BIASREDUCER: ADAPTIVE BIAS MITIGATION FOR REWARD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward models score responses from large language models (LLMs) and guide LLM training toward human preferences. However, reward models can favor superficial attributes such as length or confidence, leading LLMs to produce higher-scoring but not more correct responses. Existing mitigation methods either retrain the reward model or apply a fixed correction to one known bias, such as a preference for longer responses. Retraining requires additional data and computational resources, while existing editing methods require the target bias to be specified in advance and use a fixed edit for that bias. To this end, we propose BIASREDUCER, a lightweight framework that edits only the linear reward head and selects the relevant edits for each new dataset. First, BIASREDUCER uses a sparse autoencoder (SAE)-style encoder to learn which attributes (e.g., length and confidence) the reward model is sensitive to. Second, it learns how to reduce the reward model's dependence on each attribute by determining which direction to adjust the reward head and how much to adjust it. Third, for a new dataset, it ranks the attributes by their influence on reward scores, selects the relevant ones, and edits the reward model accordingly. BIASREDUCER consistently improves reward-model robustness to biases toward superficial response attributes. Across five reward models, BIASREDUCER-M improves the three benchmarks by 8.3, 18.0, and 6.9 percentage points on average, outperforming the two training-based baselines. The gains transfer downstream, reducing unnecessary verbosity and sycophancy while maintaining comparable judged quality. The code is available at https://anonymous.4open.science/r/biasreducer-55D3/.

1 INTRODUCTION

Reward models are widely used in reinforcement learning from human feedback (RLHF) to score outputs from large language models (LLMs) and guide training toward human-preferred responses (Stiennon et al., 2020; Ouyang et al., 2022). However, reward models are imperfect proxies for human preferences and may favor response attributes such as length, formatting, confidence, or sycophancy (Liu et al., 2025b; Bharadwaj et al., 2026; Zhang et al., 2025). For example, a longer or more confident response may receive a higher reward even when it is less correct or helpful. Under response selection or downstream LLM training, reward signals tied to superficial attributes can encourage models to optimize for those attributes rather than response quality. This behavior is commonly known as reward hacking (Gao et al., 2023; Coste et al., 2024; Rafailov et al., 2024).

Existing methods address reward-model bias in two main ways: training-based methods and model-editing methods. Training-based methods change the training data, loss, model design, or fine-tuning procedure (Chen et al., 2024; Liu et al., 2025a; Srivastava et al., 2026; Bharadwaj et al., 2026). While effective, these training-based methods can be computationally expensive and often require substantial new training data. Model-editing methods avoid retraining, but usually require the target bias to be specified in advance (Liu et al., 2026b). However, reward models can rely on multiple attributes, and their relevance varies across target distributions. This raises a key question: *can we choose suitable reward-model corrections for each new dataset without retraining the model or applying the same preselected attribute correction across datasets?*

To this end, we propose BIASREDUCER, a lightweight framework that edits only the linear reward head. BIASREDUCER separates the problem into three steps. First, BIASREDUCER learns internal

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

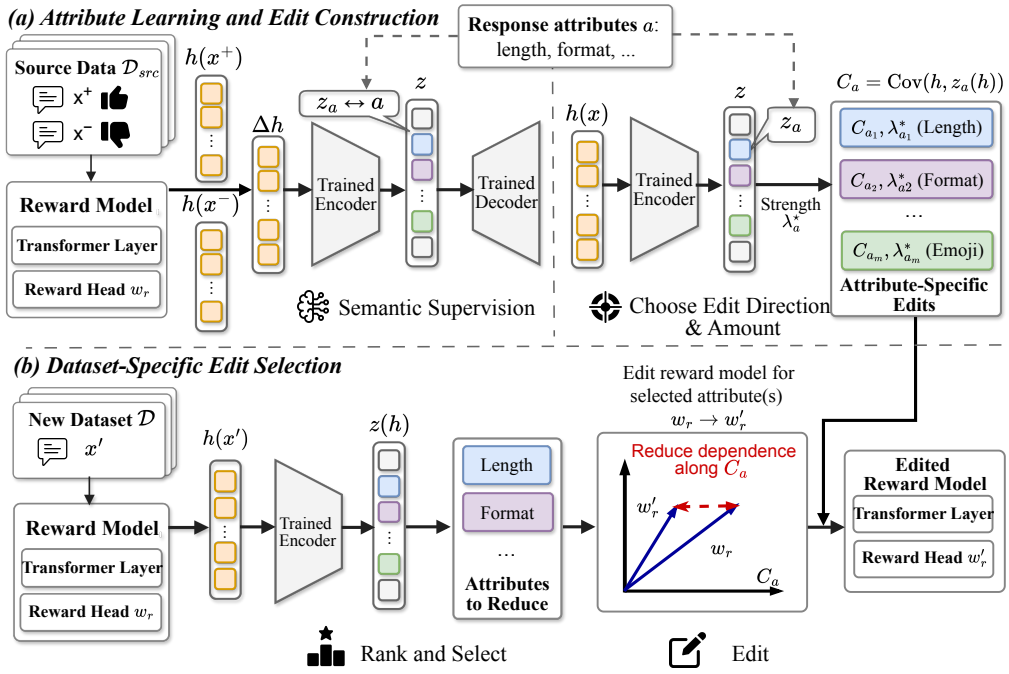


Figure 1: Overview of BIASREDUCER. From preference pairs (x^+, x^-), BIASREDUCER learns a dimension z_a associated with a predefined response attribute a, and determines how to edit the reward head w_r to reduce its dependence on that attribute. For a new dataset $\mathcal{D}$, BIASREDUCER selects what to edit using only responses and reward scores, without knowing which response is preferred. w_r' denotes the edited reward head. $h(x)$ is the reward-model hidden representation, $\Delta h = h(x^+) - h(x^-)$, C_a is the direction associated with a, and $\lambda_a^\star$ controls how much the reward head is edited along C_a.

representations that correspond to predefined attributes, such as length and confidence, giving selected dimensions clear, human-interpretable meanings. Second, it learns how to reduce the influence of each attribute on the reward score by determining the direction in which to adjust the reward head and by how much. Third, for a new dataset, it determines which attributes matter most and edits the reward model accordingly. For example, if reward scores depend strongly on length but little on confidence, it applies the length-related edit rather than the confidence-related one.

