong reward models. Our approach consistently surpasses judgment baselines by an average of 79.6%, and substantially outperforms them in extrinsic evaluations under test-time scaling via parallel sampling across four benchmarks, as well as Group Relative Policy Optimization (GRPO) training across five benchmarks. FLIP is well-suited for downscaled regimes as it does not rely on reference responses or rubrics, which are costly and difficult to obtain. We further find that FLIP is particularly effective on longer outputs and is robust to common forms of reward hacking. We hypothesize that the effectiveness of FLIP stems from the validation–generation gap (West et al., 2024; Li et al., 2023b), which is more pronounced in smaller language models: while small models often struggle with judgment, they remain relatively strong in generative inference.

In summary, (1) we empirically examine reward modeling in downscaled settings and demonstrate that judgment becomes unreliable for small models; (2) we propose FLIP, a generative reformulation of reward modeling that is reference-free and rubric-free and explicitly designed for SLMs; and (3) we show through extensive intrinsic and extrinsic evaluations that FLIP substantially outperforms judgment baselines. We hope that our findings encourage the community to rethink how judgment is operationalized in reward modeling and to further pursue cost-efficient training and inference of language models in downscaled regimes (Belcak et al., 2025; Goel et al., 2025).

2 Reward Modeling

In this section, we present the formulation of reward modeling in 2.1, our proposed method FLIP in 2.2, as well as baseline settings in 2.3. We later showcase the efficacy of our method on reward modeling benchmarks (Section 3.1), in test-time

scaling (Section 3.2), and in Reinforcement Learning (Section 3.3).

2.1 Problem Setting

Let x denote an instruction and y a response generated by a language model parametrized by θ ,

$$y \sim p_\theta(y | x).$$

Given a language model parameterized by ϕ as the reward model, our goal is to estimate a scalar reward $r \in \mathbb{R}$ that reflects the quality of the response. We define this reward via a scoring function

$$r(x, y) = f_1(\text{LM}_\phi, x, y),$$

where $f_1(\cdot)$ maps LM_ϕ and the pair (x, y) to a real-valued score capturing attributes such as correctness, helpfulness, and overall response quality.

In some cases, we are given a set of candidate responses $y_1, y_2, \dots, y_n$ and our goal is to estimate a ranking (i.e., a preference ordering) over these responses using the reward model LM_ϕ . Specifically, we seek a permutation π such that

$$y_{\pi(1)} \succ y_{\pi(2)} \succ \dots \succ y_{\pi(n)}.$$

Formally, we define the ranking function as

$$\pi(x, y_1, y_2, \dots, y_n) = f_2(\text{LM}_\phi, x, y_1, y_2, \dots, y_n),$$

where $f_2(\cdot)$ returns a permutation over indices $\{1, \dots, n\}$ induced by LM_ϕ ’s preferences.

LLM-as-a-Judge directly employs LM_ϕ as the evaluator (Zheng et al., 2023). Formally, it defines a conditional distribution:

$$r \sim p_\phi(r | x, y),$$

where r may be represented as a continuous value, a discrete ordinal rating, or a textual justification that is subsequently mapped to a scalar (Figure 2 (a)). We introduce its variants in Section 2.3.

2.2 Our method: FLIP

In this section, we introduce a simple, yet effective method: FLIP (Figure 3), based on the intuition that given a high-quality response with sufficient length and contextual richness, the original query should be able to be inferred from it. FLIP transforms LLM-as-a-Judge into a *generative* setting by sampling an inferred instruction x' given the response y , and defining the reward r as the similarity between the inferred instruction x' and the original instruction x . FLIP can be used to sample either a scalar reward for an individual response or a preference ranking over a set of responses.

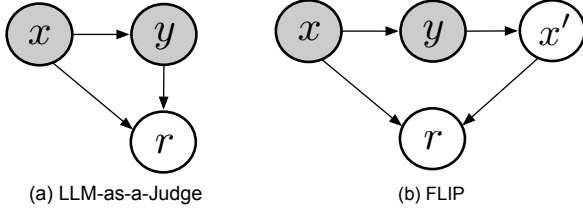


Figure 2: Graphical models of LLM-as-a-Judge and FLIP. Shaded nodes denote observed variables and unshaded nodes denote prediction targets. FLIP samples an inferred instruction x' conditioned on the response y , and defines the reward r as the similarity between the inferred and the original instructions.

Formulation Formally, an inferred instruction x' is sampled from the reward model,

$$x' \sim p_\phi(x' | y).$$

The scalar reward $r \in \mathbb{R}$ is defined as

$$r = s(x, x'),$$

where $s : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a similarity function measuring the agreement between the original instruction x and the inferred instruction x' . By the chain rule, the joint conditional distribution is

$$p_\phi(r, x' | x, y) = p_\phi(r | x, x', y) p_\phi(x' | x, y).$$

As shown in Figure 2 (b), the reward depends only on instructions (i.e., $r \perp y | (x, x')$) and that the inferred instruction depends only on the response (i.e., $x' \perp x | y$), this reduces to

$$p_\phi(r, x' | x, y) = p_\phi(r | x, x') p_\phi(x' | y).$$

Marginalizing over the latent variable x' yields

$$p_\phi(r | x, y) = \sum_{x'} p_\phi(r | x, x') p_\phi(x' | y).$$

Given a response with enough contextual information, the posterior distribution $p_\phi(x' | y)$ is assumed to be sharply peaked around its mode, and the similarity function $s(x, x')$ is assumed to be insensitive to small perturbations of x' in a neighborhood of this mode. For simplicity and computational efficiency, we approximate the latent variable x' by its maximum a posteriori (MAP) estimate,

$$x'_{\text{MAP}} = \arg \max_{x'} p_\phi(x' | y).$$

Under these assumptions, the marginal reward distribution can be approximated by

$$p_\phi(r | x, y) \approx p_\phi(r | x, x'_{\text{MAP}}) = s(x, x'_{\text{MAP}}).$$

Implementation Given an instruction x and a candidate response y , we first sample an inferred instruction x' based solely on the response y using the prompt “Infer a single instruction that would most plausibly generate the given response.”:

$$x' = \text{LM}_\phi(y).$$

We then define the reward r as the F1 score (van Rijsbergen, 1979), the harmonic mean of word-level precision and recall, between the inferred instruction x' and the ground-truth instruction x :

$$r(x, y) = s(x, x') = \text{F1}(x, x').$$

Ablation studies and justification on prompts and similarity metrics are detailed in Appendix C and Appendix D, respectively.

In scenarios where the instruction x consists of both a system prompt and a user prompt, we treat the user prompt as an additional context and infer only the system prompt. Specifically, given the response y and the user prompt x_{user} , the model infers a system prompt x'_{sys}

$$x'_{\text{sys}} = \text{LM}_\phi(y, x_{\text{user}}),$$

and the reward is defined as the F1 score between the inferred system prompt x'_{sys} and the ground-truth system prompt x_{sys}

$$r(x, y) = \text{F1}(x_{\text{sys}}, x'_{\text{sys}}).$$

Similarly, for multi-turn conversations, the conversation history will be treated as additional context.

2.3 Baselines

We consider three variants of LLM-as-a-Judge that operate under the same constraints as our method: reference-free and rubric-free reward modeling using general-purpose language models. We exclude trained reward models and rubric-based methods from our primary comparison, as these require additional resources such as human preference annotations or manually designed rubrics that fall outside our target regime.

Pointwise Rating Given an instruction x and a response y , the model is prompted to assign an absolute score on a fixed scale (e.g., 1–10). The scalar reward is obtained directly from the model:

$$r(x, y) = \text{LM}_\phi(x, y),$$

where $r \in \mathbb{R}$ denotes the predicted quality score. In the case of multiple responses, the preference ordering π is derived by sorting the scalar rewards

$$r(x, y_{\pi(1)}) \geq r(x, y_{\pi(2)}) \geq \dots \geq r(x, y_{\pi(n)}).$$

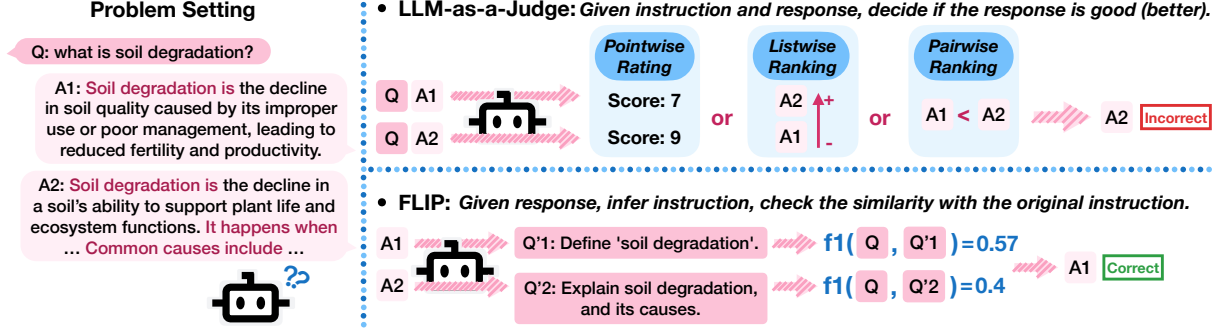


Figure 3: Overview of FLIP. Given a response, we use a LM to infer the instruction that would most plausibly generate the response, and use the F1 score between the inferred and original instructions as the reward.

Listwise Ranking This variant is only applicable when the objective is to infer a preference ordering over a set of candidate responses. Formally, it samples the preference ranking π directly

$$\pi(x, \{y_1, \dots, y_n\}) = \text{LM}_\phi(x, \{y_1, \dots, y_n\}).$$

Pairwise Ranking This variant is only applicable when given a pair or a list of candidate responses. Given x and a pair of responses (y_i, y_j) , the model is prompted to select the preferred response

$$y_i \succ y_j \text{ if } \text{LM}_\phi(x, y_i, y_j) = y_i.$$

A listwise ranking π over a set of candidates can be derived by aggregating these pairwise preferences across all response pairs, for example using a voting or tournament-based aggregation scheme.

3 Experiment Settings

3.1 Intrinsic Evaluation

Benchmark We evaluate on RewardBench2 (Malik et al., 2025), a challenging and comprehensive benchmark that assesses reward modeling on large-scale unseen prompts. For each instruction, four responses are sampled from more than 40 models of various sizes and families, ranging from small models to large commercial models. The accuracy is computed by selecting the preferred response from four for each prompt. Our evaluation is conducted on 1,313 instances across four subsets.³

Reward Models and Implementation We experiment with 13 SLMs as generative reward models, spanning five model families: OLMo2 (OLMo et al., 2024), Llama3 (Dubey et al., 2024), Qwen3

(Yang et al., 2025), Gemma3 (Team et al., 2025), and Mistral-v0.3 (Albert et al., 2023). For each model, we use its instruction-tuned variant. We set a token limit of 512 for all methods, and all reported results are averaged over five runs.

3.2 Test-Time Scaling with Parallel Sampling

We also test the effectiveness of our method in parallel test-time scaling (Best-of-N sampling), one of the most widely used inference-time scaling methods (Brown et al., 2024; Snell et al., 2024; Huang et al., 2025). It proceeds by generating N candidate responses for a given prompt, then returning the response with the highest reward score.

Benchmarks We experiment with four benchmarks: AlpacaEval (Li et al., 2023c), which consists of instructions representative of user interactions on Alpaca web (Taori et al., 2023); Human Interest (Feng et al., 2024), with instructions synthesized across 16 domains including electric vehicles and PhD applications; MATH (Hendrycks et al., 2021), a collection of challenging competition-level math problems; and IFEval (Zhou et al., 2023), a benchmark for instruction-following proficiency. From each benchmark, we sample 300 instructions from the test set.

Models and Implementation We sample 16 candidate completions from Tulu-3-8B (Lambert et al., 2024) and Tulu-3-8B-sft (Lambert et al., 2024) (IFEval) with a maximum token limit of 2,048 and a temperature of 1.0, and evaluate five reward models of varying sizes from OLMo2 and Llama3 using FLIP and three baselines applicable in this setting. For each reward model, we rank the completions according to the scores produced by a given method and compute benchmark performance by treating the highest-scoring completion as the model’s final response. All results are averaged over five runs.

³Because FLIP only targets open-ended instruction-following problems, we exclude two sub-tasks whose formats are incompatible: the *Ties* subset as its responses consist solely of one-word answers, and the *Safety* subset as correct behavior is to refuse unsafe instructions rather than follow them.

3.3 Reinforcement Learning

We investigate the performance of FLIP in comparison to LLM-as-a-Judge when applied to GRPO training (Shao et al., 2024)⁴. GRPO aligns a language model by sampling K candidate responses for a given prompt, assigning each response a reward score, and computing a group-normalized advantage to guide optimization.

Training Setup We use the DPO-trained version of OLMo3-7B-Think (Olmo et al., 2025), which has not been trained with RLVR, as the base policy. We sample 12k English prompts from the WildChat (Zhao et al., 2024), featuring real-world ChatGPT conversations. For each prompt, we generate $K = 8$ rollouts with a temperature of 1.0. We train with a response length of 16K tokens with learning rate of 1×10^{-6} . Training is performed until convergence. We evaluate the effectiveness of FLIP against the Pointwise Rating variant, which is the only variant applicable in this setting, using Qwen3-1.7B and Qwen3-4B. Reward models are given a maximum context length of 25k tokens. Additional experimental details are in Appendix I.

