Constraint Decomposition for Multi-Objective Instruction-Following in Large Language Models

Eva Paunova
NVIDIA Research
e.hpaunova@gmail.com

December 2025

Abstract

Large language models (LLMs) trained with reinforcement learning from human feedback (RLHF) struggle with complex instructions that bundle multiple, potentially conflicting requirements. We introduce *constraint decomposition*, a framework that separates multi-objective instructions into orthogonal components—semantic correctness, structural organization, format specifications, and meta-level requirements—and optimizes each independently before hierarchical combination. Our approach addresses the fundamental limitation of monolithic reward models: their inability to distinguish which specific constraint failed when multiple requirements conflict. We train decomposed reward models with aspect-level human preferences and demonstrate that explicit constraint separation, combined with conflict-aware weight adaptation, enables more effective multi-objective optimization. On the IFEval benchmark, our method achieves 73.8% prompt-level accuracy (±1.6%), a 32.6 percentage point improvement over standard RLHF (41.2%). Ablation studies show that constraint decomposition contributes 54% of the total improvement, with hierarchical combination adding 17%, weight prediction 15%, and conflict detection 14%. Our method generalizes to GSM8K (+15.6 points), HumanEval (+11.1 points), and MT-Bench (+1.4 points). Code and data are available at https://github.com/epaunova/constraint-decomposition-llm.

1 Introduction

The alignment of large language models (LLMs) with human preferences through reinforcement learning from human feedback (RLHF) has become the dominant paradigm for instruction-following systems [17, 1]. However, real-world instructions often bundle multiple requirements that must be satisfied simultaneously. Consider the instruction: "Solve the equation $x^2+5x+6 = 0$ using the quadratic formula, show all steps clearly, keep your answer under 100 words, and explain like I'm in high school." This single instruction conflates four distinct constraints:

1. **Semantic correctness**: Mathematical accuracy of the solution
2. **Structural organization**: Step-by-step reasoning clarity
3. **Format specification**: Length constraint (<100 words)
4. **Meta-level requirement**: Audience-appropriate language (high school level)

Standard RLHF approaches learn a single scalar reward function $R(x, y)$ that must simultaneously capture all these dimensions [6]. This monolithic formulation creates two critical problems:

Problem 1: Constraint Interference. When requirements conflict—such as providing comprehensive explanations while respecting strict length limits—a scalar reward cannot distinguish which constraint to prioritize. The model receives a low reward signal but no information about which constraint failed, leading to inefficient learning and suboptimal compromises.

Problem 2: Credit Assignment Ambiguity. When a response fails, the monolithic reward provides no mechanism to attribute failure to specific constraint violations. A response that is mathematically correct but exceeds the word limit receives the same undifferentiated penalty as one that is incorrect but well-formatted.

These limitations manifest empirically: on the IFEval benchmark [27], which explicitly tests multi-constraint instruction-following, standard RLHF achieves only 41.2% prompt-level accuracy. Analysis of failures reveals that 78% involve constraint interference—satisfying one requirement while violating another.

1.1 Our Contributions

We propose *constraint decomposition*, a framework that addresses these limitations through three key innovations:

1. **Decomposed Reward Architecture** (§3.2). We factorize the reward function into four orthogonal components: R_{sem}, R_{struct}, R_{fmt}, and R_{meta}, each learned from aspect-specific human preferences. All components share a Nemotron-7B encoder with lightweight per-aspect heads (16K parameters total), enabling efficient training while providing distinct gradient signals for each constraint.

2. **Hierarchical Combination Function** (§3.3). Rather than linearly combining component rewards, we introduce a structured composition that respects constraint dependencies: safety gates block unsafe responses absolutely, semantic and structural quality form the base score, and format acts as a multiplicative modulator.

3. **Conflict-Aware Weight Adaptation** (§3.4). We train auxiliary models to detect constraint conflicts and dynamically adjust component weights. When conflicts are detected (e.g., “be concise but comprehensive”), semantic correctness receives higher priority while format constraints are relaxed.

Empirical Results. On IFEval, our method achieves 73.8% accuracy (+32.6 points over baseline RLHF), with statistically significant improvements across five independent runs ($p < 0.0001$, Cohen’s $d = 8.97$). Ablation studies demonstrate that decomposition contributes 54% of the improvement, hierarchical combination adds 17%, weight prediction provides 15%, and conflict adaptation contributes 14%. Critically, improvements generalize beyond instruction-following: GSM8K math reasoning (+15.6 points), HumanEval code generation (+11.1 points), and MT-Bench conversational quality (+1.4 points) all show substantial gains.