We consider two variants: BIASREDUCER-S selects a single correction for each dataset, while BIASREDUCER-M combines multiple corrections. We evaluate both variants across five reward models on RM-Bench-Hard (Liu et al., 2025b), JudgeBiasBench (Zhou et al., 2026), and Arena-StyleConflict, which is constructed from Arena Human Preference 140K (lmarena-ai, 2025). Our experiments show that across five reward models, BIASREDUCER-M improves the three benchmarks by 8.3, 18.0, and 6.9 percentage points on average, outperforming the two training-based baselines. These gains also transfer to downstream LLM training, reducing unnecessary verbosity and sycophancy. Ablations further show that semantic supervision, choosing the correction direction and strength, and dataset-specific edits all contribute to the gains.

Our contributions in this work are threefold:

- We propose BIASREDUCER, a lightweight reward-model editing framework that learns representations of predefined response attributes using semantic supervision.
- We separate learning how to reduce dependence on predefined attributes from deciding which attributes to address for each dataset, enabling the same edits to be reused without retraining or specifying the target bias in advance.
- Experiments across five reward models and three benchmarks show consistent gains over existing baselines, with further benefits from dataset-specific edit selection and combining multiple edits.

2 Related Work

Reward hacking and reward-model bias. Reward models can favor response attributes such as length, formatting, confidence, or agreement with the user rather than response correctness or helpfulness (Liu et al., 2025b; Bharadwaj et al., 2026; Zhang et al., 2025). When these signals guide response selection or LLM training, models may optimize for the rewarded attributes instead of better responses, leading to reward hacking (Gao et al., 2023; Coste et al., 2024; Rafailov et al., 2024).

Sparse representations. Sparse autoencoders (SAEs) have been used to identify interpretable features in reward models. SARM uses sparse features to explain reward-model decisions, while SparseRM uses sparse representations for preference modeling (Zhang et al., 2026; Liu et al., 2026a). However, existing sparse representation and concept-erasure methods do not connect interpretable reward-model attributes with reusable edits for bias mitigation.

Mitigating reward-model bias. Existing methods for mitigating reward-model bias broadly fall into two categories: training-based methods and model-editing methods. Training-based methods modify the reward architecture, objective, data, or fine-tuning procedure: ODIN separates reward signals, while RRM, CROME, and counterfactual fine-tuning improve robustness through training-time changes (Chen et al., 2024; Liu et al., 2025a; Srivastava et al., 2026; Bharadwaj et al., 2026). Model-editing methods modify a trained reward model, for example by removing a subspace or direction associated with a specified bias (Liu et al., 2026b; Fein et al., 2026). These methods reduce reward-model bias but have clear limitations: training-based methods require retraining and new data construction, while existing editing methods apply a fixed correction to a bias specified in advance.

3 Methodology

Reward models can rely on response attributes such as length, confidence, or sycophancy. Our goal is to reduce the reward model's dependence on response attributes and decide which attributes to edit for each new dataset, without retraining the reward model. BiasReducer addresses this goal in three stages. First, BiasReducer learns internal representations that correspond to predefined response attributes, such as length and confidence, giving selected dimensions clear, human-interpretable meanings. Second, it learns how to reduce the reward model's sensitivity to each attribute. Third, for a new dataset, it determines which attribute dependencies should be reduced and edits the reward model accordingly. Consider response length as an example. Word-count differences teach one dimension to represent length, while controlled lengthening and shortening rewrites show how it changes with length. Using this representation, BiasReducer determines how the reward head should be adjusted, and by how much, to reduce its dependence on length. For a new dataset, if reward scores depend more strongly on length than on confidence, BiasReducer prioritizes the length edit. Figure 1 summarizes the main framework. Full method details are provided in Appendix B.

3.1 Problem Statement

Given a user prompt q and a candidate response x, a scalar reward model outputs a reward score $r(q, x)$, where higher scores indicate stronger preference for the response. For example, the score can be used to choose the highest-scoring response from a set of candidates or as the reward signal for policy training. We consider a reward model with a linear reward head,

$$r(q, x) = w_r^\top h(q, x) + b,$$

where $h(q, x) \in \mathbb{R}^d$ is the hidden representation used as input to the reward head, w_r is the reward-head weight, and b is the constant term. For notational simplicity, we write $r(x)$ and $h(x)$ for $r(q, x)$ and $h(q, x)$, respectively. All reward-model parameters remain fixed except w_r; BiasReducer edits w_r to obtain w_r'. We denote the predefined response attributes by $\mathcal{A}$, and $a \in \mathcal{A}$ for attributes such as length, confidence, and sycophancy. We use source preference data $\mathcal{D}_{\text{src}}$, split into a training set $\mathcal{D}_{\text{src}}^{\text{train}}$ and a held-out validation set $\mathcal{D}_{\text{src}}^{\text{val}}$. Each source pair (x^+, x^-) contains a preferred response x^+ and a rejected response x^-, with preference contrast $\Delta h = h(x^+) - h(x^-)$. For a new dataset $\mathcal{D}$, BiasReducer may use its responses, hidden representations, and original reward scores, but does not know which response in each pair is preferred.