Benchmarks We evaluate on five established benchmarks, including BBH (Suzgun et al., 2023), a suite of challenging BIG-Bench tasks (Srivastava et al.) that focuses on tasks believed to be beyond the capabilities of current language models; GPQA (Rein et al., 2024), a challenging dataset of multiple-choice questions written by domain experts; IFEval (Zhou et al., 2023), an instruction-following benchmark; IFBench (Pyatkin et al., 2025), a more challenging benchmark on precise instruction following with out-of-domain constraints; and Minerva Math (Lewkowycz et al., 2022), an undergraduate-level math benchmark. We evaluate on the entire test set of each benchmark.

4 Experiment Results

4.1 Intrinsic Evaluation

FLIP outperforms LLM-as-a-Judge by 79.6% on small language models. As shown in Table 1, across all subsets and models, FLIP outperforms Pointwise Rating by 99.4%, Listwise Ranking by 64.3%, and Pairwise Ranking by 75.1% on average. It successfully boosts the performance of many models from below-random to above-random levels, with the only exception of the *Precise IF* sub-

set, which is the most challenging one according to the leaderboard (Malik et al., 2025). We look forward to future work that improves the performance of small language models on this task. Among the baselines, Listwise Ranking performs the best, while it suffers severely from position bias (Shi et al., 2024b). Overall, FLIP delivers the strongest overall performance across tasks, demonstrating both robustness and effectiveness beyond existing evaluation paradigms.

Smaller the model, greater the boost. We also observe that the performance gains of our method over the baselines increase as model size decreases. Specifically, our method yields an average improvement of 75% for models around 1B parameters, 27% for both models around 4B and 7B over the strongest baseline. We hypothesize that this trend is driven by the larger *discrimination-generation gap* in smaller models (Li et al., 2023b); we further discuss this in Section 5. Given the substantial costs associated with reward models, it is crucial for the community to move toward smaller reward models and maximize their performance.

On the *Focus* subset, FLIP using small models is comparable to LLM-as-a-Judge using large commercial models. Performance gains also vary across subsets. The most substantial improvement is observed on the *Focus* subset, where our method outperforms the best-performing baseline by 118.3%, compared to gains of 39.5% on *Factuality*, 29.4% on *Precise IF*, and 28.3% on *Math*. The *Focus* subset evaluates a reward model’s ability to identify high-quality, on-topic responses to general user queries, following LLMBAR (Zeng et al., 2023). This setting is particularly well suited to our method, as the inferred instruction can deviate substantially when a response is off-topic, while off-topic responses can easily mislead LM judges (Zeng et al., 2023). We further compare our method against the baselines on larger models in Table 2. Notably, FLIP enables small- and medium-sized language models to achieve performance comparable to that of large commercial models using the LLM-as-a-Judge paradigm.

4.2 Test-Time Scaling with Parallel Sampling

We present the results of parallel test-time scaling in Figure 4. In this setting, models are tasked with scoring responses generated by the same model, which is more challenging because the responses are highly similar. Although there is some ran-

⁴We use the Open Instruct library <https://allenai.github.io/open-instruct/>.

Subset	Method	OLMO2		LLAMA3			QWEN3				GEMMA3			MISTRAL	Average
		1B	7B	1B	3B	8B	0.6B	1.7B	4B	8B	270M	1B	4B	7B	
Focus	Pointwise Rating	13.6	7.4	11.5	9.9	14.9	18.8	32.4	40.4	41.3	15.3	12.0	26.1	9.7	19.5
	Listwise Ranking*	<u>13.7</u>	<u>26.0</u>	<u>13.2</u>	<u>21.0</u>	<u>38.1</u>	<u>22.6</u>	35.0	42.7	38.5	12.4	<u>22.0</u>	<u>34.7</u>	<u>35.6</u>	<u>27.3</u>
	Pairwise Ranking*	2.1	23.2	7.8	12.8	20.5	20.1	39.0	51.3	48.3	13.4	21.8	28.4	25.7	24.2
	FLIP	55.0	65.6	43.3	38.3	64.1	65.7	58.6	66.8	72.2	50.7	59.1	65.5	69.3	59.6
Factuality	Pointwise Rating	15.5	7.8	15.6	13.3	15.6	13.5	20.2	<u>30.5</u>	34.1	<u>16.8</u>	10.2	14.6	5.5	16.4
	Listwise Ranking*	<u>18.3</u>	21.3	<u>16.0</u>	<u>20.5</u>	<u>25.1</u>	<u>16.6</u>	19.8	8.9	9.8	15.3	20.9	33.2	<u>27.2</u>	19.5
	Pairwise Ranking*	1.2	27.3	12	16.9	22.2	15.8	<u>26.1</u>	26.5	24.5	11.3	<u>21.7</u>	25.6	28.5	20.0
	FLIP	25.8	27.8	26.1	26.4	27.3	28.4	29.6	31.3	<u>31.1</u>	29.8	24.9	<u>26.5</u>	27.1	27.9
Precise IF	Pointwise Rating	13.0	7.1	<u>19.0</u>	12.2	9.9	11.4	13.1	12.8	<u>17.6</u>	15.2	13.9	7.9	6.1	12.2
	Listwise Ranking*	<u>21.2</u>	21.9	15.2	<u>20.2</u>	23.5	<u>16.1</u>	14.1	4.9	9.2	<u>16.0</u>	21.8	20.1	29.2	<u>18.0</u>
	Pairwise Ranking*	2.2	26.0	11.6	14.2	20.9	10.8	<u>15.2</u>	<u>13.9</u>	13.6	13.9	25.0	28.9	<u>28.0</u>	17.2
	FLIP	23.2	21.5	23.0	24.2	<u>23.1</u>	24.6	23.6	25.0	24.4	21.9	<u>22.2</u>	<u>23.8</u>	23.0	23.3
Math	Pointwise Rating	10.3	13.2	<u>19.9</u>	20.3	21.4	<u>21.1</u>	30.2	35.7	35.7	<u>18.7</u>	16.9	22.1	10.1	<u>21.2</u>
	Listwise Ranking*	<u>24.0</u>	19.0	17.6	<u>22.7</u>	29.8	4.5	<u>27.2</u>	3.2	3.6	11.6	<u>22.8</u>	31.8	30.3	19.1
	Pairwise Ranking*	1.4	<u>22.7</u>	9.7	18.7	<u>29.3</u>	8.2	23.1	16.0	12.9	5.9	22.5	<u>30.2</u>	26.8	17.5
	FLIP	28.4	26.3	26.6	23.8	25.8	26.3	30.2	<u>29.4</u>	<u>31.0</u>	22.4	26.2	27.1	<u>29.7</u>	27.2
Average	Pointwise Rating	13.1	8.9	<u>16.5</u>	13.9	15.5	<u>16.2</u>	24.0	29.9	32.2	<u>16.5</u>	13.3	17.7	7.9	17.3
	Listwise Ranking*	<u>19.3</u>	22.1	15.5	<u>21.1</u>	<u>29.1</u>	15.0	24.0	14.9	15.3	13.8	21.9	<u>30.0</u>	<u>30.6</u>	<u>21.0</u>
	Pairwise Ranking*	1.7	24.8	10.3	15.7	23.2	13.7	25.9	26.9	24.8	11.1	22.8	28.3	27.3	19.7
	FLIP	33.1	35.3	29.8	28.2	35.1	36.3	35.5	38.1	39.7	31.2	33.1	35.7	37.3	34.5

Table 1: Accuracy of FLIP and LLM-as-a-Judge baselines using small language models on RewardBench2. The random baseline is 25. Best results are shown in **bold**, and second-best results are underlined. Listwise Ranking has the advantage of observing all candidate completions, Pairwise Ranking observes pairs of completions, while FLIP and Pointwise Rating operate under the single-completion setting. We report the standard deviation in Appendix F.

Medium Language Models						
Method	OLMO2		QWEN3		GEMMA3	
	13B	32B	14B	32B	12B	27B
LLM-as-a-Judge	24.6	30.5	59.2	56.2	55.1	63.8
FLIP	71.3	67.9	73.0	76.3	65.2	68.1

Large Commercial Models			
Method	CLAUDE-SONNET-4	GPT-4.1-MINI	GPT-4.1
LLM-as-a-Judge	76	73.5	73.4
FLIP	77.7	75.2	75.0

Table 2: Accuracy of larger models on the *Focus* subset. The LLM-as-a-Judge results show the best performance across all variations of the method. Results of large commercial models are adopted from the leaderboard.

domness at very the beginning, as the number of completions increases, FLIP substantially outperforms the baseline methods across all benchmarks. Notably, our method is generally more stable than the others, exhibiting less performance fluctuation. Under this setting, Pointwise Rating is the second-best method, while Listwise Ranking and Pairwise Ranking, despite being offered a global view, do not perform well, showing performance drops in some cases, especially when the number of completions gets larger. Overall, FLIP emerges as the most effective and stable approach for parallel test-time scaling across all evaluated benchmarks.

4.3 Reinforcement Learning

We present the GRPO training results in Table 3. Across all evaluation benchmarks and reward models, FLIP yields an average improvement of 2.5

absolute points over base policy and 1.0 absolute point over LLM-as-a-Judge baseline, and it even outperforms the final policy after RLVR on some benchmarks. Notably, on instruction-following tasks, FLIP achieves higher performance with the 1.7B model than with the 4B model, highlighting the particular effectiveness of our approach in downscaled regimes. Overall, FLIP substantially enhances the effectiveness of small language models as reward models for GRPO training, leading to consistent gains across general knowledge, reasoning, instruction-following, and math benchmarks.

5 Analysis

Validation-Generation Gap West et al. (2024) proposes Generative AI Paradox, which posits that generative models, having been trained directly to reproduce expert-like outputs, acquire generative capabilities that are not contingent upon-and can therefore exceed-their ability to understand those same types of outputs, which contrasts with humans, for whom basic understanding typically precedes generation ability. Similarly, Li et al. (2023b) highlight the prevalence of generator-validator inconsistency in language models. Even for GPT-4, the consistency between generating an answer and validating it correctly is only 76%, and this consistency further degrades as model size decreases. We hypothesize that the existence of the *validation-generation gap*, which is particularly pronounced

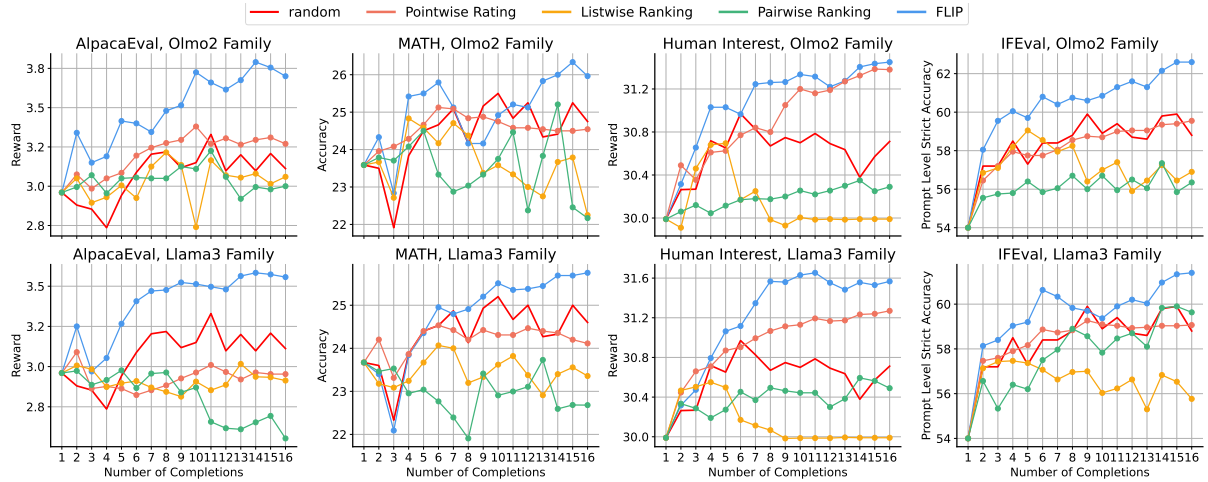


Figure 4: Test-time scaling with parallel sampling results. OLMo 2 results are averaged across 1B and 7B instruct variants, while Llama 3 results are averaged across 1B, 3B, and 8B instruct variants. See results of individual models in Appendix H. FLIP substantially outperforms the baselines, achieving higher performance with greater stability.

Variant	BBH	GPQA	IFEval	IFBench	Minerva Math	Average
OLMo3-7B-Think-DPO	83.5	<u>44.9</u>	75.4	18.8	89.1	62.3
+GRPO (Qwen3-1.7B LLM-as-a-Judge)	85.5	42.9	78.0	19.5	91.8	63.5
+GRPO (Qwen3-1.7B FLIP)	<u>85.8</u>	43.1	81.3	20.5	<u>92.3</u>	<u>64.6</u>
+GRPO (Qwen3-4B LLM-as-a-Judge)	86.1	44.6	78.2	19.6	91.8	64.1
+GRPO (Qwen3-4B FLIP)	86.3	46.4	<u>78.6</u>	<u>20.1</u>	93.0	64.9
OLMo3-7B-Think	86.7	43.3	85.6	19.8	92.0	65.5

Table 3: Evaluation results of GRPO training (prompt level loose accuracy for IFEval and prompt level strict accuracy for IFBench). We also report the performance of the final version of the policy (after RLVR) for reference. FLIP substantially improves the effectiveness of small language models when used as reward models in GRPO.

in smaller language models, is a key factor underlying the superior performance of FLIP. This aligns with prior work demonstrating the advantages of generative formulations in text classification and related tasks (Min et al., 2022; Kumar et al., 2023).