2 Related Work

Multi-Objective Reinforcement Learning. The challenge of optimizing multiple competing objectives has been extensively studied in RL [25, 10]. Prior work has explored Pareto-optimal solutions [24], scalarization with learned weights [19], and multi-policy approaches [2]. However, these methods have primarily been applied to low-dimensional discrete action spaces rather than the high-dimensional spaces of language generation. We adapt multi-objective RL principles to the unique challenges of LLM alignment.

Reward Modeling for LLMs. Bradley-Terry preference models [3] have become the standard for learning reward functions from human feedback [6, 22]. Recent work has explored reward model ensembles for uncertainty estimation [8], process-based rewards for reasoning [23], and implicit reward learning via DPO [18]. Our work is orthogonal: we decompose the reward structure itself rather than improving the learning algorithm.

Constraint Satisfaction in LLMs. Prior work has addressed specific constraint types: length control via length tokens [13], format constraints through constrained decoding [12], and style transfer with disentangled representations [21]. NeuroLogic [15] enforces logical constraints via beam search. However, these approaches handle constraints at inference time, while we learn constraint satisfaction through training-time reward decomposition.

3 Method

3.1 Problem Formulation

We formalize instruction-following as a multi-objective reinforcement learning problem. Let x denote an instruction (prompt) and y denote a response (completion). Standard RLHF optimizes:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{x,y\sim\pi(\cdot|x)} \left[R(x,y) - \beta \cdot \mathrm{KL}(\pi \| \pi_{\mathrm{ref}})\right] \tag{1}$$

where $R : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is a scalar reward function learned from human preferences, π_{ref} is a reference policy, and β controls the KL penalty.

We identify that instructions x can be decomposed into constraint tuples:

$$x = (c_{\mathrm{sem}}, c_{\mathrm{struct}}, c_{\mathrm{fmt}}, c_{\mathrm{meta}}) \tag{2}$$

where c_{sem} captures semantic constraints (factual accuracy, logical validity), c_{struct} captures structural constraints (organization, reasoning steps), c_{fmt} captures format constraints (length, style, presentation), and c_{meta} captures meta constraints (safety, tone, audience).

3.2 Decomposed Reward Architecture

We propose decomposing $R(x, y)$ into orthogonal components:

$$R(x,y) = f(R_{\mathrm{sem}}(x,y), R_{\mathrm{struct}}(x,y), R_{\mathrm{fmt}}(x,y), R_{\mathrm{meta}}(x,y)) \tag{3}$$

where each $R_i : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is a specialized reward model trained on aspect-specific preferences, and f is a combination function.

Architecture. All component reward models share a common encoder E_θ to amortize computation:

$$h = E_\theta([x; y]) \tag{4}$$

$$R_i(x,y) = W_i^\top h_{[-1]} + b_i \tag{5}$$

where E_θ is a 7B parameter transformer (Nemotron architecture [16]), $[x; y]$ denotes concatenation, $h_{[-1]}$ is the final token representation, and $W_i \in \mathbb{R}^d$, $b_i \in \mathbb{R}$ are lightweight per-aspect heads. This design provides:

- **Parameter efficiency**: $4 \times 4097 = 16,388$ parameters for all heads vs. 28B for 4 separate models
- **Shared representations**: E_θ learns features useful across all constraints
- **Independent scoring**: Each head produces uncorrelated signals (validated in §5.3)

Training Objective. We train on aspect-level preference data. Each example consists of a prompt x and two responses y_A, y_B with aspect-specific labels:

$$\mathcal{D} = \{(x, y_A, y_B, \ell_{\text{sem}}, \ell_{\text{struct}}, \ell_{\text{fmt}}, \ell_{\text{meta}})\} \tag{6}$$

where $\ell_i \in \{A, B, \text{tie}\}$ indicates which response is preferred for constraint type i.

For each aspect i, we use the Bradley-Terry model:

$$P(y_A \succ_i y_B | x) = \sigma(R_i(x, y_A) - R_i(x, y_B)) \tag{7}$$

where σ is the sigmoid function. The training loss combines all aspects:

$$\mathcal{L} = \sum_i w_i \cdot \mathbb{E}_{(x, y_A, y_B, \ell) \sim \mathcal{D}} \left[-\ell_i \cdot \log P(y_A \succ_i y_B | x) - (1 - \ell_i) \cdot \log P(y_B \succ_i y_A | x)\right] \tag{8}$$

where $\ell_i = 1$ if $\ell_i = A$, $\ell_i = 0$ if $\ell_i = B$, $\ell_i = 0.5$ if tie, and we use uniform $w_i = 0.25$.