3.2 Learning Representations of Response Attributes

The first step is to identify which internal patterns in the reward model correspond to specific response attributes, such as length, sycophancy, and confidence. A preference contrast Δh can reflect several response attributes at once. For example, the preferred response x^+ may be both longer and more confident than the rejected response x^-. BiasReducer therefore learns a latent representation and trains selected dimensions to represent predefined response attributes. Given a preference contrast $u = \Delta h$ from the training set of source data $\mathcal{D}_{\text{src}}^{\text{train}}$, we train a sparse autoencoder (SAE)-style encoder–decoder. The encoder maps u to a latent representation and keeps only the k_{act} largest activations in magnitude, while the decoder reconstructs the original contrast:

$$z_{\text{pre}} = W_{\text{enc}} u + b_{\text{enc}}, \qquad z_{\text{post}} = \text{TopK}(z_{\text{pre}}), \qquad \hat{u} = V z_{\text{post}}, \tag{1}$$

where W_{enc} denotes the encoder weight matrix, b_{enc} denotes the encoder bias, and V denotes the decoder matrix. $z_{\text{pre}}, z_{\text{post}} \in \mathbb{R}^K$ denote the latent representations before and after top-k_{act} sparsification, respectively. Each coordinate is one scalar dimension of the latent representation. The top-k_{act} operator retains the coordinates with the largest absolute activations while preserving their signs.

A sparse decomposition alone does not determine which coordinate represents which response attribute. We therefore assign one coordinate z_a to each predefined attribute $a \in \mathcal{A}$ and use semantic supervision to give these coordinates interpretable meanings. Behavioral supervision $\mathcal{L}_{\text{beh}}$ links z_a to natural differences in the corresponding attribute, while interventional supervision $\mathcal{L}_{\text{int}}$ establishes its direction using controlled rewrites. For example, the length coordinate tracks natural differences in response length, while lengthening and shortening rewrites determine which direction corresponds to longer responses. A separate artifact term $\mathcal{L}_{\text{art}}$ captures changes shared across controlled rewrites so that they are not assigned to the named attributes. We jointly train the encoder–decoder with

$$\mathcal{L} = \mathcal{L}_{\text{rep}} + \gamma_b \widetilde{\mathcal{L}}_{\text{beh}} + \gamma_i \widetilde{\mathcal{L}}_{\text{int}} + \gamma_a \widetilde{\mathcal{L}}_{\text{art}}, \tag{2}$$

where $\mathcal{L}_{\text{rep}}$ reconstructs preference contrasts with sparsity regularization, $\gamma_b, \gamma_i, \gamma_a$ weight the three semantic-supervision terms, and $\widetilde{\mathcal{L}}$ denotes their scale-normalized versions.

3.3 Reducing the Reward Model's Dependence on Response Attributes

The second step is to determine how the reward model should be edited to reduce its dependence on each attribute, including which direction to edit it in and how much to edit it. The learned attribute coordinates z_a tell us how to detect each response attribute, but not how the reward head depends on it or how strongly it should be corrected. We therefore identify a direction for each attribute along which the reward head can be edited to reduce its dependence on that attribute. For each attribute a, we apply its learned coordinate z_a to responses from the training set of source data $\mathcal{D}_{\text{src}}^{\text{train}}$ and estimate

$$C_a = \text{Cov}_{h \sim \mathcal{D}_{\text{src}}^{\text{train}}} \left(h, z_{\text{post},a}(h) \right), \tag{3}$$

where C_a captures the hidden-state direction that co-varies with the attribute activation. Motivated by linear concept erasure (Belrose et al., 2023), we define a candidate edit for each attribute along C_a:

$$w_a(\lambda) = w_r - \lambda \frac{w_r^\top C_a}{C_a^\top C_a + \epsilon_{\text{edit}}} C_a, \tag{4}$$

where λ controls how far the reward head is adjusted along C_a. For each attribute, we determine how far to adjust the reward head by choosing the edit strength $\lambda_a^\star$ that induces the largest reward-score change while keeping the preference accuracy drop on $\mathcal{D}_{\text{src}}^{\text{val}}$ within a fixed budget. For each eligible attribute, C_a and $\lambda_a^\star$ specify the direction and amount of the reward-head edit. We collect these attribute-specific edits into an edit bank $\mathcal{B} = \{(a, C_a, \lambda_a^\star)\}$, which BiasReducer uses to decide which edits to apply for a new dataset.

3.4 Selecting What to Edit for a New Dataset

The third step is to determine which attributes influence the reward scores in a new dataset. BiasReducer then adjusts the reward model to reduce its dependence on those attributes. Given a new dataset $\mathcal{D}$, BiasReducer ranks the candidate corrections using only its responses and original

reward scores, without knowing which response in each pair is preferred. For each attribute a, BIASREDUCER computes two signals. The first measures how strongly the original reward varies with attribute a on the new dataset: $G_a(\mathcal{D}) = \left|\mathrm{Cov}_{h\sim\mathcal{D}}(w_r^\top h, z_{\mathrm{pre},a}(h))\right|$, where $z_{\mathrm{pre},a}(h)$ is the continuous pre-top-k_{act} activation of attribute a. The second measures how much the corresponding attribute-specific edit changes the reward scores: $I_a(\mathcal{D}) = \mathbb{E}_{h\sim\mathcal{D}}\left|h^\top w_a(\lambda_a^\star) - h^\top w_r\right|$. We rank candidate corrections separately by G_a and I_a. Following Borda-style rank aggregation (Dwork et al., 2001), we combine the two rankings by their rank sum, where a smaller rank sum indicates higher selection priority:

$$B_a(\mathcal{D}) = \mathrm{rank}_{\downarrow G}(a) + \mathrm{rank}_{\downarrow I}(a). \tag{5}$$

BIASREDUCER-S selects the highest-ranked eligible attribute to address, while BIASREDUCER-M combines edits for up to $k_{\max}$ attributes. Algorithm 1 summarizes the main BIASREDUCER framework.