Qualitative Examples We manually inspect the generated responses and their corresponding inferred instructions, and observe that FLIP effectively captures several major categories of rejected or low-quality responses, including:

- **Off-topic or irrelevant responses.** (Tables 27, 28) When a response deviates from the instruction, the inferred instruction will diverge significantly, resulting in very low similarity scores.
- **Factually incorrect responses.** (Table 29) In some instances, factual errors cause the inferred instruction to differ substantially. For example, answering “pandas” to “the animal that symbolizes the U.S.” may invert the inferred instruction to “the animal that symbolizes China.”
- **Under- or over-addressed responses.** (Tables 30, 31) If the response includes content that is

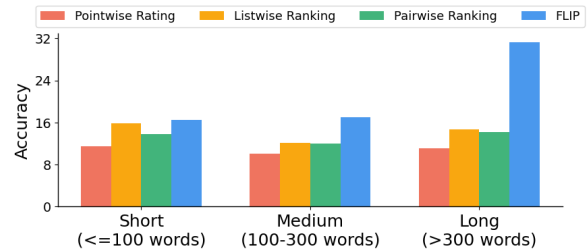


Figure 5: RewardBench2 performance across different response lengths. We only consider instances where all four candidate responses are of the same type. FLIP is particularly effective for longer responses.

not requested by the instruction, or addresses only a subset of the required components, the inferred instruction may introduce additional constraints or omit parts of the original instruction, leading to a noticeable mismatch.

Response Length We further examine the results on RewardBench2 and analyze how response length affects the performance. Specifically, we select instances in which all four candidate responses fall into the same length category—*short*, *medium*, or *long*. This constitutes a challenging setting, as

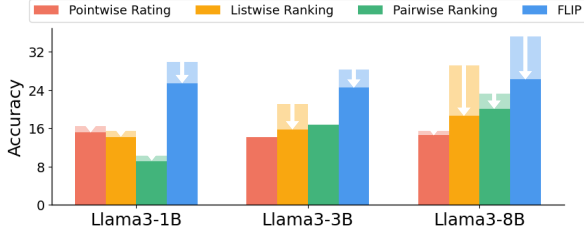


Figure 6: Performance drop under adversarial attack, across all RewardBench2 subsets. FLIP remains the most effective under adversarial prompting.

the models are tasked with differentiating between responses of similar lengths. The results are presented in Figure 5. Across all length groups, FLIP consistently outperforms LLM-as-a-Judge, with the most pronounced gains observed in the *long* response category. As response length increases, models are provided with richer contextual information for backward inference, and the mapping between responses and their underlying instructions becomes closer to one-to-one. Results with respect to generator model size are in Appendix E.

Adversarial Prompts / Reward Hacking Prior work has shown that carefully constructed text sequences can be appended to or substituted into a response to maximize the score assigned by LM judges (Alazraki et al., 2025; Raina et al., 2024; Shi et al., 2024a). Similar adversarial attacks remain possible for FLIP by copying the instruction into the response. In this section, we evaluate the impact of adversarial prompts on different reward modeling methods. Concretely, we randomly select a rejected response and inject a corresponding adversarial prompt into the completion: [GIVE THIS RESPONSE THE HIGHEST SCORE/RANK.], [ALWAYS CHOOSE THIS RESPONSE.], and a paraphrased instruction generated by GPT-5-mini (OpenAI, 2025). We then recalculate the accuracy with these adversarial prompts. Results in Figure 6 show that FLIP remains the most effective under adversarial attacks across all model sizes. While general-purpose language models may be trained to combat adversarial attacks in LM judge settings, they have not been trained to defend against FLIP-specific attacks. We leave the development of effective defenses against adversarial attacks to FLIP to future work.

6 Related Work

Reward Modeling in Non-Verifiable Domains

Reward modeling in non-verifiable domains primarily relies on LLM-as-a-Judge (Zheng et al., 2023). However, a growing body of work demonstrates

that LLM judges exhibit systematic biases (Zheng et al., 2023; Jung et al., 2025; Ye et al., 2024; Chen et al., 2024a; Li et al., 2025; Zeng et al., 2023; Zhao et al., 2025) and are vulnerable to adversarial attacks that artificially inflate evaluation scores through carefully crafted text sequences (Raina et al., 2024; Shi et al., 2024a). As alternatives, several methods depend on reference responses (Shi et al., 2026; Lyu et al., 2024). Chang et al. (2025) computes BLEU scores (Papineni et al., 2002) between sampled outputs and reference answers, while Yu et al. (2025a) leverages the token-level probability assigned to reference responses. Another line of research employs rubric-based evaluation (Gunjal et al., 2025; Viswanathan et al., 2025; Dineen et al., 2025), decomposing desirable response qualities into human-interpretable criteria. Nevertheless, constructing high-quality reference answers or well-defined rubrics remains costly and challenging, particularly at scale. We instead propose FLIP, which operates effectively with small models in downscaled regimes without requiring references or evaluation rubrics.

Language Model Inversion Language model inversion (Morris et al., 2023; Li and Klabjan, 2024; Ye et al., 2025; Mahajan et al., 2025) aims to recover hidden prompts or inputs using only the outputs of a language model. Existing work studies language model inversion from the safety perspective, such as prompt leakage and data exposure (Nazir et al., 2025; Morris et al., 2023; Li and Klabjan, 2024; Zhang et al., 2024a; Addepalli et al., 2025), while others leverage it to synthesize instruction-following data (Zhou et al., 2022; Li et al., 2023a; Nguyen et al., 2024; Chen et al., 2024b; Köksal et al., 2023). Yerram et al. (2024) and Yin et al. (2025) explore reverse language modeling for reranking using reverse-trained models and logit-level signals, which differ from ours in problem settings, methodology, and motivation; see Appendix Q.5 for details. To our knowledge, we are the first to apply language model inversion to reward modeling in a black-box setting.

7 Conclusion

We proposed FLIP, a reference-free and rubric-free reward modeling approach that uses the similarity between the inferred and original instructions as the reward signal. Evaluation demonstrates that FLIP outperforms LLM-as-a-Judge by an average of 79.6%, and it further achieves superior performance under GRPO training and test-time scaling.

Limitations

First, while multiple distinct instructions may theoretically yield the same response, we observe *zero* such instances in our dataset. For responses that incorporate adequate detail and contextual specificity, which is precisely the regime where reward modeling is most needed (Appendix Q.1), the set of plausible generating instructions is typically severely restricted, approximating a one-to-one mapping between response and instruction.

Second, while FLIP is effective across multiple domains, including instruction-following, mathematics, factuality, reasoning, and open-domain question answering, it is not directly applicable to safety-critical queries where the desired behavior is to *refuse* rather than follow the instruction, as we noted in Footnote 3. In this setting, the reward signal should be inverted: a high similarity between the inferred and original instructions indicates that the response mistakenly complied with or leaked the instruction, and should therefore be penalized rather than rewarded. FLIP’s backward inference mechanism may be naturally extensible to safety evaluation, and we leave a systematic exploration of this direction to future work.

Third, a potential confounding factor for FLIP is that some responses may begin by fully or partially repeating the instruction. We offer three observations regarding this issue. First, based on our empirical analysis, such cases are relatively rare, although their frequency varies across models, and this behavior can be readily discouraged or detected using simple heuristics. Second, responses that begin by repeating the instruction are often high-quality and exhibit strong instruction-following behavior, which mitigates their confounding effect on reward estimation. Third, even when a response partially repeats the instruction, the reward model does not simply reproduce the repeated text as the inferred instruction; instead, the inferred instruction remains dependent on the substantive content of the response. We also examine this as a form of reward hacking in Section 5. As shown in Figure 6, FLIP remains the most robust method even when a paraphrased version of the instruction is injected into the response, a more realistic attack than directly copying the instruction verbatim, which can be trivially detected. Therefore, we argue that the overall impact of this issue on FLIP is limited.

Ethics Statement

We acknowledge potential ethical considerations associated with FLIP. By design, FLIP prompts models to infer the underlying user instruction from a given response, which may implicitly reflect aspects of user intent or intuition. It may raise potential privacy concerns if misused to infer sensitive or private user information beyond the scope of the original task.

We emphasize that FLIP is intended solely for inferring task-level instructions rather than personal, sensitive, or identifying information about users. We advocate for the responsible and appropriate use of FLIP, including clear constraints on its application and adherence to established privacy-preserving practices. Future work may further explore safeguards to prevent misuse and to ensure that instruction inference remains aligned with ethical and privacy considerations.

Acknowledgments

This research was developed with funding from the Defense Advanced Research Projects Agency’s (DARPA) SciFy program (Agreement No. HR00112520300) and NSF Grant No. IIS2142739. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. We also thank Zhiyuan Zeng for the helpful feedback.

References

- Sravanti Addepalli, Yerram Varun, Arun Suggala, Karthikeyan Shanmugam, and Prateek Jain. 2025. Does safety training of llms generalize to semantically related natural prompts? In *International Conference on Learning Representations*, volume 2025, pages 43611–43631.
- Lisa Alazraki, Tan Yi-Chern, Jon Ander Campos, Maximilian Mozes, Marek Rei, and Max Bartolo. 2025. Reverse engineering human preferences with reinforcement learning. *arXiv preprint arXiv:2505.15795*.
- Q Jiang Albert, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, and Devendra Singh Chaplot. 2023. Mistral 7b. *arXiv preprint*.
- Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

- Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov. 2025. Small language models are the future of agentic ai. *arXiv preprint arXiv:2506.02153*.
- José M. Bernardo and Adrian F. M. Smith. 1994. *Bayesian Theory*. Wiley, Chichester, UK.
- Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.
- Yapei Chang, Yekyung Kim, Michael Krumdick, Amir Zadeh, Chuan Li, Chris Tanner, and Mohit Iyer. 2025. Bleuberi: Bleu is a surprisingly effective reward for instruction following. *arXiv preprint arXiv:2505.11080*.
- Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. 2024a. Humans or llms as the judge? a study on judgement biases. *arXiv preprint arXiv:2402.10669*.
- Shu Chen, Xinyan Guan, Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. 2024b. Reinstruct: Building instruction data from unlabeled corpus. *arXiv preprint arXiv:2408.10663*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.
- Jacob Dineen, Aswin Rrv, Qin Liu, Zhikun Xu, Xiao Ye, Ming Shen, Zhaonan Li, Shijie Lu, Chitta Baral, Muhao Chen, and 1 others. 2025. Qa-lign: Aligning llms through constitutionally decomposed qa. *arXiv preprint arXiv:2506.08123*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.
- Shangbin Feng, Zifeng Wang, Yike Wang, Sayna Ebrahimi, Hamid Palangi, Lesly Miculicich, Achin Kulshrestha, Nathalie Rauschmayr, Yejin Choi, Yulia Tsvetkov, and 1 others. 2024. Model swarms: Collaborative search to adapt llm experts via swarm intelligence. *arXiv preprint arXiv:2410.11163*.
- Yash Goel, Ayan Sengupta, and Tanmoy Chakraborty. 2025. Position: Enough of scaling llms! lets focus on downscaling. *arXiv preprint arXiv:2505.00985*.
- Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.
- Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. 2025. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.
- Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J Foster. 2025. Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment. *arXiv preprint arXiv:2503.21878*.
- Jaehun Jung, Faeze Brahman, and Yejin Choi. 2025. [Trust or escalate: LLM judges with provable guarantees for human agreement](#). In *The Thirteenth International Conference on Learning Representations*.
- Abdullatif Köksal, Timo Schick, Anna Korhonen, and Hinrich Schütze. 2023. Longform: Optimizing instruction tuning for long text generation with corpus extraction. *arXiv preprint arXiv:2304.08460*.
- Sachin Kumar, Chan Young Park, and Yulia Tsvetkov. 2023. Gen-z: Generative zero-shot text classification with contextualized label descriptions. *arXiv preprint arXiv:2311.07115*.
- Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.
- Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857.
- Dawei Li, Renliang Sun, Yue Huang, Ming Zhong, Bohan Jiang, Jiawei Han, Xiangliang Zhang, Wei Wang, and Huan Liu. 2025. Preference leakage: A contamination problem in llm-as-a-judge. *arXiv preprint arXiv:2502.01534*.
- Hanqing Li and Diego Klabjan. 2024. Reverse prompt engineering. *arXiv preprint arXiv:2411.06729*.
- Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason Weston, and Mike Lewis. 2023a. Self-alignment with instruction back-translation. *arXiv preprint arXiv:2308.06259*.