Data Collection. We sampled 180,000 prompt-response pairs from publicly available datasets (ShareGPT [5], UltraFeedback [9], and OpenOrca [14]). Three trained annotators labeled each pair with aspect-level preferences. Inter-rater agreement (Fleiss' κ) was 0.73 (substantial agreement).

3.3 Hierarchical Combination Function

Linear combination of component rewards ($f(R_1, \ldots, R_4) = \sum w_i R_i$) achieved only 68.2% on IFEval, suggesting that constraint relationships require explicit structure. We introduce a hierarchical combination function:

$$f(R_{\text{sem}}, R_{\text{struct}}, R_{\text{fmt}}, R_{\text{meta}}) = \begin{cases} -5.0 & \text{if } R_{\text{meta}} < \tau_{\text{safety}} \\ (w_{\text{sem}} \cdot R_{\text{sem}} + w_{\text{struct}} \cdot R_{\text{struct}}) \times (\alpha + (1 - \alpha) \cdot w_{\text{fmt}} \cdot R_{\text{fmt}}) & \text{otherwise} \end{cases} \tag{9}$$

where $\tau_{\text{safety}} = 0.7$, $\alpha = 0.8$, and weights $\{w_i\}$ are either fixed or predicted by a meta-model.

This hierarchy reflects three principles:

1. **Safety Gates**: Meta-level constraints (safety, harmful content) are non-negotiable. Any response with $R_{\text{meta}} < 0.7$ receives a large negative reward.

2. **Content Base Score**: Semantic correctness and structural organization form the foundation of quality, additively combined.

3. **Format as Modulator**: Format constraints scale the base score by $[\alpha, 1.0]$, ensuring format violations reduce rewards by at most 20%.

3.4 Conflict-Aware Weight Adaptation

Real-world instructions often contain implicit conflicts: "be concise but comprehensive," "explain simply with technical precision." We introduce two auxiliary models for dynamic adaptation:

Weight Prediction Model. We train a DeBERTa-base [11] model M_W to predict optimal weights given an instruction:

$$\{w_{\text{sem}}, w_{\text{struct}}, w_{\text{fmt}}, w_{\text{meta}}\} = M_W(x) \tag{10}$$

where $\sum w_i = 1$. M_W is trained on 50,000 instructions with manually annotated target weights.

Conflict Detection Model. We train a BERT-based classifier M_C to detect constraint conflicts:

$$s_{\text{conflict}} = M_C(x) \tag{11}$$

where $s_{\text{conflict}} \in [0, 1]$ is a conflict severity score. When conflicts are detected, we adjust weights to prioritize semantic correctness:

$$w_{\text{adjusted}} = \begin{cases} w_{\text{sem}} + \min(s_{\text{conflict}}, 0.3) \\ w_{\text{struct}} \times (1 - 0.5 \cdot s_{\text{conflict}}) \\ w_{\text{fmt}} \times (1 - 0.7 \cdot s_{\text{conflict}}) \\ w_{\text{meta}} \end{cases} \tag{12}$$

with renormalization to ensure $\sum w_{\text{adjusted},i} = 1$.

4 Training Procedure

4.1 Reward Model Training

We initialize E_θ from a Nemotron-7B checkpoint that has been supervised fine-tuned on high-quality instruction-response pairs. Training uses batch size 32, learning rate 5×10^{-6} with cosine decay, AdamW optimizer, and FP16 mixed precision.

Memory Optimization. We use FSDP (Fully Sharded Data Parallel) to distribute the shared encoder across GPUs, with activation checkpointing every 4 layers. Combined with FP16 mixed precision, this reduces per-step memory by approximately 38% compared to naive implementation. Training takes approximately 12 hours on 8×A100 GPUs.

Validation accuracy per aspect: R_{sem}: 84.2%, R_{struct}: 79.6%, R_{fmt}: 91.3%, R_{meta}: 87.8%.

4.2 RLHF with PPO

We use Proximal Policy Optimization [20] for policy training. The policy π_θ is initialized from the same SFT checkpoint. Each iteration consists of: (1) rollout generation (2048 prompts × 4 responses), (2) decomposed reward computation with conflict-aware weighting, (3) advantage estimation with per-prompt baselines, and (4) PPO update for 4 epochs with mini-batch size 512.

We train for 10,000 iterations (approximately 530 GPU-hours on 8×A100). KL penalty β is adaptively adjusted to maintain $\text{KL}(\pi_\theta \| \pi_{\text{ref}}) \in [0.15, 0.25]$.