Algorithm 1 BIASREDUCER

Require: RM (h, w_r); source data $\mathcal{D}_{\mathrm{src}}$ split into $\mathcal{D}_{\mathrm{src}}^{\mathrm{train}}$ and $\mathcal{D}_{\mathrm{src}}^{\mathrm{val}}$; attributes $\mathcal{A}$; new dataset $\mathcal{D}$
Ensure: edited reward heads $w_{\mathrm{S}}^\star$ and $w_{\mathrm{M}}^\star$

```
Learn attribute-aligned representations using preference contrasts from D_src^train
for a ∈ A do
    Estimate source-side covariance direction C_a using D_src^train
    Select λ_a^⋆ to control how far the reward head is edited along C_a using D_src^val
end for
Freeze edit bank B = {(a, C_a, λ_a^⋆)}
Rank candidate edits on D without knowing which responses are preferred
Let (a_1, ..., a_|B|) denote the resulting order
BIASREDUCER-S: w_S^⋆ ← w_{a_1}(λ_{a_1}^⋆)
BIASREDUCER-M: w_M^⋆ ← Compose(w_r, (a_1, ...), B)
```

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Reward models and source data. The experiments cover five scalar reward models: Skywork-Reward-V2-Qwen3-1.7B (Qwen3-1.7B) (Liu et al., 2026c), RM-Mistral-7B (Mistral-7B) (Xiong et al., 2024), GRM-Llama3.2-3B-rewardmodel-ft (GRM-3B) (Yang et al., 2024), internlm2-7b-reward (InternLM2-7B) (Cai et al., 2024), and GRM-Llama3.1-8B-rewardmodel-ft (GRM-8B) (Yang et al., 2024). We use Skywork-Reward-Preference-80K-v0.2 (Liu et al., 2024) as the source data for learning the attribute representations and determining how to edit the reward model for each attribute.

Benchmarks and baselines. We evaluate on two existing bias-focused benchmarks, RM-Bench-Hard (Liu et al., 2025b) and JudgeBiasBench (Zhou et al., 2026), together with Arena-StyleConflict, a style-conflict test set constructed from Arena Human Preference 140K (lmarena-ai, 2025). We compare BIASREDUCER-S and BIASREDUCER-M with the original reward model and two fine-tuning baselines. The first is an RRM-based baseline, which further fine-tunes the existing reward model using the RRM mitigation procedure (Liu et al., 2025a); the second baseline, intervention fine-tuning, uses our constructed attribute-intervention pairs mixed with source data to further fine-tune the same reward model. We report pairwise preference accuracy, $\mathrm{Acc} = \frac{1}{N}\sum_i \mathbb{1}[r(x_i^+) > r(x_i^-)]$, where x_i^+ is the benchmark-preferred response. Appendix A provides detailed information.

4.2 MAIN RESULTS

Table 1 shows two major findings. First, BIASREDUCER improves pairwise preference accuracy across reward models and bias-focused benchmarks. Both BIASREDUCER-S and BIASREDUCER-M outperform the original reward model in every model–benchmark combination. Second, BIASREDUCER-M further achieves the strongest average performance on all three benchmarks, indicating that combining multiple attribute-specific corrections can provide additional benefits over selecting a single correction. Averaged over five reward models, BIASREDUCER-S improves the

Table 1: Main evaluation of reward-model bias mitigation. RRM denotes the RRM-based training baseline; Interv. FT denotes intervention fine-tuning. Arena denotes Arena-StyleConflict, and JBB denotes JudgeBiasBench. We report pairwise preference accuracy (%); superscripts show gains over the original model in percentage points, and the last row gives the five-model average for each benchmark. Bold = best result in each row; shaded columns = our methods. ↑ higher is better.