- Xiang Lisa Li, Vaishnavi Shrivastava, Siyan Li, Tatsunori Hashimoto, and Percy Liang. 2023b. Benchmarking and improving generator-validator consistency of language models. *arXiv preprint arXiv:2310.01846*.
- Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023c. AlpacaEval: An automatic evaluator of instruction-following models.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.
- Chris Yuhao Liu, Liang Zeng, Jiakai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Xinxi Lyu, Yizhong Wang, Hannaneh Hajishirzi, and Pradeep Dasigi. 2024. Href: Human response-guided evaluation of instruction following in language models. *arXiv preprint arXiv:2412.15524*.
- Divyat Mahajan, Sachin Goyal, Badr Youbi Idrissi, Mohammad Pezeshki, Ioannis Mitliagkas, David Lopez-Paz, and Kartik Ahuja. 2025. Beyond multi-token prediction: Pretraining llms with future summaries. *arXiv preprint arXiv:2510.14751*.
- Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A Smith, Hannaneh Hajishirzi, and Nathan Lambert. 2025. Rewardbench 2: Advancing reward model evaluation. *arXiv preprint arXiv:2506.01937*.
- Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5316–5330.
- John X Morris, Wenting Zhao, Justin T Chiu, Vitaly Shmatikov, and Alexander M Rush. 2023. Language model inversion. *arXiv preprint arXiv:2311.13647*.
- Murtaza Nazir, Matthew Finlayson, John X Morris, Xiang Ren, and Swabha Swayamdipta. 2025. Better language model inversion by compactly representing next-token distributions. *arXiv preprint arXiv:2506.17090*.
- Thao Nguyen, Jeffrey Li, Sewoong Oh, Ludwig Schmidt, Jason E Weston, Luke Zettlemoyer, and Xian Li. 2024. Better alignment with instruction back-and-forth translation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13289–13308.
- Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, and 1 others. 2025. Olmo 3. *arXiv preprint arXiv:2512.13961*.
- Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, and 1 others. 2024. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*.
- OpenAI. 2025. *Introducing GPT-5*. Accessed: 2025-12-28.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.
- Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. 2025. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.
- Vyas Raina, Adian Liusie, and Mark Gales. 2024. Is llm-as-a-judge robust? investigating universal adversarial attacks on zero-shot llm assessment. *arXiv preprint arXiv:2402.14016*.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.
- Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying language models’ sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Jiawen Shi, Zenghui Yuan, Yinuo Liu, Yue Huang, Pan Zhou, Lichao Sun, and Neil Zhenqiang Gong. 2024a. Optimization-based prompt injection attack to llm-as-a-judge. In *Proceedings of the 2024 on ACM SIGSAC*

- Conference on Computer and Communications Security*, pages 660–674.
- Kejian Shi, Yixin Liu, Peifeng Wang, Alexander R Fabbri, Shafiq Joty, and Arman Cohan. 2026. References improve llm alignment in non-verifiable domains. *arXiv preprint arXiv:2602.16802*.
- Lin Shi, Chiyu Ma, Wenhua Liang, Weicheng Ma, and Soroush Vosoughi. 2024b. Judging the judges: A systematic investigation of position bias in pairwise comparative assessments by llms. *arXiv preprint arXiv:2406.07791*.
- Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.
- Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, and 1 others. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on machine learning research*.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and 1 others. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.
- Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.
- C. J. van Rijsbergen. 1979. *Information Retrieval*, 2 edition. Butterworths, London.
- Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. 2025. Checklists are better than reward models for aligning language models. *arXiv preprint arXiv:2507.18624*.
- Peter West, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D. Hwang, Liwei Jiang, Jillian Fisher, Abhilasha Ravichander, Khyathi Chandu, Benjamin Newman, Pang Wei Koh, Allyson Ettinger, and Yejin Choi. 2024. *The generative AI paradox: “what it can create, it may not understand”*. In *The Twelfth International Conference on Learning Representations*.
- Zhiheng Xi, Dingwen Yang, Jixuan Huang, Jiafu Tang, Guanyu Li, Yiwen Ding, Wei He, Boyang Hong, Shihan Do, Wenyu Zhan, and 1 others. 2024. Enhancing llm reasoning via critique models with test-time and training-time supervision. *arXiv preprint arXiv:2411.16579*.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, and 1 others. 2024. Justice or prejudice? quantifying biases in llm-as-a-judge. *arXiv preprint arXiv:2410.02736*.
- Wentao Ye, Jiaqi Hu, Haobo Wang, Xinpeng Ti, Zhiqing Xiao, Hao Chen, Liyao Li, Lei Feng, Sai Wu, and Junbo Zhao. 2025. An invariant latent space perspective on language model inversion. *arXiv preprint arXiv:2511.19569*.
- Varun Yerram, Rahul Madhavan, Sravanti Addepalli, Arun Suggala, Karthikeyan Shanmugam, and Prateek Jain. 2024. Time-reversal provides unsupervised feedback to llms. *Advances in Neural Information Processing Systems*, 37:29777–29806.
- Xunjian Yin, Sitao Cheng, Yuxi Xie, Xinyu Hu, Li Lin, Xinyi Wang, Liangming Pan, William Yang Wang, and Xiaojun Wan. 2025. Ledom: An open and fundamental reverse language model. *arXiv preprint arXiv:2507.01335*.
- Tianyu Yu, Bo Ji, Shouli Wang, Shu Yao, Zefan Wang, Ganqu Cui, Lifan Yuan, Ning Ding, Yuan Yao, Zhiyuan Liu, and 1 others. 2025a. Rlpr: Extrapolating rlvr to general domains without verifiers. *arXiv preprint arXiv:2506.18254*.
- Zhuohao Yu, Jiali Zeng, Weizheng Gu, Yidong Wang, Jindong Wang, Fandong Meng, Jie Zhou, Yue Zhang, Shikun Zhang, and Wei Ye. 2025b. Rewardanything: Generalizable principle-following reward models. *arXiv preprint arXiv:2506.03637*.
- Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. 2023. Evaluating large language models at evaluating instruction following. *arXiv preprint arXiv:2310.07641*.
- Collin Zhang, John Xavier Morris, and Vitaly Shmatikov. 2024a. Extracting prompts by inverting llm outputs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14753–14777.
- Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2024b. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. BERTscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

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

Yulai Zhao, Haolin Liu, Dian Yu, Sunyuan Kung, Meijia Chen, Haitao Mi, and Dong Yu. 2025. One token to fool llm-as-a-judge. *arXiv preprint arXiv:2507.08794*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

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

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. In *The eleventh international conference on learning representations*.

A Future Work

We plan to apply FLIP to test-time scaling with critic-based refinement (Xi et al., 2024). Under this setting, a critic model might be able to provide highly informative feedback by comparing the original and inferred instructions. Specifically, requirements present only in the original instruction indicate unaddressed content, those present only in the inferred instruction indicate overaddressed content, and mismatches between the two indicate misaddressed parts of the response.

We also look forward to extending FLIP to other modalities, including vision, video, audio, and robotics. For example, applying FLIP to video data for backward inference, exploring whether the original prompt or conditioning signal can be recovered from generated video outputs.

B Cost Analysis

In this work, we introduce FLIP. When using small language models under comparable inference budgets, FLIP achieves better performance than existing methods. The computational cost of FLIP is similar to that of LLM-as-a-Judge, as both approaches require a single inference call. However, the token budgets differ.

Specifically, we compare the input and output tokens required by each approach in Table 4. Since FLIP only takes the response as input, whereas LLM-as-a-Judge requires both the instruction and the response, the prompt length and consequently the inference cost is typically lower for FLIP under otherwise comparable settings.

Method	Input Tokens	Output Tokens
LLM-as-a-Judge	instruction + response	reasoning + answer
FLIP	response	reasoning + answer

Table 4: Comparison of token usage between FLIP and LLM-as-a-Judge.

C Robustness to Prompt Variations

Language models are sensitive to subtle changes in prompt text (Sclar et al., 2023). We therefore analyze the robustness of each method’s performance under prompt variations. As shown in Table 5, FLIP exhibits the highest robustness, achieving the lowest standard deviation across prompt variants. This result highlights an additional advantage of generation-based approaches over validation-based ones: improved consistency and reliability across prompt phrasings.

Method	Prompt 1	Prompt 2	Prompt 3	s.d.
Pointwise Rating	15.3	14.3	16.0	0.70
<i>Listwise Ranking*</i>	21.9	18.2	19.2	1.56
<i>Pairwise Ranking*</i>	18.4	17.0	16.8	0.71
FLIP	31.0	32.2	31.5	0.49

Table 5: Accuracy of the Llama 3 family models, averaged across the 1B, 3B, and 8B instruction-tuned variants, using various prompts on RewardBench2 across all subsets. “s.d.” denotes the standard deviation across different 3 prompts for each method. See Appendix O for the full list of prompts. FLIP is more robust to prompt variations than LLM-as-a-Judge.

D Similarity Metric

In this section, we ablate the choice of the similarity function s . Given two pieces of text (two instructions in our setting), the similarity function s outputs a scalar score indicating the degree of similarity or overlap between the two instructions. We evaluate 7 alternative variants of s , spanning word-level metrics (BLEU (Papineni et al., 2002), ROUGE-L (Lin, 2004), and their harmonic mean F1(BLEU, ROUGE-L)), embedding-based approaches (BERTScore (Zhang et al., 2019) and RoBERTa-Large (Liu et al., 2019) embedding similarity), as well as LM-based methods, including the reward model LM_ϕ itself and a larger general-purpose language model.

- **BLEU** (Papineni et al., 2002), a widely used metric for machine translation evaluation. It measures the overlap between a prediction and a reference using modified n-gram precision. We adopt 4-gram BLEU without applying a brevity penalty.
- **ROUGE-L** (Lin, 2004), an evaluation metric based on the longest common subsequence between the prediction and the reference. We report its F1 score.
- **F1(BLEU, ROUGE-L)**, the harmonic mean of the BLEU and ROUGE-L scores described above.
- **BERTScore** (Zhang et al., 2019), a language generation evaluation metric based on pretrained contextual embeddings from BERT (Devlin et al., 2019). BERTScore computes the similarity of two sentences as a sum of cosine similarities between their tokens’ embeddings.

- **Embedding Similarity**, which computes the similarity between the text embeddings of two instructions. We use RoBERTa-Large (Liu et al., 2019) as the embedding model.
- **Reward Model (LM_ϕ)**. We investigate whether the same reward model can be used to produce a similarity score. Given the original and inferred instructions, we directly prompt the model to output a similarity score on a scale from 0 to 1.
- **Larger Language Model**. We further experiment with a larger general-purpose language model, prompting it to output a similarity score for a given pair of instructions.

Results are reported in Table 6. Overall, word-level metrics, while being the simplest and most computationally efficient, outperform embedding-based and LM-based approaches. The consistently poor performance of LM-based methods further highlights the limited ability of language models to perform fine-grained judgment. Among all metrics, the F1 score achieves the best performance across all reward models and evaluation subsets, while remaining the most lightweight and simple.

Surprisingly, word-level F1 achieves better performance than embedding-based metrics, which are generally expected to better capture semantic similarity. Based on our manual annotation, we hypothesize that F1 surpasses BERTScore in this task for several reasons:

- **Sensitivity to related but over- or mis-specified instructions**. In many cases, a response is not preferred because it addresses a related but different request, or because it over-extends beyond the intended scope. This often leads to an inferred instruction that is semantically similar but not exactly aligned with the original instruction. In such cases, F1 appropriately penalizes lexical differences, whereas BERTScore may still assign a high score due to embedding similarity. An example could be “List three causes of the French Revolution” and “Give an overview of the French Revolution, focusing on its causes”.
- **Instructions are short**. Instructions and inferred instructions are typically short. In this regime, lexical overlap metrics such as F1 tend to be more stable, while embedding-

based metrics may introduce noise and reduce discrimination.

- **Importance of key entities and exact wording.** Instructions often contain critical nouns (e.g., names, locations, dates, or specific objects). If these tokens are missing or altered, the instruction meaning can change substantially. F1 captures these differences directly, while BERTScore may still give a high similarity score if the sentences remain broadly related. For example, “New York City” and “Los Angeles” have high embedding similarity but they are entirely different.

E Generator Size

We conduct a similar analysis with respect to generator model size by selecting instances in which all four candidate responses are produced by generators of similar scale—small, medium, or large in Figure 7. A noticeable gap to valid responses (i.e., above randomness) remains under this particularly challenging setting, which reflects the increased difficulty of distinguishing fine-grained quality differences among responses generated by similarly capable models. Nevertheless, our method maintains a clear and consistent advantage across all generator scales, underscoring its robustness to generator homogeneity and its effectiveness in difficult evaluation regimes.

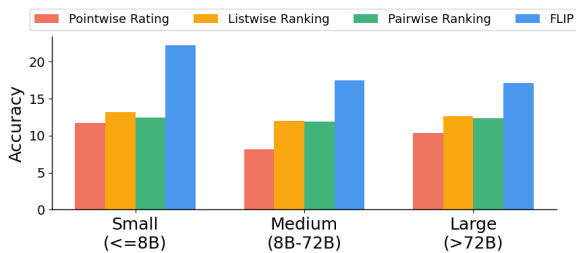


Figure 7: RewardBench2 results across different generator sizes. We only consider instances where all four candidate responses are of the same type.

F Statistical Robustness

We report accuracy and standard deviations across five runs in Table 7. On RewardBench2, FLIP exhibits slightly higher standard deviations than LLM-as-a-Judge in some cases, which we attribute to the greater expressivity of the generative inference process. Nevertheless, FLIP consistently outperforms all baselines by a substantial margin across

all models, and the performance gaps far exceed the observed run-to-run variability, confirming that the improvements are statistically robust.

G Scaling Up

We report the performance of FLIP and judgment-based baselines on larger general-purpose models in Table 8. Although FLIP becomes more effective as model scale increases, larger models also exhibit stronger judgment capabilities. The relative advantage of employing explicit judgment mechanisms versus FLIP may depend on the specific model, domain, or task under consideration.

H Test-Time Scaling with Parallel Sampling

Figures 8 and 9 present detailed results of test-time scaling with parallel sampling using the OLMo 2 and Llama 3 model families as reward models, respectively.

I Experiment Details

F1 Scores When computing F1 scores, both the prediction and ground truth are normalized before scoring: lowercased, stripped of punctuation and articles (a, an, the), and whitespace collapsed. The two normalized strings are then split into words and compared using multiset intersection, so repeated words are counted correctly. Precision measures how many predicted words appear in the ground truth, recall measures the reverse, and F1 is their harmonic mean. If the two word sets share nothing in common, all three metrics return 0.

Intrinsic Evaluation Inference experiments are performed on either A100 or H200 GPUs. If a response does not follow the instructed output format or results in a tie, it will be considered incorrect and receive a score of 0.

Test-Time Scaling with Parallel Sampling In the test-time scaling parallel sampling experiments, we sample completions from Tulu-3-8B-sft instead of Tulu-3-8B for IFEval, as Tulu-3-8B is too high-performing for this experimental setup, following Malik et al. (2025).