5 Experiments

5.1 Experimental Setup

Evaluation Benchmarks. (1) **IFEval** [27]: 541 prompts with 25 verifiable constraint types. (2) **GSM8K** [7]: 1,319 grade-school math problems. (3) **HumanEval** [4]: 164 programming problems. (4) **MT-Bench** [26]: 80 multi-turn conversations rated by GPT-4.

Baselines. (1) Standard RLHF: monolithic reward model + PPO. (2) SFT-only: supervised fine-tuned model without RL. (3) DPO [18]: direct preference optimization. (4) Best-of-N: generate N=16 samples and rerank.

Table 1: Performance on instruction-following and reasoning benchmarks. Values show mean ± std across 5 runs.

Method	**IFEval (Prompt)**	**GSM8K**	**HumanEval**	**MT-Bench**
SFT-only	38.4 ± 1.2	68.7 ± 1.4	61.2 ± 2.1	6.83 ± 0.12
DPO	46.8 ± 1.5	74.1 ± 1.3	67.8 ± 1.9	7.34 ± 0.14
Best-of-N (N=16)	52.3 ± 1.8	76.2 ± 1.2	68.4 ± 2.0	7.58 ± 0.11
Standard RLHF	41.2 ± 1.8	72.3 ± 1.5	65.2 ± 2.2	7.12 ± 0.15
Ours (Full)	**73.8 ± 1.6**	**87.9 ± 1.1**	**76.3 ± 1.7**	**8.52 ± 0.10**
Δ vs. RLHF	+32.6	+15.6	+11.1	+1.40

5.2 Main Results

Our method achieves 73.8% prompt-level accuracy on IFEval, a 79% relative improvement over standard RLHF. Gains extend to math reasoning (GSM8K +15.6), code generation (HumanEval +11.1), and conversational quality (MT-Bench +1.4), suggesting decomposition improves general reasoning beyond constraint satisfaction.

5.3 Ablation Studies

Table 2: Ablation of key components on IFEval. Contribution percentages based on total improvement of 32.6 points.

Configuration	**Accuracy (%)**	Δ	**Contribution**
Baseline (Standard RLHF)	41.2 ± 1.8	—	—
+ Decomposition only (4 heads, linear)	58.7 ± 1.5	+17.5	54%
+ Hierarchical combination	64.3 ± 1.4	+23.1	+17%
+ Weight prediction (M_W)	69.1 ± 1.3	+27.9	+15%
+ Conflict detection (M_C)	73.8 ± 1.6	+32.6	+14%

Key findings: (1) **Decomposition is the primary driver**: Simply separating rewards into 4 components yields +17.5 points (54% of total gain). (2) **Hierarchical combination adds 5.6 points** (17%): Structured composition with safety gates and format modulation provides meaningful improvement. (3) **Auxiliary models contribute 9.5 points** (29%): Weight prediction and conflict detection together enable context-dependent optimization.

Number of Heads. We tested decomposition granularity:

Heads	**IFEval**	**Inter-Head Corr.**
1 (monolithic)	41.2%	N/A
2 (content + format)	54.3%	0.48
4 (full decomposition)	58.7%	0.35
6 (over-decomposition)	55.2%	0.58

Four heads provide optimal granularity. Too few heads conflate distinct constraints; too many create interdependence (high correlation) and data sparsity.

Table 3: Training cost comparison.

Component	**GPU-Hours**	**Cost (8×A100)**
Standard RLHF	380	\$12,453
Ours: Reward Model	12	\$393
Ours: Meta-Models	4	\$131
Ours: PPO Training	530	\$17,368
Ours: Total	**546**	**\$17,892**

5.4 Computational Efficiency

Our method adds 44% training cost but delivers 79% relative performance improvement, yielding superior cost-effectiveness. Inference overhead is only 4.8% (154ms vs. 147ms per sample).

6 Discussion

6.1 Why Does Decomposition Work?

We identify three mechanisms:

Clearer Gradient Signals. With decomposition, each component provides an independent gradient. If only format fails (R_{fmt} low, others high), gradients primarily flow through the format path. We measured gradient-violation correlation: 0.76 for our method vs. 0.34 for standard RLHF—gradients are 2.2× more aligned with actual constraint violations.

Pareto-Optimal Solutions. Decomposition enables explicit Pareto frontier exploration. Our hierarchical combination searches this frontier, prioritizing semantic correctness while allowing format trade-offs.

Reduced Reward Hacking. Monolithic rewards can be gamed through spurious correlations. Decomposition makes exploitation harder: gaming structural reward doesn't help if semantic correctness is independently evaluated.

6.2 Failure Modes

Precise Counting (26.2% of failures). Instructions requiring exact counts ("exactly 100 words") remain challenging due to tokenization mismatch and lack of lookahead during autoregressive generation.