Reward model	Original	RRM	Interv. FT	BIASREDUCER-S	BIASREDUCER-M
RM-Bench-Hard ↑					
Qwen3-1.7B	59.96	65.29$^{+5.33}$	61.42$^{+1.46}$	64.38$^{+4.42}$	**66.92**$^{+6.96}$
GRM-3B	48.93	52.50$^{+3.57}$	50.01$^{+1.08}$	52.98$^{+4.04}$	**54.08**$^{+5.15}$
Mistral-7B	44.64	49.08$^{+4.44}$	51.70$^{+7.06}$	53.47$^{+8.83}$	**55.38**$^{+10.74}$
InternLM2-7B	58.25	60.84$^{+2.59}$	57.42$^{-0.83}$	62.65$^{+4.40}$	**63.93**$^{+5.68}$
GRM-8B	50.89	56.02$^{+5.13}$	52.10$^{+1.21}$	60.11$^{+9.22}$	**63.90**$^{+13.01}$
Mean	52.53	56.75$^{+4.21}$	54.53$^{+2.00}$	58.72$^{+6.18}$	**60.84**$^{+8.31}$
Arena-StyleConflict ↑					
Qwen3-1.7B	22.90	30.20$^{+7.30}$	26.50$^{+3.60}$	39.25$^{+16.35}$	**39.65**$^{+16.75}$
GRM-3B	36.20	40.30$^{+4.10}$	38.90$^{+2.70}$	42.15$^{+5.95}$	**43.55**$^{+7.35}$
Mistral-7B	20.30	27.50$^{+7.20}$	35.30$^{+15.00}$	51.65$^{+31.35}$	**56.05**$^{+35.75}$
InternLM2-7B	50.70	48.90$^{-1.80}$	48.20$^{-2.50}$	67.00$^{+16.30}$	**69.55**$^{+18.85}$
GRM-8B	34.10	41.20$^{+7.10}$	36.30$^{+2.20}$	42.50$^{+8.40}$	**45.15**$^{+11.05}$
Mean	32.84	37.62$^{+4.78}$	37.04$^{+4.20}$	48.51$^{+15.67}$	**50.79**$^{+17.95}$
JudgeBiasBench ↑					
Qwen3-1.7B	73.93	**77.28**$^{+3.35}$	75.99$^{+2.06}$	75.21$^{+1.29}$	75.64$^{+1.72}$
GRM-3B	70.10	76.37$^{+6.27}$	72.29$^{+2.19}$	76.74$^{+6.64}$	**78.93**$^{+8.83}$
Mistral-7B	63.19	63.53$^{+0.34}$	68.69$^{+5.50}$	75.02$^{+11.83}$	**76.55**$^{+13.36}$
InternLM2-7B	57.82	62.33$^{+4.51}$	61.90$^{+4.08}$	63.45$^{+5.63}$	**63.68**$^{+5.86}$
GRM-8B	70.06	**78.39**$^{+8.33}$	73.71$^{+3.65}$	74.79$^{+4.73}$	74.66$^{+4.60}$
Mean	67.02	71.58$^{+4.56}$	70.52$^{+3.50}$	73.04$^{+6.02}$	**73.89**$^{+6.87}$

Table 2: Policy training using GRPO. Qwen3-4B and Llama-3.2-3B-Instruct are trained using the original or BIASREDUCER-M-edited Mistral-7B reward model. Initial denotes the policy before GRPO training. Claude Sonnet 5 evaluates response quality (Qual., 1–10), unnecessary verbosity (Verb.), and sycophancy (Syco.); Words denotes mean response length.

Policy	Training reward	Qual. ↑	Verb. ↓	Syco. ↓	Words
Qwen3-4B	*Initial*	4.24	0.53	0.06	274
	Original RM	4.29	0.74	0.08	331
	BIASREDUCER-M	4.53	0.58	0.04	285
Llama-3.2-3B	*Initial*	3.85	0.66	0.06	301
	Original RM	4.02	0.70	0.11	318
	BIASREDUCER-M	4.01	0.55	0.03	291

three benchmarks by 6.2, 15.7, and 6.0 percentage points, respectively. BIASREDUCER-M increases these gains to 8.3, 18.0, and 6.9 percentage points. The improvements also exceed those of the training-based baselines: compared with RRM, the stronger baseline on average, BIASREDUCER-M provides an additional 4.1, 13.2, and 2.3 percentage points on the three benchmarks.

4.3 DOWNSTREAM REWARD OPTIMIZATION

The gains also transfer downstream: edited rewards reduce verbosity and sycophancy in both response selection and policy training while preserving response quality. We use Qwen3-4B (Yang et al., 2025) and Llama-3.2-3B-Instruct (Meta, 2024) as policy models and evaluate both response selection and policy training with the original and edited Mistral-7B reward model. For response selection, we use best-of-N, where the policy generates N candidate responses and the reward model selects the highest-scoring one. For policy training, we use Group Relative Policy Optimization (GRPO) (Shao et al., 2024), where the reward model provides the training signal for updating the policy. Claude Sonnet 5 serves as an independent evaluator of response quality, unnecessary verbosity, and sycophancy (Anthropic, 2026). Additional results are reported in Appendix D.

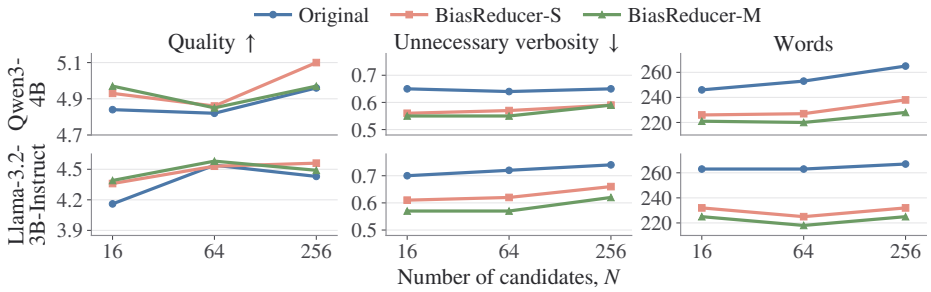


Figure 2: Best-of-N selection. Qwen3-4B (top) and Llama-3.2-3B-Instruct (bottom) are evaluated for $N \in \{16, 64, 256\}$ on the same fixed candidate pools. Claude Sonnet 5 evaluates response quality (1–10) and unnecessary verbosity; Words denotes mean response length.

Table 3: Analysis of semantic supervision. Left: train-time semantic supervision vs. post-training attribute identification for BIASREDUCER-S and BIASREDUCER-M. Right: ablation of individual supervision signals. Corr. is the held-out $|\mathrm{corr}|$ between each attribute's coordinate and its supervision signal, averaged over attributes; Axis, Leak., and Sign are intrinsic dictionary metrics (attribute-axis correlation, cross-attribute leakage, and intervention sign accuracy); all other entries are mean gains in pairwise preference accuracy (pp) across five reward models.