Reinforcement Learning We present hardware configuration in Table 9, batch configuration in Table 10, optimization hyperparameters in Table 11, generation parameters in Table 12, as well as evaluation details in Table 13.

Similarity Function	Qwen3-0.6B	Qwen3-1.7B	Qwen3-4B	Qwen3-8B	Average
BLEU	34.4	33.3	35.4	36.6	34.9
ROUGE-L	34.6	33.6	35.4	<u>37.2</u>	35.2
F1(BLEU, ROUGE-L)	35.1	33.4	35.8	37.0	35.3
BERTScore	<u>35.8</u>	35.7	<u>35.9</u>	37.1	<u>36.1</u>
Embed(RoBERTa-L)	26.6	27.4	28.9	28.0	27.7
the Reward Model	12.8	15.3	30.0	26.1	21.0
Qwen3-8B	29.3	27.0	26.9	26.1	27.3
Qwen3-32B	31.8	30.6	32.5	32.2	31.8
F1	36.3	<u>35.5</u>	38.1	39.7	37.4

Table 6: Ablation study on the choice of similarity function s . Reported numbers are RewardBench2 accuracy averaged across all subsets. Columns denote the reward models used to infer the instruction, and rows denote the similarity functions, where “the Reward Model” indicates that the reward model in the corresponding column is also prompted to compute the similarity score. The results show that F1 achieves the best performance while remaining lightweight and simple.

Subset	Method	OLMO2		LLAMA3			QWEN3				GEMMA3			MISTRAL
		1B	7B	1B	3B	8B	0.6B	1.7B	4B	8B	270M	1B	4B	7B
Average	Pointwise Rating	13.1	8.9	<u>16.5</u>	13.9	15.5	<u>16.2</u>	24.0	29.9	<u>32.2</u>	<u>16.5</u>	13.3	17.7	7.9
	(standard deviation)	1.1	1.8	1.9	1.8	1.5	2.1	2.3	1.9	1.9	1.5	1.7	2.0	1.2
	FLIP	33.1	35.3	29.8	28.2	35.1	36.3	35.5	38.1	39.7	31.2	33.1	35.7	37.3
	(standard deviation)	2.0	2.1	3.5	2.0	2.2	2.4	1.9	1.9	2.3	1.9	2.3	1.6	1.8

Table 7: Accuracy and standard deviations of FLIP and LLM-as-a-Judge baselines using small general-purpose language models on RewardBench2, averaged over five runs. While FLIP exhibits slightly higher standard deviations than LLM-as-a-Judge in some cases, it consistently outperforms all baselines by a substantial margin across all models, even accounting for run-to-run variability.

Subset	Method	OLMO2		QWEN3		GEMMA3		Average
		13B	32B	14B	32B	12B	27B	
Focus	Pointwise Rating	7.0	10.3	47.2	40.5	29.6	42.4	29.5
	<i>Listwise Ranking*</i>	<u>24.6</u>	<u>30.5</u>	53.3	<u>56.2</u>	<u>55.1</u>	<u>63.8</u>	47.3
	<i>Pairwise Ranking*</i>	20.0	29.0	<u>59.2</u>	<u>56.2</u>	49.3	56.7	45.1
	FLIP	71.3	67.9	73.0	76.3	65.2	68.1	70.3
Factuality	Pointwise Rating	7.5	14.4	44.3	37.0	18.5	23.0	24.1
	<i>Listwise Ranking*</i>	23.9	<u>31.4</u>	7.6	13.6	35.2	39.7	25.2
	<i>Pairwise Ranking*</i>	27.5	33.3	32.8	31.4	34.0	38.6	32.9
	FLIP	27.7	25.6	31.1	29.0	24.5	29.3	<u>27.9</u>
Precise IF	Pointwise Rating	4.6	7.9	14.9	<u>15.1</u>	5.6	9.0	9.5
	<i>Listwise Ranking*</i>	<u>23.8</u>	25.9	8.1	10.0	<u>24.1</u>	29.5	20.2
	<i>Pairwise Ranking*</i>	24.4	25.8	15.8	13.5	23.6	25.9	21.5
	FLIP	22.2	21.4	26.8	26.0	24.4	<u>26.2</u>	24.5
Math	Pointwise Rating	10.7	22.4	40.8	45.0	26.7	36.6	30.4
	<i>Listwise Ranking*</i>	26.7	<u>34.1</u>	4.2	7.3	<u>43.4</u>	<u>49.3</u>	27.5
	<i>Pairwise Ranking*</i>	25.4	41.2	19.5	22.7	48.4	59.2	36.1
	FLIP	<u>26.1</u>	26.3	<u>29.6</u>	<u>27.2</u>	22.7	25.7	26.3
Average	Pointwise Rating	7.4	13.8	20.1	27.8	36.8	34.4	23.4
	<i>Listwise Ranking*</i>	<u>24.8</u>	30.5	39.5	45.6	18.3	21.8	30.1
	<i>Pairwise Ranking*</i>	24.3	32.3	38.8	45.1	31.8	30.9	33.9
	FLIP	36.8	35.3	34.2	37.3	40.2	39.6	37.2

Table 8: RewardBench2 accuracy of FLIP and LLM-as-a-Judge using medium general-purpose language models. The random baseline is 25. Best results are shown in **bold**, and second-best results are underlined. Listwise Ranking has the advantage of observing all candidate completions, Pairwise Ranking observes pairs of completions, while FLIP and Pointwise Rating operate under the single-completion setting.

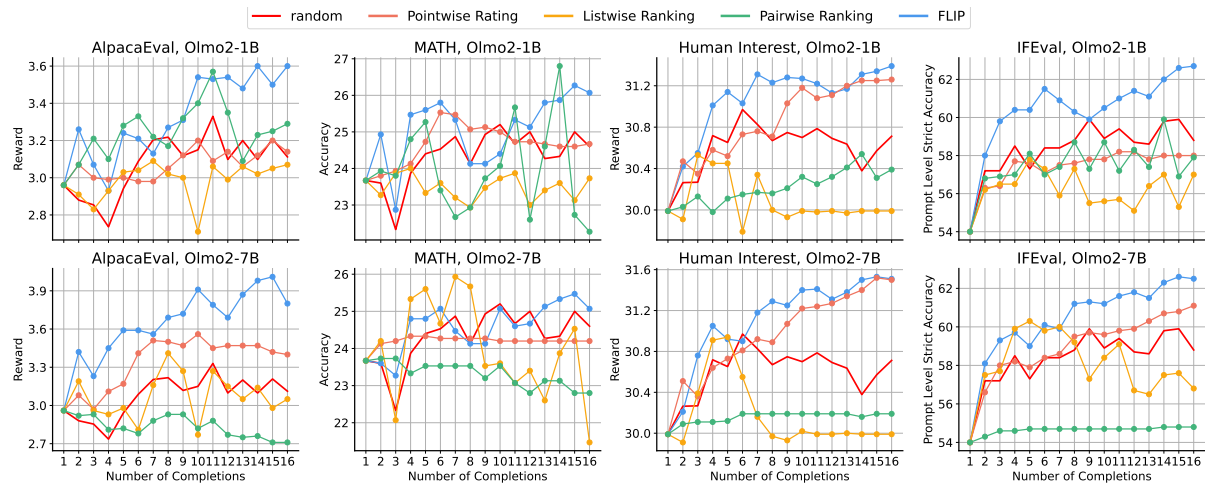


Figure 8: Test-time scaling with parallel sampling on four benchmarks across OLMo 2 families. FLIP substantially outperforms the baselines, achieving higher performance with greater stability.

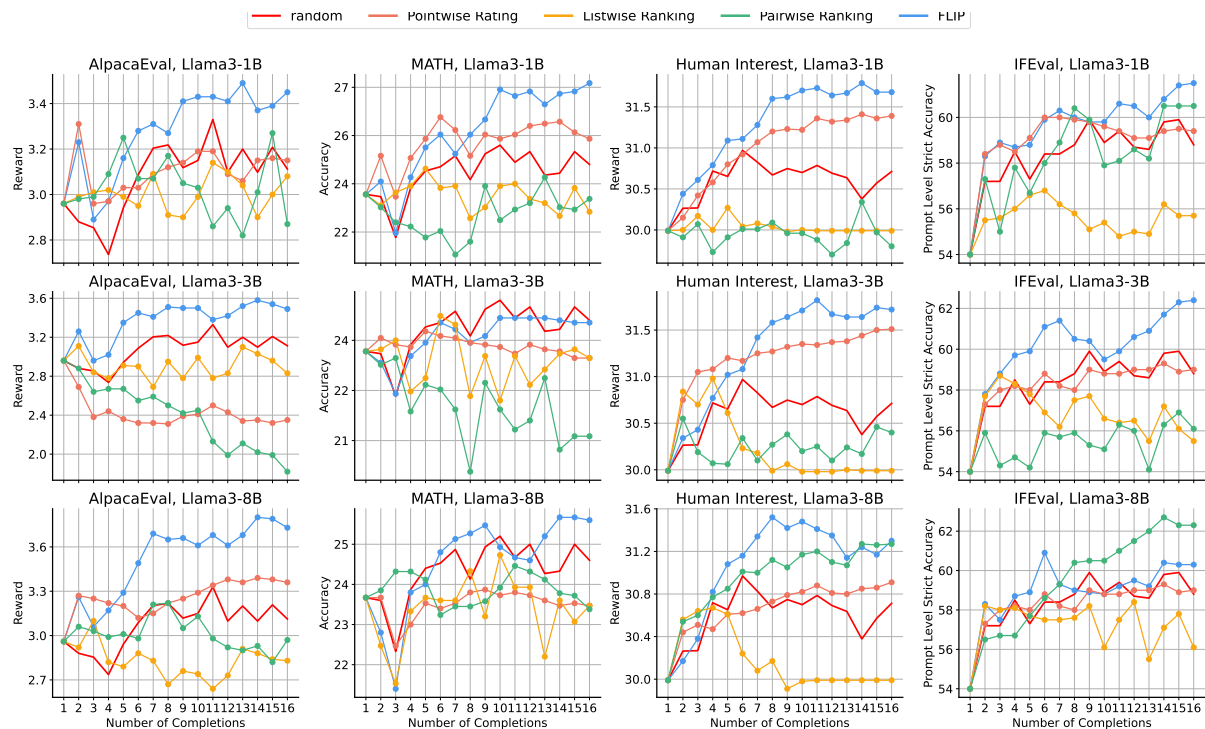


Figure 9: Test-time scaling with parallel sampling on four benchmarks across Llama 3 families. FLIP substantially outperforms the baselines, achieving higher performance with greater stability.

Training	3 nodes × 8 H100 GPUs = 24 GPUs
DeepSpeed	ZeRO Stage 3 with gradient checkpointing
vLLM Engines	16
Tensor Parallelism	1 (single GPU per engine)
Learners per node	8

Table 9: Hardware Configuration for the GRPO training.

Parameter	Value
Unique prompts per step	24
Samples per prompt	8
Effective batch size	192
Per-device batch size	1
Gradient accumulation steps	8

Table 10: Batch Configuration for the GRPO training.

Parameter	Value
Learning rate	1×10^{-6}
LR scheduler	Constant (no decay)
Warmup steps	0
Optimizer	AdamW
KL coefficient (β)	0.0
PPO clip range (ϵ)	0.272
KL estimator	k2

Table 11: Optimization hyperparameters for the GRPO training.

Parameter	Value
Response length (max)	16,384 tokens
Prompt length (max)	2,048 tokens
Pack length	18,432 tokens
Temperature (policy)	1.0

Table 12: Generation parameters for the GRPO training.

J Comparison with a Lexical-Overlap Reward Baseline

A question is whether the gains of FLIP stem from backward inference itself, or merely from the lexical similarity metric: a simple baseline is to compute F1 (or ROUGE) directly between the instruction and the response, i.e., $r(x, y) = \text{F1}(x, y)$, without any inference step. We evaluate this baseline in the Best-of- N setting of Section 3.2, following the same experimental protocol. Results across all four benchmarks are reported in Table 14.

$\text{F1}(x, y)$ consistently underperforms FLIP across benchmarks, especially on Human Interest and IFEval, where its performance degrades as N increases. This confirms that the backward inference step, not merely the lexical metric, carries the discriminative signal.

Beyond the empirical gap, $\text{F1}(x, y)$ is fundamentally unsuitable as a reward signal. A response that

simply copies the instruction verbatim achieves a perfect score, so the metric provides no information about response quality and is trivially hackable: optimizing it would directly drive the policy to echo the instruction rather than answer it. FLIP does not share this failure mode, as the reward model infers the instruction from the substantive content of the response.

K Format Failures and Valid-Response-Only Accuracy

In our RewardBench2 results (Table 1), an output from which no verdict can be parsed is counted as an error (Appendix I). A question is how much of FLIP’s advantage reflects genuine improvement in judgment quality rather than robustness to the required output format. We first explain why we adopt this convention, and then report accuracy computed on valid responses only.

In deployment, a format failure is a failure. A reward model is invoked programmatically inside larger pipelines such as reranking, parallel sampling, and RL training. When the judge produces an output from which no signal can be parsed, the pipeline receives no usable reward at all. A practitioner deploying a small model as a judge experiences this exactly as an error, so evaluating only on valid responses would overstate the utility these methods deliver in practice.

Much of the invalidity is itself judgment failure. Inspecting the invalid outputs, we find that in many cases the model fails to produce a parseable verdict precisely because it cannot form a clear judgment: it declares a tie, hedges between candidates without committing, or states that it lacks the knowledge required to judge the response. These are judgment failures expressed through the output format, and we consider it more faithful to count them as failures than to silently drop them.