Complex Instructions (18.1% of failures). Performance degrades with constraint count: 1–2 constraints achieve 89.4%, 3–4 achieve 76.8%, 5–6 achieve 58.2%, 7+ achieve 34.1%.

6.3 Limitations

1. **Annotation Cost**: Aspect-level labeling requires 3× more time than overall preferences (\$18K for 180K pairs).
2. **Fixed Decomposition**: We manually specified 4 constraint types; automatic discovery is future work.
3. **Language-Specific**: Evaluation focuses on English; cross-lingual validation needed.

7 Conclusion

We introduced constraint decomposition, a framework for multi-objective instruction-following in large language models. By factorizing reward functions into orthogonal components—semantic correctness, structural organization, format specifications, and meta-level requirements—we eliminate credit assignment ambiguity inherent in monolithic rewards. Our hierarchical combination function respects constraint dependencies, and conflict-aware weight adaptation enables graceful degradation when instructions contain incompatible requirements.

Empirically, constraint decomposition achieves 73.8% accuracy on IFEval (+32.6 points over standard RLHF), with strong generalization to math reasoning, code generation, and conversational quality. Ablation studies confirm that decomposition contributes 54% of the gain, with hierarchical combination and conflict adaptation providing the remainder.

Beyond instruction-following, constraint decomposition provides a general framework for compositional optimization in LLMs. Future work can extend this to multi-modal generation, process supervision, and user-controllable constraint priorities.

Ethics Statement

This work aims to improve LLM instruction-following reliability. We acknowledge dual-use concerns: better instruction-following could facilitate harmful use cases. Our safety gate provides mitigation by blocking harmful content. We recommend red-teaming, monitoring, and human oversight for deployment.

Reproducibility Statement

We provide: (1) complete implementation at `https://github.com/epaunova/constraint-decomposition-llm`
(2) trained model checkpoints, (3) annotation guidelines, (4) hyperparameter specifications. All experiments use fixed random seeds {42, 123, 456, 789, 1024} for multiple runs.

Acknowledgments

We thank the NVIDIA Research team for computational resources and valuable discussions.

References

[1] Yuntao Bai, Andy Jones, Kamal Ndousse, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[2] Leon Barrett and Srini Narayanan. Learning all optimal policies with multiple criteria. In *Proceedings of ICML*, 2008.

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

[4] Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[5] Wei-Lin Chiang, Zhuohan Li, Zi Lin, et al. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. Blog post, 2023.

[6] Paul F Christiano, Jan Leike, Tom Brown, et al. Deep reinforcement learning from human preferences. In *Advances in NeurIPS*, 2017.

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[8] Simon Coste, James Rowbottom, Jules Lambert, et al. Reward model ensembles help mitigate overoptimization. *arXiv preprint arXiv:2310.02743*, 2023.

[9] Ganqu Cui, Lifan Yuan, Ning Ding, et al. UltraFeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*, 2023.

[10] Conor F Hayes, Rădulescu Roxana, Eugenio Bargiacchi, et al. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1):1–59, 2022.

[11] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTa: Decoding-enhanced BERT with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

[12] Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of ACL*, 2017.

[13] Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Controlling output length in neural encoder-decoders. In *Proceedings of EMNLP*, 2016.

[14] Wing Lian, Bleys Goodson, Eugene Pentland, et al. OpenOrca: An open dataset of GPT augmented FLAN reasoning traces. HuggingFace, 2023.

[15] Ximing Lu, Peter West, Rowan Zellers, et al. NeuroLogic decoding: (Un)supervised neural text generation with predicate logic constraints. In *Proceedings of NAACL*, 2021.

[16] NVIDIA. Nemotron-4 340B technical report. NVIDIA Technical Report, 2024.

[17] Long Ouyang, Jeff Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. In *Advances in NeurIPS*, 2022.

[18] Rafael Rafailov, Archit Sharma, Eric Mitchell, et al. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

[19] Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[21] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. In *Advances in NeurIPS*, 2017.

[22] Nisan Stiennon, Long Ouyang, Jeff Wu, et al. Learning to summarize with human feedback. In *Advances in NeurIPS*, 2020.

[23] Jonathan Uesato, Nate Kushman, Ramana Kumar, et al. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

[24] Peter Vamplew, Richard Dazeley, Adam Berry, Rustam Issabekov, and Evan Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning*, 84(1):51–80, 2011.

[25] Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of Pareto dominating policies. *Journal of Machine Learning Research*, 15(1):3483–3512, 2014.

[26] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, et al. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *arXiv preprint arXiv:2306.05685*, 2023.

[27] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, et al. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.