Training strategy						*Supervision ablation*						
Representation	**Variant**	**Corr.**	**RMB**	**Arena**	**JBB**	**Variant**	**Axis**	**Leak.**	**Sign**	**RMB**	**Arena**	**JBB**
Vanilla SAE	BIASREDUCER-S	0.26	+4.36	+6.90	+1.05	Full	**0.55**	**0.08**	0.99	**+8.31**	**+17.95**	**+6.87**
Vanilla SAE	BIASREDUCER-M	0.26	+4.99	+7.41	+1.09	w/o L_{beh}	0.17	0.31	**1.00**	+7.12	+13.70	+2.77
BIASREDUCER (full)	BIASREDUCER-S	**0.55**	**+6.18**	**+15.67**	**+6.02**	w/o L_{int}	0.53	**0.08**	0.75	+6.59	+14.74	+5.76
BIASREDUCER (full)	BIASREDUCER-M	**0.55**	**+8.31**	**+17.95**	**+6.87**	w/o L_{art}	**0.55**	0.09	**1.00**	+6.68	+15.25	+6.14

5 ANALYSIS AND INSIGHTS

Our extensive experiments further indicate four major findings. First, semantic supervision helps the learned dimensions better represent their intended response attributes and leads to stronger reward-model editing. Second, effective editing depends on both choosing the right direction and deciding how much to edit the reward head. Third, different datasets benefit from reducing different attributes, making dataset-specific edit selection more effective. Finally, BIASREDUCER's performance is relatively stable across edit-complexity settings.

5.1 SEMANTIC SUPERVISION IMPROVES ATTRIBUTE ALIGNMENT AND EDITING

Semantic supervision helps the learned coordinates better capture the intended response attributes and improves editing performance. We compare BIASREDUCER with a vanilla SAE trained on the same preference pairs. For the vanilla SAE, we assign each attribute to the latent coordinate most correlated with its measured difference across response pairs (e.g., word-count difference for length). As shown in Table 3, BIASREDUCER increases the mean held-out attribute correlation from 0.26 to 0.55 and yields substantially larger editing gains. We further ablate the three supervision signals to examine their individual contributions. Table 3 reports three intrinsic measures: attribute-axis correlation tracks the intended attribute, cross-attribute leakage captures associations with other attributes, and sign accuracy measures whether controlled rewrites move the coordinate in the expected direction. Removing behavioral supervision lowers attribute-axis correlation from 0.55 to 0.17, while removing intervention supervision reduces sign accuracy from 0.99 to 0.75. Artifact supervision has a smaller intrinsic effect but still improves editing across all three benchmarks. Together, the three signals play complementary roles in learning interpretable attributes and effective edits.

5.2 REWARD-AWARE DIRECTIONS AND ATTRIBUTE-SPECIFIC STRENGTHS IMPROVE EDITING

Reward-relevant edit directions and attribute-specific strengths both improve editing performance. To test the edit direction, we compare our covariance-based direction with the SAE decoder vector

Table 4: Analysis of edit construction and deployment. Mean change in pairwise preference accuracy (pp) across five reward models. *Left*: BIASREDUCER-S with different edit directions (decoder vs. covariance) and edit amounts (preset λ, one shared λ chosen on the validation set of source data, or separate $\lambda_a^\star$ for each attribute). *Right*: same attribute across all datasets (Syco = sycophancy; MaxSrc = the attribute whose edit performs best on the validation set of source data) versus selecting attributes for each dataset. Shaded columns mark the default settings; bold marks the best result in each row.

Single-edit construction (BIASREDUCER-S)

	Direction		*Strength*			
	Dec.	Cov.	λ=1	λ=.5	Global	Per-attr.
RMB	+0.54	**+6.18**	**+7.68**	+1.30	+4.95	+6.18
Arena	+0.40	**+15.67**	+12.60	+2.71	+9.67	**+15.67**
JBB	+0.59	**+6.02**	+4.10	+1.61	+4.18	**+6.02**

Edit Selection

	Fixed Edit		*Dataset-Specific*	
	Syco	MaxSrc	BIASREDUCER-S	BIASREDUCER-M
RMB	+3.93	+6.58	+6.18	**+8.31**
Arena	+5.01	+10.07	+15.67	**+17.95**
JBB	+1.79	+1.00	+6.02	**+6.87**

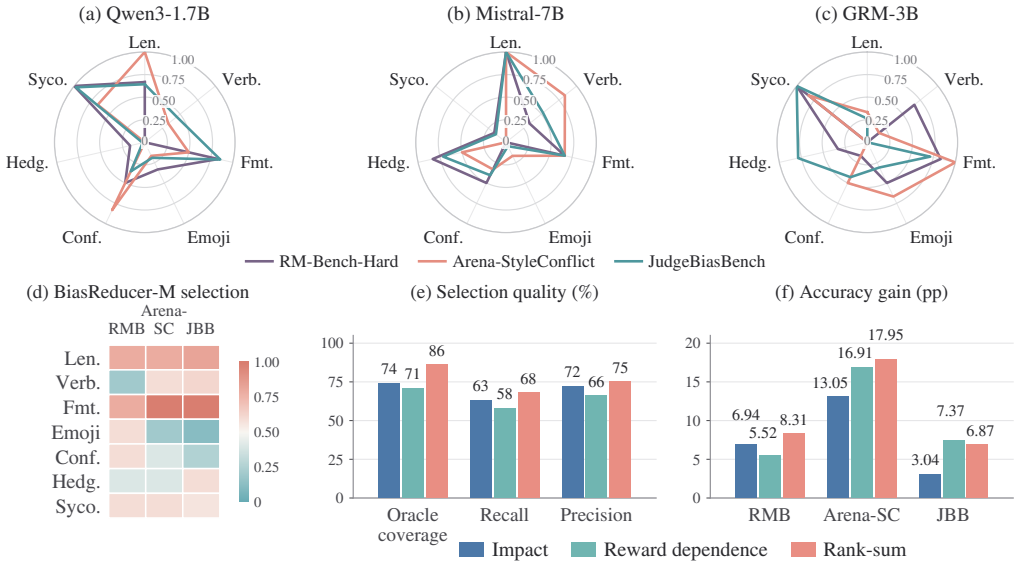
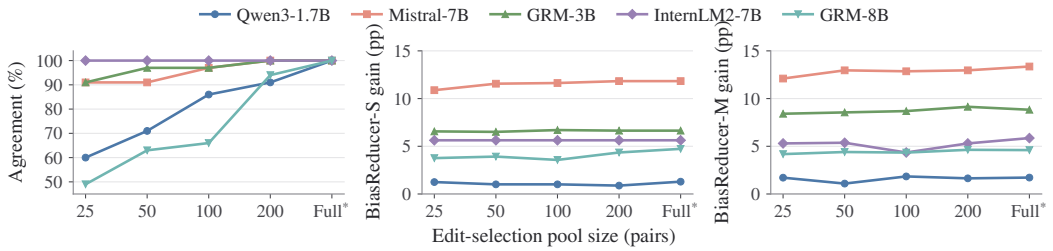