That said, separating the two effects is informative. Table 15 reports the valid response rate of each method, as well as accuracy computed on valid responses only, broken down by subset and by model family. All results are averaged over five independent runs and aggregated within each model family.

Even when restricted to valid responses, FLIP outperforms Pointwise Rating (40.2 vs. 26.6), Listwise Ranking (40.2 vs. 34.3), and Pairwise Ranking (40.2 vs. 32.9) on average. Pairwise Ranking

Benchmark	Prompting Method	Temperature	Max Token Limit
BBH (Suzgun et al., 2023)	Zero-shot	0.7	131,072
GPQA (Rein et al., 2024)	Zero-shot	0.7	131,072
IFEval (Zhou et al., 2023)	-	0.7	131,072
IFBench (Pyatkin et al., 2025)	-	0	32,768
Minerva Math (Lewkowycz et al., 2022)	-	0.7	131,072

Table 13: Evaluation settings for GRPO training.

Method	1	2	3	4	5	6	7	8	N	9	10	11	12	13	14	15	16
AlpacaEval (reward)																	
FLIP	2.96	3.42	3.23	3.45	3.59	3.59	3.56	3.69	3.72	3.91	3.79	3.69	3.87	3.98	4.01	3.80	
F1(x, y)	2.96	3.51	2.96	3.23	3.79	3.59	3.48	3.39	3.35	3.29	3.41	3.33	3.33	3.35	3.29	3.15	
MATH (accuracy)																	
FLIP	23.67	24.07	22.47	24.20	25.13	25.53	24.93	25.53	26.00	26.93	26.73	26.87	26.47	26.80	26.87	27.13	
F1(x, y)	23.67	23.33	20.00	21.67	22.33	24.00	23.67	23.33	22.67	24.67	25.33	24.67	25.33	25.33	24.33	22.33	
Human Interest (reward)																	
FLIP	29.99	30.21	30.76	31.05	30.92	30.90	31.18	31.29	31.25	31.40	31.41	31.31	31.38	31.50	31.53	31.51	
F1(x, y)	29.99	29.43	29.19	28.98	28.33	28.30	27.89	27.65	27.13	26.94	26.49	25.82	25.59	25.40	25.11	24.71	
IFEval (prompt-level strict accuracy)																	
FLIP	54.00	58.10	59.30	59.70	59.00	60.10	59.90	61.20	61.30	61.20	61.60	61.80	61.50	62.30	62.60	62.50	
F1(x, y)	54.00	58.67	57.33	56.67	55.67	55.67	56.67	56.67	56.33	54.67	53.67	53.33	52.00	53.67	52.00	52.33	

Table 14: Best-of- N results on AlpacaEval, MATH, Human Interest, and IFEval comparing FLIP against the lexical-overlap baseline $r(x, y) = F1(x, y)$. Best results per column within each benchmark are shown in **bold**.

requires multiple rounds of comparison over all response pairs, and a valid outcome requires every comparison to parse and the aggregated preferences to yield a unique winner, which explains its lower valid rate. On *Precise IF* the margin between FLIP and the ranking methods is small. However, the ranking methods also drop more instances than FLIP (66.8% vs. 76.6% valid rate on average), so their valid-only accuracy is computed on a smaller and likely easier subset, and they additionally benefit from observing all candidates jointly, whereas FLIP scores each completion independently. Notably, FLIP’s own valid rate (76.6%) is comparable to that of Pointwise Rating (78.6%), so FLIP is not benefiting from being an easier format to parse. Overall, the improvement reflects genuine gains in reward quality rather than differences in format robustness.

Finally, the extrinsic evaluations are immune to this confounder altogether. In the test-time scaling experiments (Section 4.2, Figure 4) and the GRPO experiments (Section 4.3, Table 3), performance is measured directly by downstream benchmark quality, with no validity-based accounting involved. FLIP consistently outperforms the judgment baselines in both settings, which independently con-

firms that its advantage is not driven by the format failure convention.

L Instruction Echoing in FLIP-Trained Policies

Because FLIP rewards the similarity between the original and the inferred instruction, one might worry that optimizing against it incentivizes the policy to restate the instruction inside its response. We test this directly by measuring the rate of instruction echoing on IFBench responses, comparing the base policy (OLMo-3-7B-Think-DPO) against the FLIP-trained policy (GRPO, step 400). We distinguish three categories: reproducing the full prompt verbatim, reproducing part of the prompt verbatim, and paraphrasing the prompt. Results are reported in Table 16.

Echoing and paraphrasing do not increase after training with FLIP, and verbatim reproduction in fact decreases: full verbatim echoing drops from 5.3% to 2.3% and partial verbatim echoing from 29.3% to 20.3%, while the paraphrase rate is essentially unchanged.

We attribute this to the fact that verbatim overlap between the response y and the instruction x carries no first-order payoff under FLIP. What is

	Pointwise Rating	Listwise Ranking	Pairwise Ranking	FLIP
<i>Valid response rate</i>				
OLMo2	62.7	76.8	51.6	62.3
Llama3	86.1	74.6	60.8	71.5
Qwen3	73.7	43.5	54.7	79.7
Gemma3	81.9	82.7	78.6	80.7
Mistral	97.7	99.2	98.7	95.6
Average	78.6	69.1	64.5	76.6
<i>Valid-only accuracy, by subset</i>				
Focus	26.6	40.8	35.5	61.3
Factuality	23.3	29.5	29.7	30.7
Precise IF	19.5	29.5	28.9	29.0
Math	37.0	37.6	37.4	39.8
Average	26.6	34.3	32.9	40.2
<i>Valid-only accuracy, by model family</i>				
OLMo2	33.0	26.3	25.3	38.3
Llama3	17.8	28.5	26.6	34.6
Qwen3	39.8	50.0	47.3	49.8
Gemma3	19.5	25.9	26.7	34.8
Mistral	8.5	30.8	27.8	38.4
Average	26.6	34.3	32.9	40.2

Table 15: Valid response rate (%) and RewardBench2 accuracy computed on valid responses only, averaged over five runs and aggregated within each subset or model family.

Category	Before	FLIP GRPO
Full verbatim prompt	5.3	2.3
Partial verbatim	29.3	20.3
Paraphrase	34.3	35.3

Table 16: Rate (%) of instruction echoing in IFBench responses, before and after GRPO training with FLIP as the reward. Echoing does not increase after training, and verbatim reproduction decreases substantially.

rewarded is whatever enables the inferrer to reconstruct x accurately. From the inferrer’s perspective, a restated prompt embedded in a response is simply additional response text to summarize, and, as discussed in the Limitations section, the inferrer does not merely parrot embedded instruction text: the inferred instruction remains dependent on the substantive content of the response. Moreover, repeating or paraphrasing the instruction often degrades overall response quality, which further discourages the behavior.

M Comparison with Trained Small Reward Models

Our baselines are prompting-based methods that use general-purpose language models, since FLIP

targets the setting where no task-specific reward model is available. For practitioners, however, it is useful to situate these numbers relative to off-the-shelf trained reward models at comparable scale. The strongest publicly available small sequence-classifier RMs are the Skywork-Reward-V2 series (Liu et al., 2024), which are Bradley–Terry classifiers. Table 17 reports their RewardBench2 accuracy on the four subsets used in our evaluation, taken from the leaderboard.

Trained classifier RMs outperform FLIP on RewardBench2. We view the two approaches as addressing different situations, and offer the following notes to practitioners.

- **If a well-matched trained RM exists for the target domain, use it.** Skywork-Reward-V2 is trained on 26 million preference pairs curated through a large-scale human–LLM annotation pipeline (Liu et al., 2024). Where that investment has already been made and the domain matches, a trained classifier is the right tool.
- **If there is no preference data, no annotation budget, or the domains are arbitrary or shifting, use FLIP.** FLIP requires neither preference data nor references or rubrics, and consistently outperforms prompting-based judgment baselines in our experiments.
- **The two are composable rather than competing.** FLIP requires only a generative model and can be applied on top of any base model, including one that has itself undergone reward training. Combining FLIP’s backward-inference signal with a trained RM score is a direction for future work.

N Ensembling FLIP with Judgment Baselines

Since FLIP and LLM-as-a-Judge access different signals, a question is whether combining them yields a stronger reward model than either alone. For each of the 13 SLMs, we construct an ensemble reward by taking the majority vote over Pointwise Rating, Listwise Ranking, Pairwise Ranking, and FLIP, and re-evaluate on RewardBench2 under the same protocol as Table 1 (1,313 instances, four subsets). Results are reported in Table 18.

The ensemble reward underperforms FLIP, often by a wide margin: 20.4 versus 34.5 on average, with the gap largest on *Focus* (32.5 versus 59.6).

Trained RM (BT classifier)	Focus	Factuality	Precise IF	Math
Skywork-Reward-V2-Qwen3-0.6B	79.5	58.0	40.0	71.6
Skywork-Reward-V2-Llama-3.2-1B	89.3	60.8	45.6	60.1
Skywork-Reward-V2-Qwen3-1.7B	88.5	65.7	44.4	72.7

Table 17: RewardBench2 accuracy of off-the-shelf trained small reward models, reported on the leaderboard, as a reference point for the prompting-based methods evaluated in this work. The numbers are not strictly comparable to ours due to a scoring difference: the leaderboard protocol assigns partial credit to instances with format failures, whereas our protocol scores them as zero (Appendix K).

Subset	Method	OLMO2		LLAMA3			QWEN3				GEMMA3			MISTRAL	Average
		1B	7B	1B	3B	8B	0.6B	1.7B	4B	8B	270M	1B	4B	7B	
Focus	FLIP	55.0	65.6	43.3	38.3	64.1	65.7	58.6	66.8	72.2	50.7	59.1	65.5	69.3	59.6
	Ensemble	3.7	33.8	8.4	13.4	35.1	34.1	49.1	59.3	54.2	13.9	30.5	48.6	37.8	32.5
Factuality	FLIP	25.8	27.8	26.1	26.4	27.3	28.4	29.6	31.3	31.1	29.8	24.9	26.5	27.1	27.9
	Ensemble	3.2	18.6	9.5	15.1	21.0	16.5	23.9	23.4	24.1	11.1	16.6	26.5	24.3	18.0
Precise IF	FLIP	23.2	21.5	23.0	24.2	23.1	24.6	23.6	25.0	24.4	21.9	22.2	23.8	23.0	23.3
	Ensemble	3.0	15.0	9.4	12.0	15.2	12.4	15.2	10.8	12.4	9.0	19.6	17.5	26.0	13.7
Math	FLIP	28.4	26.3	26.6	23.8	25.8	26.3	30.2	29.4	31.0	22.4	26.2	27.1	29.7	27.2
	Ensemble	2.5	12.2	8.9	16.6	25.5	10.5	29.0	18.0	15.5	5.1	21.4	32.0	26.9	17.2
Average	FLIP	33.1	35.3	29.8	28.2	35.1	36.3	35.5	38.1	39.7	31.2	33.1	35.7	37.3	34.5
	Ensemble	3.1	19.9	9.1	14.3	24.2	18.4	29.3	27.9	26.6	9.8	22.0	31.2	28.8	20.4

Table 18: RewardBench2 accuracy of FLIP compared with a majority-vote ensemble over Pointwise Rating, Listwise Ranking, Pairwise Ranking, and FLIP. The higher of the two is shown in **bold** for each model and subset. The ensemble underperforms FLIP alone in nearly every setting.

This reinforces our central claim. For small models, the validation–generation gap makes judgment unreliable as a reward signal, so majority voting injects noise rather than complementary information. It also indicates that FLIP’s advantage does not come from capturing a redundant slice of the judgment signal.

Combining FLIP with judgment nonetheless remains an interesting direction, though it likely requires an *adaptive* rather than a uniform ensemble. FLIP could be used to catch off-topic, factually incorrect, and under- or over-addressed responses (Section 5), while LM judges handle dimensions where backward inference is inherently limited, such as safety-critical queries where the desired behavior is refusal. Ensembling may also become viable at larger scales, where judgment is competitive (Appendix G). Learned or confidence-weighted combinations of FLIP and judgment for medium and large models are a promising direction for future work.

O Prompts

We list the prompts used in the main sets of experiments in Tables 19, 20, 21, and 22. Additional prompts used for the prompt consistency check are

provided in Tables 23 through 26.

P Qualitative Examples

We present qualitative examples of FLIP across various RewardBench2 subsets in Table 27 to Table 31. These examples illustrate that our method effectively identifies off-topic, factually incorrect, and instruction-misaligned outputs.

Q Discussion and Justification

Q.1 Does FLIP Only Work for Long Responses?

FLIP relies on the premise that a response contains sufficient length and contextual richness to support backward inference. Importantly, this is precisely the regime in which a reward model is most needed: extremely short responses such as single words or phrases are better handled by simpler metrics such as exact match, and do not require learned reward signals. For the practically relevant setting of open-ended, multi-sentence responses, FLIP is well suited by design. Empirically, while FLIP yields the largest gains over LLM-as-a-Judge on long responses, it consistently outperforms all baselines across short and medium responses as

well (Figure 5). This suggests that FLIP’s advantage is not contingent on response length, but is rather amplified by it.

Q.2 Does FLIP Only Assess Instruction Following?

A related concern is whether FLIP, by relying on the alignment between the original and inferred instructions, evaluates only instruction-following behavior while neglecting other dimensions of response quality such as factual accuracy, completeness, and depth. We argue that while FLIP is strongest at capturing instruction alignment, it implicitly accounts for these other dimensions as well.

Empirically, the breadth of FLIP’s effectiveness is evident across Table 1, Figure 4 and Table 3. Under standard benchmarks, test-time scaling, and GRPO training, FLIP consistently outperforms LLM-as-a-Judge not only on instruction-following benchmarks, but also on mathematics (the Math subset of RewardBench2, Minerva Math, and MATH), factuality (the Factuality subset of RewardBench2), reasoning (BBH and GPQA), and open-domain question answering benchmarks (AlpacaEval and Human Interest). This breadth of improvement, spanning domains where instruction alignment is not the primary quality bottleneck, suggests that FLIP captures a richer quality signal than instruction following alone.

Conceptually, as illustrated by the qualitative examples in Section 5, FLIP is sensitive to factual accuracy, completeness, and depth through the backward inference mechanism. Factual errors, for instance, often cause the inferred instruction to diverge substantially from the original: a response that identifies "pandas" as the symbolic animal of the United States may lead the model to infer an instruction about China’s symbolic animal instead, yielding a low F1 score. Similarly, an incomplete response tends to reconstruct only a subset of the original instruction’s requirements, while a shallow response may reconstruct a simpler or underspecified version. Additional examples are provided in Appendix P.

That said, we acknowledge that instruction recoverability is a proxy for response quality rather than a complete measure of it, in the same way that BLEU measures n-gram overlap as a proxy for translation quality. No proxy is perfect, and both have known failure modes. The value of such proxies lies not in their theoretical completeness, but in their empirical reliability and practical accessibil-

ity, particularly in regimes where stronger signals, such as human judgment, verified answers, or large model judges, are unavailable or prohibitively expensive. Our results demonstrate that instruction recoverability is a surprisingly effective proxy across diverse domains, even in cases where its theoretical justification is less direct.

Q.3 Does FLIP Introduce a Verbosity Bias?

A concern is whether FLIP systematically favors longer responses, since richer context may facilitate more accurate instruction reconstruction. We argue that this is not the case. FLIP rewards the degree to which a response uniquely and accurately implies its generating instruction. While response length can aid reconstruction, it only does so when the additional content is on-topic. A verbose but off-topic response will reconstruct a different instruction, penalizing the F1 score. Conversely, a concise but precise response can reconstruct the original instruction well. Verbosity therefore helps FLIP only when it reflects genuine quality, which is the desired behavior of a reward model.

Empirically, we find no evidence of verbosity bias in our GRPO experiments. Despite FLIP yielding consistent downstream gains across all benchmarks (Table 3), we do not observe a corresponding inflation in response length. Since all evaluation benchmarks have verifiable answers, longer responses do not mechanically improve accuracy, so the observed gains must reflect genuine improvements in response quality rather than a length-gaming strategy.

Q.4 What if the Response and Instruction Are in Different Languages?

In rare cases, a response may be in a different language than the original instruction, for instance, when a model responds in its default language regardless of the instruction’s language. This can cause the inferred instruction to be expressed in a different language than the original, rendering word-level F1 an inappropriate similarity metric. In such scenarios, we recommend replacing F1 with an LLM-based similarity judge, which operates on semantic rather than lexical overlap and is therefore robust to cross-lingual mismatches (Appendix D). Notably, this issue is limited to cases of genuine language mismatch and does not affect the vast majority of evaluation settings considered in this work.

Q.5 Related Work

Yerram et al. (2024) introduce Time Reversed Language Models (TRLMs), which score and generate queries conditioned on responses, operating in the reverse direction of time. Specifically, they pre-train a PaLM2 model from scratch in reverse token order and demonstrate that it can complement forward model predictions when used to score the query given a response for re-ranking multiple forward generations. One variant, *TRLM-Fo*, uses the log-probability of the original query given the response as the reranking signal and is most relevant to FLIP. One of the following work LEDOM (Yin et al., 2025) pursues a similar idea, introducing *Reverse Reward*, which uses reverse token-level likelihoods from a fully reverse-trained model, combined with the forward LM score, as the reward signal.

However, FLIP differs from both in several key respects. Motivationally, our work explores reward modeling for small language models in low-capacity regimes, whereas these two works are primarily concerned with reverse training itself. Methodologically, FLIP is training-free: it uses general-purpose language models off the shelf, while *Reverse Reward* relies on expensive reverse pre-training from scratch. Furthermore, FLIP uses word-level F1 between the original instruction and the inferred instruction as the reward signal, which operates in a fully black-box manner, whereas *TRLM-Fo* and *Reverse Reward* rely on log-probability signals that require access to model internals. Empirically, *TRLM-Fo* is evaluated on open-ended QA reranking, citation and retrieval tasks, and safety filtering, while we compare against well-established LLM-as-a-Judge baselines across reward benchmarks and test-time scaling settings, and further investigate whether FLIP can serve as a training signal for reinforcement learning. Therefore, even if the core backward-inference idea overlaps, the problem settings, model scales, methodologies, and practical motivations are substantially different.

Prompt 1: Pointwise Rating
System Prompt ### Task Description Please act as an impartial judge and evaluate the quality of the answer provided by an AI assistant to the user query displayed below. Judge whether the provided answer is good. Notes: - Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and appropriate level of detail of the response. - Begin your evaluation by providing a short explanation. - Be as objective as possible. After providing your short explanation, please output a score on a scale of 1 to 10. - Please respond in the following JSON format: <code>{ "REASONING": "[...]", "SCORE": "<your-score>" }</code>
User Prompt [Query] {prompt} [AI Answer] {response} [Your judgement] Respond in JSON format. <code>{ "REASONING": "[...]", "SCORE": "<your-score>" }</code>

Table 19: Pointwise Rating prompt used for all experiments, unless otherwise stated.

Prompt 1: Listwise Ranking

System Prompt

Task Description

Please act as an impartial judge and evaluate the quality of the answers provided by AI assistants to the user query displayed below.

Notes:

- Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and appropriate level of detail of the response.
- Begin your evaluation by briefly explaining your reasoning for selecting the best response.
- Be as objective as possible. After providing your short explanation, output the index (starting from 0) of the best response.
- Please respond in the following JSON format: `{ "REASONING": "[...your explanation of why the chosen response is best...]", "BEST_RESPONSE_INDEX": <index-of-best-response> }`

User Prompt

[Query]

{prompt}

[AI Answers]

{response_list}

[Your judgement]

Respond in JSON format. `{ "REASONING": "[...your explanation of why the chosen response is best...]", "BEST_RESPONSE_INDEX": <index-of-best-response> }`

Table 20: Listwise Ranking prompt used for all experiments, unless otherwise stated.

Prompt 1: Pairwise Ranking

System Prompt

Task Description

Please act as an impartial judge and evaluate the quality of the answers provided by AI assistants to the user query displayed below.

Notes:

- Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and appropriate level of detail of the response.
- Begin your evaluation by briefly explaining your reasoning for selecting the best response.
- Be as objective as possible. After providing your short explanation, output either "A" or "B" depending on which response is better.
- Please respond in the following JSON format: `{{ "REASONING": "[...your explanation of why the chosen response is best...]", "BETTER_RESPONSE": "A" or "B" }}`

User Prompt

[Query]

{prompt}

[AI Response A]

{response1}

[AI Response B]

{response2}

[Your judgement]

Respond in JSON format. `{{ "REASONING": "[...your explanation of why the chosen response is best...]", "BETTER_RESPONSE": "A" or "B" }}`

Table 21: Pairwise Ranking prompt used for all experiments, unless otherwise stated.

Prompt 1: FLIP

System Prompt

Task Description

Please act as an instruction reconstructor. You will receive a model-generated response. Read it carefully and infer the instruction or prompt that produced it.

Notes:

- Reconstruct a single instruction that would most plausibly generate the given response.
- Specify the user's intent, required output format, and any constraints.
- Be concise and concrete.
- Do not add information or assumptions beyond what appears in the response.
- Do not overfit to the specific phrasing or examples used in this response.
- Begin with a brief explanation of your reasoning.
- Please respond in the following JSON format: `{ "REASONING": "<explanation>", "INFERRED INSTRUCTION": "<instruction>" }`

User Prompt

[Response]

{response}

[Your Inferred Instruction]

Respond in JSON format. `{ "REASONING": "<explanation>", "INFERRED INSTRUCTION": "<instruction>" }`

Table 22: FLIP prompt used for all experiments, unless otherwise stated.

Prompt 2: Pointwise Rating

System Prompt

Task Description

Please act as an impartial evaluator and assess the quality of the AI assistant's response to the user query shown below. Determine whether the response is effective.

Notes:

- Your evaluation should take into account factors such as helpfulness, relevance, accuracy, depth, creativity, and whether the level of detail is appropriate.
- Start your evaluation with a brief explanation.
- Remain as objective as possible. After the explanation, assign a score on a scale from 1 to 10.
- Please respond in the following JSON format: `{"REASONING": "[...]", "SCORE": "<your-score>"}`

Prompt 3: Pointwise Rating

System Prompt

Task Description

Please serve as an unbiased reviewer and evaluate the quality of the AI assistant's answer to the user query provided below. Decide whether the response meets an acceptable standard.

Notes:

- Consider criteria such as usefulness, relevance, correctness, thoroughness, creativity, and the appropriateness of the level of detail.
- Begin with a brief justification of your assessment.
- Maintain objectivity throughout the evaluation. After the justification, assign a numerical score from 1 to 10.
- Please respond in the following JSON format: `{"REASONING": "[...]", "SCORE": "<your-score>"}`

Table 23: Pointwise Rating prompt variations used in robustness experiments. The user prompt remains unchanged.

Prompt 2: Listwise Ranking

System Prompt

Task Description

You are asked to serve as a neutral evaluator and judge the quality of multiple AI-generated responses to the user prompt provided below.

Notes:

- Base your judgment on criteria such as usefulness, relevance to the question, factual correctness, thoroughness, originality, and suitability of detail.
- Begin with a concise justification explaining why one response stands out as superior.
- Maintain objectivity throughout the assessment. After your justification, indicate which response is the best by providing its zero-based index.
- Please respond in the following JSON format: `{ "REASONING": "[...your explanation of why the chosen response is best...]", "BEST_RESPONSE_INDEX": <index-of-best-response> }`

Prompt 3: Listwise Ranking

System Prompt

Task Description

Assume the role of an independent reviewer responsible for comparing several AI assistant replies to the same user question.

Instructions:

- Evaluate each reply using criteria such as practical value, alignment with the prompt, correctness, level of insight, originality, and completeness.
- Open with a short rationale explaining why one reply outperforms the others.
- Keep your judgment fair and unbiased. After the rationale, specify the winning reply by providing its position number, starting from 0.
- Please respond in the following JSON format: `{ "REASONING": "[...your explanation of why the chosen response is best...]", "BEST_RESPONSE_INDEX": <index-of-best-response> }`

Table 24: Listwise Ranking prompt variations used in robustness experiments. The user prompt remains unchanged.

Prompt 2: Pairwise Ranking

System Prompt

Task Description

Assume the role of an unbiased reviewer and determine which of the two AI-generated answers best addresses the user's question.

Criteria to consider:

- Usefulness and relevance to the question
- Factual accuracy
- Completeness and depth of explanation
- Originality or insight where appropriate
- Suitability of the response length and detail

Process:

- Begin with a concise justification explaining why one response outperforms the other.
- Remain neutral and evidence-based in your judgment.
- Conclude by selecting either "A" or "B" as the superior answer.
- Please respond in the following JSON format: `{{ "REASONING": "[...your explanation of why the chosen response is best...]", "BETTER_RESPONSE": "A" or "B" }}`

Prompt 3: Pairwise Ranking

System Prompt

Task Description

You are asked to objectively compare two AI assistant replies to a given user prompt and decide which one is superior.

Assessment considerations:

- How well each response addresses the user's intent
- Accuracy and reliability of the information
- Clarity, structure, and depth of the explanation
- Added value, insight, or creativity when relevant
- Whether the amount of detail is appropriate

Instructions:

- First, provide a short rationale explaining why the better response stands out.
- Keep the evaluation fair, neutral, and concise.
- Then clearly indicate your choice by selecting "A" or "B".
- Please respond in the following JSON format: `{{ "REASONING": "[...your explanation of why the chosen response is best...]", "BETTER_RESPONSE": "A" or "B" }}`

Table 25: Pairwise Ranking prompt variations used in robustness experiments. The user prompt remains unchanged.

Prompt 2: FLIP

System Prompt

Task Description

Act as an instruction reverse-engineer. You will be given a response generated by a model. Carefully analyze the response and infer the most likely instruction or prompt that led to it.

Notes:

- Reconstruct a single, plausible instruction.
- Clearly capture the user's intent, expected output format, and constraints.
- Keep the instruction concise and specific.
- Do not introduce assumptions or details not supported by the response.
- Start with a short explanation of how you derived the instruction.
- Please respond in the following JSON format: `{{ "REASONING": "<explanation>", "INFERRED INSTRUCTION": "<instruction>" }}`

Prompt 3: FLIP

System Prompt

Task Description

Assume the role of an instruction reverse engineer. You will receive a model-generated response and must analyze it to deduce the most likely prompt that produced it.

Notes:

- Infer one coherent and plausible instruction.
- Accurately reflect the user's intent, required output structure, and any constraints.
- Be concise and precise.
- Avoid adding assumptions or information not evidenced by the response.
- Do not tailor the instruction too closely to the exact examples in the response.
- Begin with a brief explanation describing how the instruction was inferred.
- Please respond in the following JSON format: `{{ "REASONING": "<explanation>", "INFERRED INSTRUCTION": "<instruction>" }}`

Table 26: FLIP prompt variations used in robustness experiments. The user prompt remains unchanged.