Figure 3: Analysis of dataset-specific edit selection. (a–c) Selected attributes across reward models and benchmarks. (d) BIASREDUCER-M attribute-selection frequency across the five reward models; each cell shows the fraction of model–benchmark cases in which the corresponding attribute is selected on that benchmark. (e) JudgeBiasBench selection quality for three signals: edit impact, reward dependence, and rank-sum. (f) Mean BIASREDUCER-M gains under three signals. Attribute abbreviations: Len = length, Verb = verbosity, Fmt = formatting, Conf = confidence, Hedg. = hedging, Syco = sycophancy, and Emoji = emoji use. Oracle coverage measures how often the selected edits contain the candidate edit that performs best on the benchmark.

associated with each attribute coordinate. As shown in Table 4, using these decoder vectors produces average gains of only 0.5, 0.4, and 0.6 percentage points on the three benchmarks, compared with 6.2, 15.7, and 6.0 points using our covariance-based direction. We also compare one validation-selected strength shared across attributes with separate strengths for each attribute. On JudgeBiasBench with Mistral-7B, the gain increases from 8.4 to 11.8 points.

5.3 DIFFERENT DATASETS BENEFIT FROM DIFFERENT ATTRIBUTE-SPECIFIC EDITS

The most useful attribute edits vary across datasets and reward models, making a single fixed edit less effective than dataset-specific attribute selection. Figure 3(a–d) shows that attribute selections differ across datasets and reward models, and that BIASREDUCER-M selects different combinations of corrections across the three benchmarks. Table 4 shows that BIASREDUCER-M improves the three benchmarks by 8.3, 18.0, and 6.9 percentage points on average, outperforming both a fixed sycophancy edit and the strongest source-selected fixed edit. We next examine how candidate



(a) Agreement (b) BiasReducer-S gain (c) BiasReducer-M gain

Figure 4: Sensitivity to the amount of unlabeled target data. (a) Agreement with the attribute selection decision obtained from the full pool. (b–c) Pairwise-accuracy gains for BIASREDUCER-S and BIASREDUCER-M.

Table 5: Sensitivity to editing and representation settings. Entries report mean BIASREDUCER-M gains in pairwise preference accuracy (pp) across five reward models on RM-Bench-Hard (RMB), Arena-StyleConflict (Arena), and JudgeBiasBench (JBB). Shaded columns denote the default settings.

	Edit-complexity cap $k_{\max}$				**Dictionary size**		**Active coordinates**	
	2	3	4	6	$K/2$	K	$K/4$	$K/2$
RMB	+8.35	+7.70	+8.31	+8.44	+9.38	+8.31	+5.56	+8.31
Arena	+17.67	+18.54	+17.95	+17.41	+17.30	+17.95	+7.38	+17.95
JBB	+6.39	+6.45	+6.87	+7.07	+4.68	+6.87	+1.68	+6.87

corrections should be ranked. Figure 3(e–f) shows that the rank-sum of reward dependence and edit impact performs better than either signal alone. On JudgeBiasBench, the rank-sum rule includes the best eligible correction in 86% of cases, compared with 71–74% for either signal alone, and also gives the largest average gains across the three benchmarks. Finally, attribute selection remains stable with substantially fewer responses. As shown in Figure 4, using 100 examples per subset matches the selection decision from the full response pool in 89% of trials, increasing to 97% with 200 examples.

5.4 BIASREDUCER REMAINS STABLE ACROSS EDIT-COMPLEXITY SETTINGS

BIASREDUCER performs similarly across a range of edit-complexity settings, while dictionary size and the number of active coordinates have more benchmark-dependent effects. We vary the maximum number of combined corrections $k_{\max}$, dictionary size, and number of active coordinates. As shown in Table 5, changing $k_{\max} \in \{2, 3, 4, 6\}$ leads to only modest differences. Reducing the dictionary size improves RM-Bench-Hard but lowers gains on the other two benchmarks, while reducing the number of active coordinates lowers gains across all three benchmarks. Overall, the main results do not depend on a particular edit-complexity setting. Detailed results are provided in Appendix C.4.