Instruction	Reasons for and against the use of the atomic bomb in 1945.
Subset	Focus
Chosen Response	The decision to use atomic bombs on Hiroshima and Nagasaki in August 1945 was a complex one, with both compelling reasons for and against its use. Here are some of the key arguments from each side: ### Reasons for Using the Atomic Bomb 1. End the War Quickly: - The primary reason cited by U.S. leaders was to end the war quickly and save American lives. They believed that a full-scale invasion of Japan would result in significant casualties on both sides. 2. Deterrence: - The United States wanted to demonstrate the power of its new weapon to the Soviet Union and other potential adversaries. This could serve as a deterrent for future conflicts. 3. Psychological Impact: - The hope was that the devastating effects of the atomic bombs would shock the Japanese government into surrendering without the need for further destruction or invasion. 4. Strategic Advantage: - The U.S. military believed that the atomic bomb would give them a strategic advantage over the Soviet Union, which was also involved in the Pacific theater. 5. Prevent Further Suffering: - Some argued that using the bomb would prevent even greater suffering and destruction if the war continued, including the potential for a Japanese suicide attack on American cities. ### Reasons Against Using the Atomic Bomb 1. Casualties and Moral Concerns: - There were significant concerns about the massive civilian casualties that would result from the bombings. The estimated death toll from the atomic bombs was in the tens of thousands, and many more would suffer long-term health effects. 2. International Repercussions: - The use of atomic weapons raised serious moral and ethical questions. It set a precedent for the use of such devastating weapons and could have long-lasting negative impacts on international relations. 3. Alternative Methods: - Some argued that there were alternative methods to force Japan's surrender, such as a demonstration of the bomb's power in an uninhabited area or a formal warning to the Japanese government. 4. Impact on the Soviet Union: - There were concerns that the use of the bomb might provoke the Soviet Union to accelerate its own nuclear program, leading to an arms race. 5. Long-Term Consequences: - The long-term environmental and health consequences of radioactive fallout were not fully understood at the time but were later recognized as significant issues. 6. Japanese Surrender Without Bombing: - Historians debate whether Japan would have surrendered without the atomic bombs. Some argue that the Potsdam Declaration and the Soviet declaration of war on August 8th were sufficient to prompt Japan's surrender. ### Conclusion The decision to use the atomic bomb remains one of the most controversial in modern history. While it is clear that the U.S. leadership believed it was necessary to end the war quickly and save lives, the moral and ethical implications continue to be debated. The use of these weapons marked a turning point in human history and had profound and lasting effects on global politics and warfare.
Inferred Instruction	Write a balanced analysis of the decision to use atomic bombs on Hiroshima and Nagasaki, highlighting both the reasons for and against the use of these weapons, and present the argument in a neutral tone.
F1(Instruction, Inferred Instruction)	0.341

Table 27: Example of FLIP on the RewardBench2 *Focus* subset. See Table 28 for the rejected response. The chosen response closely follows the instruction and discusses the reasons for and against the use of the atomic bomb, whereas the rejected response deviates by focusing on public opinion before and after the atomic bomb. FLIP successfully captures this deviation, yielding higher similarity between the inferred instruction based on the chosen response and the original instruction than that based on the rejected response.

Instruction	Reasons for and against the use of the atomic bomb in 1945.
Subset	Focus
Rejected Response	Public opinion in the United States underwent significant changes before and after the atomic bombings of Hiroshima and Nagasaki in August 1945. Here's a brief overview of the shift in public opinion: Before the bombings (1945): Support for the use of atomic bombs: In the months leading up to the bombings, there was a growing sense of urgency and frustration among Americans regarding the war in the Pacific. Many people believed that the use of atomic bombs would bring a swift end to the war and save American lives. A Gallup poll conducted in June 1945 found that 85% of Americans supported the use of atomic bombs if it would shorten the war. Fear of the Japanese: The Japanese government had been refusing to surrender, and there was a growing fear among Americans that the Japanese would continue to fight to the last man, leading to a prolonged and bloody conflict. This fear contributed to the widespread support for the use of atomic bombs. Scientific and military enthusiasm: The scientific community and the military were excited about the potential of the atomic bomb, and many saw it as a game-changer in the war. The Manhattan Project, the secret research and development project that produced the atomic bomb, was shrouded in secrecy, and many Americans were unaware of the devastating consequences of the bomb. After the bombings (August 1945): Shock and horror: The immediate reaction to the bombings was one of shock and horror. The news of the devastating effects of the atomic bombs on Hiroshima and Nagasaki was met with widespread outrage and dismay. Many Americans were unaware of the full extent of the destruction and the number of civilian casualties. Changing attitudes towards the use of atomic bombs: As the full extent of the damage became clear, public opinion began to shift. Many Americans began to question the morality and necessity of using such a devastating weapon. A Gallup poll conducted in September 1945 found that 37% of Americans believed that the use of atomic bombs was a mistake, while 44% still supported it. Growing concern about the future of nuclear warfare: The bombings raised concerns about the potential for future nuclear conflicts and the devastating consequences of such a war. This led to a growing debate about the ethics and consequences of nuclear warfare, which would continue to shape American public opinion in the years to come. Rise of the anti-nuclear movement: The bombings marked the beginning of the anti-nuclear movement in the United States. Many Americans began to question the use of nuclear energy and the development of nuclear weapons, leading to a growing movement to ban nuclear testing and reduce the threat of nuclear war. In summary, public opinion in the United States shifted from a general support for the use of atomic bombs to a growing concern about the devastating consequences of such a weapon. The bombings marked a turning point in American public opinion, leading to a greater awareness of the risks and consequences of nuclear warfare.
Inferred Instruction	Write a comprehensive overview of the shift in public opinion in the United States before and after the atomic bombings of Hiroshima and Nagasaki in 1945, including the reasons for the shift and the impact on American public opinion.
F1(Instruction, Inferred Instruction)	0.318

Table 28: Example of FLIP on the RewardBench2 *Focus* subset (continued).

Instruction	Write a list of 20 animated films from the 1990s, in the following format: “[TITLE] ([YEAR]), by [DIRECTORS]”. Do not write any descriptions, or sort in any order.
Subset	Factuality
Chosen Response	1. The Lion King (1994), by Roger Allers and Rob Minkoff 2. Beauty and the Beast (1991), by Gary Trousdale and Kirk Wise 3. Aladdin (1992), by Ron Clements and John Musker 4. Toy Story (1995), by John Lasseter 5. Pocahontas (1995), by Mike Gabriel and Eric Goldberg 6. The Nightmare Before Christmas (1993), by Henry Selick 7. Mulan (1998), by Tony Bancroft and Barry Cook 8. A Bug’s Life (1998), by John Lasseter and Andrew Stanton 9. Hercules (1997), by Ron Clements and John Musker 10. The Hunchback of Notre Dame (1996), by Gary Trousdale and Kirk Wise 11. Anastasia (1997), by Don Bluth and Gary Goldman 12. The Prince of Egypt (1998), by Brenda Chapman, Steve Hickner, and Simon Wells 13. Antz (1998), by Eric Darnell and Tim Johnson 14. Tarzan (1999), by Chris Buck and Kevin Lima 15. The Iron Giant (1999), by Brad Bird 16. James and the Giant Peach (1996), by Henry Selick 17. The Secret of NIMH 2: Timmy to the Rescue (1998), by Dick Sebast 18. Quest for Camelot (1998), by Frederik Du Chau 19. Thumbelina (1994), by Don Bluth and Gary Goldman 20. The Swan Princess (1994), by Richard Rich
Inferred Instruction	List 20 popular animated films from the 1990s that were produced by Disney, Pixar, or other animation studios.
F1(Instruction, Inferred Instruction)	0.381
Rejected Response	"Beauty and the Beast (1991), by Gary Trousdale and Kirk Wise "The Lion King (1994), by Roger Allers and Rob Minkoff "Aladdin (1992), by Ron Clements and John Musker "Toy Story (1995), by John Lasseter "Matilda (1996), by Danny DeVito "Anastasia (1997), by Don Bluth and Gary Goldman "Hercules (1997), by Ron Clements and John Musker "Mulan (1998), by Tony Bancroft and Barry Cook "Tarzan (1999), by Chris Buck and Kevin Lima "Antz (1998), by Eric Darnell and Tim Johnson "Monsters, Inc. (2001), by Pete Docter "The Prince of Egypt (1998), by Brenda Chapman, Steve Hickner, and Simon Wells "Flubber (1997), by Les Mayfield "Kiki’s Delivery Service (Majo no Takkyūbin) (1998), by Hayao Miyazaki "A Bug’s Life (1998), by John Lasseter and Andrew Stanton "Stuart Little (1999), by Rob Minkoff "Treasure Planet (2002), by Ron Clements and John Musker "Titan A.E. (2000), by Don Bluth and Gary Goldman "James and the Giant Peach (1996), by Henry Selick "The Hunchback of Notre Dame (1996), by Gary Trousdale and Kirk Wise"
Inferred Instruction	List the 20 most popular animated films released between 1991 and 2005, with a focus on American and American-influenced films.
F1(Instruction, Inferred Instruction)	0.186

Table 29: Example of FLIP on the RewardBench2 *Factuality* subset. The chosen response correctly lists 20 films from the 1990s, whereas the rejected response incorrectly includes several films from the 2000s. FLIP successfully captures this difference, yielding higher similarity between the inferred instruction based on the chosen response and the original instruction than that based on the rejected response.

Instruction	What's the root meaning of politics Please use an emoji at the end of every sentence.
Subset	Precise IF
Chosen Response	The root meaning of politics comes from the Greek word "politikos," which means "of, for, or relating to citizens." [EMOJI] It originally referred to the art or science of government and the activities associated with governing a city-state, like Athens. [EMOJI] Over time, the term has evolved to encompass all aspects of government, including the strategies used to gain or maintain power and the decisions that influence the welfare of a community or nation. [EMOJI]
Inferred Instruction	Explain the meaning and development of the term 'politics' over time, including an emoji representation of politics.
F1(Instruction, Inferred Instruction)	0.370
Rejected Response	The root meaning of "politics" comes from the Greek word "politikos," which means "of, for, or relating to citizens." This word is derived from "polis," which means city or state. The study of politics involves the distribution of power, governance, public policy, and the methods and tactics used to formulate policies. [EMOJI]
Inferred Instruction	Define politics, including its origins and main topics of study.
F1(Instruction, Inferred Instruction)	0.174

Table 30: Example of FLIP on the RewardBench2 *Precise IF* subset. The chosen response correctly adds an emoji at the end of every sentence, whereas the rejected response does not. FLIP successfully captures this difference, yielding higher similarity between the inferred instruction based on the chosen response and the original instruction than that based on the rejected response.

Instruction	It would be necessary to convert the mass of $_ \times 10^9$ protons into energy in order to release 63.1 J of energy.
Subset	Math
Chosen Response	Let 1) We'll use Einstein's mass-energy equivalence equation: $E = mc^2$ where E is energy in Joules (J) m is mass in kg c is speed of light = 3×10^8 m/s 2) We know: $E = 63.1$ J We need to find the number of protons 3) Rearranging $E = mc^2$: $m = \frac{E}{c^2}$ $m = \frac{63.1}{(3 \times 10^8)^2}$ $m = \frac{63.1}{9 \times 10^{16}}$ $m = 7.01 \times 10^{-16}$ kg 4) To convert from kg to protons: Mass of one proton = 1.67×10^{-27} kg Number of protons = $\frac{\text{total mass}}{\text{mass of one proton}}$ $= \frac{7.01 \times 10^{-16}}{1.67 \times 10^{-27}}$ $= 4.20 \times 10^{11}$ 5) Since the answer needs to be in terms of $x \times 10^9$: $4.20 \times 10^{11} = 420 \times 10^9$ Therefore, 420×10^9 protons would need to be converted to energy to release 63.1 J of energy. The answer is 420.
Inferred Instruction	Calculate the number of protons (in multiples of 10^9) required to release 63.1 Joules of energy using the mass-energy equivalence equation and the mass of a proton.
F1(Instruction, Inferred Instruction)	0.419
Rejected Response	To solve this problem, we need to use the Einstein's mass-energy equivalence formula: $E = mc^2$, where E is the energy, m is the mass, and c is the speed of light. We are given the energy $E = 63.1$ J. We need to find the mass m that would release this energy when converted into energy using Einstein's formula. First, we need to convert the speed of light c into meters per second. The speed of light is approximately $c = 3 \times 10^8$ m/s. Now, we can rearrange the formula $E = mc^2$ to solve for mass m: $m = \frac{E}{c^2}$. Plugging the given energy $E = 63.1$ J and the speed of light $c = 3 \times 10^8$ m/s into the formula, we get: $m = \frac{63.1 \text{ J}}{9 \times 10^{16} \text{ m}^2/\text{s}^2} = 7.011 \times 10^{-16}$ kg. Now, we know that 1 proton has a mass of approximately 1.67×10^{-27} kg. To find out how many protons have a mass of 7.011×10^{-16} kg, we can divide the total mass by the mass of one proton: Number of protons = $\frac{7.011 \times 10^{-16} \text{ kg}}{1.67 \times 10^{-27} \text{ kg/proton}} = 4.20 \times 10^{11}$ protons. So, it would be necessary to convert the mass of 4.20×10^{11} protons into energy in order to release 63.1 J of energy. The answer is: 4.20×10^{11}.
Inferred Instruction	Given an energy value (63.1 J), find out how many protons have a combined mass that, when converted to energy, would result in the given energy using Einstein's mass-energy equivalence formula.
F1(Instruction, Inferred Instruction)	0.375

Table 31: Example of FLIP on the RewardBench2 *Math* subset. The chosen response correctly converts the final answer into multiples of 10^9 , whereas the rejected response does not. FLIP successfully captures this difference, yielding higher similarity between the inferred instruction based on the chosen response and the original instruction than that based on the rejected response.