6 CONCLUSION

We introduced BIASREDUCER, a lightweight framework for reward-model editing that learns reusable attribute-specific corrections and selects which corrections to apply for each new dataset, without retraining the reward model. Across five reward models, BIASREDUCER-M improves the three tested benchmarks by 8.3, 18.0, and 6.9 percentage points on average, outperforming two training-based baselines. These gains also transfer downstream: using the edited reward reduces unnecessary verbosity and sycophancy in best-of-N selection and GRPO training. Our analyses show that semantic supervision, reward-aware edit directions and strengths, and dataset-specific edit selection all contribute to performance. Among the attribute-specific corrections learned by BIASREDUCER, the most effective ones vary across reward models and datasets. Combining multiple corrections also provides additional gains over selecting a single one. More broadly, our results suggest a useful design principle for reward-model editing: learn reusable corrections once, then choose which corrections to apply based on the reward behavior observed in each new dataset.

AI USE STATEMENT

In this work, we used generative AI tools for writing assistance, primarily for language polishing, improving clarity, and proofreading the manuscript. We have not used generative AI tools for other tasks requiring disclosure, and the remaining required disclosure categories are not applicable to this work. All AI-assisted writing was reviewed and revised by the authors to ensure that it accurately reflected the intended technical content. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the code for BIASREDUCER at https://anonymous.4open.science/r/biasreducer-55D3/, which contains the core representation-learning, edit-construction, dataset-specific edit selection, and benchmark-evaluation pipeline. Appendix A summarizes the models, data, benchmarks, and evaluation protocol; Appendix B provides the full method specification; Appendix D describes downstream evaluation and evaluator prompts; Appendix E lists the implementation settings and hyperparameters used in our experiments.

REFERENCES

Anthropic. Introducing claude sonnet 5. https://www.anthropic.com/news/claude-sonnet-5, 2026.

Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. Leace: Perfect linear concept erasure in closed form. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 66044–66063. Curran Associates, Inc., 2023. doi: 10.52202/075280-2884. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d066d21c619d0a78c5b557fa3291a8f4-Paper-Conference.pdf.

Anirudh Bharadwaj, Chaitanya Malaviya, Nitish Joshi, and Mark Yatskar. Flattery, fluff, and fog: Diagnosing and mitigating idiosyncratic biases in preference models. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 19171–19191, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/2096bafd3073a2224f6f0adb594068df-Paper-Conference.pdf.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang, Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. Internlm2 technical report, 2024. URL https://arxiv.org/abs/2403.17297.

Lichang Chen, Chen Zhu, Jiuhai Chen, Davit Soselia, Tianyi Zhou, Tom Goldstein, Heng Huang, Mohammad Shoeybi, and Bryan Catanzaro. ODIN: Disentangled reward mitigates hacking in RLHF. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 7935–7952. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/chen24bn.html.

Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 50905–50931, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/dda7f9378a210c25e470e19304cce85d-Paper-Conference.pdf.

Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pp. 613–622, New York, NY, USA, 2001. Association for Computing Machinery. ISBN 1581133480. doi: 10.1145/371920.372165. URL https://doi.org/10.1145/371920.372165.

Daniel Fein, Max Lamparth, Violet Xiang, Mykel J. Kochenderfer, and Nick Haber. One bias after another: Mechanistic reward shaping and persistent biases in language reward models, 2026. URL https://arxiv.org/abs/2603.03291.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10835–10866. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/gao23h.html.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms, 2024. URL https://arxiv.org/abs/2410.18451.

Dengcan Liu, Jiahao Li, Zheren Fu, Yi Tu, Jiajun Li, Zhendong Mao, and Yongdong Zhang. Sparserm: A lightweight preference modeling with sparse autoencoder. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 32051–32059. AAAI Press, 2026a. doi: 10.1609/AAAI.V40I38.40476. URL https://doi.org/10.1609/aaai.v40i38.40476.

Shuang Liu, Yuxuan Bo, Qiuyang Zhao, Caiyue Huang, Xiaorong Chen, Yanguang Liu, and Mengnan Du. Harve: Hacking-aware reward-head vector editing for robust reward models, 2026b. URL https://arxiv.org/abs/2606.03131.

Tianqi Liu, Wei Xiong, Jie Ren, Lichang Chen, Junru Wu, Rishabh Joshi, Yang Gao, Jiaming Shen, Zhen Qin, Tianhe Yu, Daniel Sohn, Anastasia Makarova, Jeremiah Zhe Liu, Yuan Liu, Bilal Piot, Abe Ittycheriah, Aviral Kumar, and Mohammad Saleh. Rrm: Robust reward model training mitigates reward hacking. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 62682–62700, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/9d46574e5baae5121180228223a11836-Paper-Conference.pdf.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. Rm-bench: Benchmarking reward models of language models with subtlety and style. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 44323–44355, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/6da1eec80095dc5937f7716db15aca4b-Paper-Conference.pdf.

Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, and Yang Liu. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 133805–133838, 2026c. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/d8836d0294651a867c2ca51012923997-Paper-Conference.pdf.

lmarena-ai. arena-human-preference-140k. https://huggingface.co/datasets/lmarena-ai/arena-human-preference-140k, 2025.

Meta. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sikchi, Joey Hejna, W. Bradley Knox, Chelsea Finn, and Scott Niekum. Scaling laws for reward model overoptimization in direct alignment algorithms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 126207–126242. Curran Associates, Inc., 2024. doi: 10.52202/079017-4009. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e45caa3d5273d105b8d045e748636957-Paper-Conference.pdf.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Pragya Srivastava, Harman Singh, Rahul Madhavan, Gandharv Patil, Sravanti Addepalli, Arun Suggala, Rengarajan Aravamudhan, Soumya Sharma, Anirban Laha, Aravindan Raghuveer, Karthikeyan Shanmugam, and Doina Precup. Robust reward modeling via causal rubrics. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 154269–154327, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/fa0a1013e171c1be6121c9bc9fb6589f-Paper-Conference.pdf.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for RLHF under KL-constraint. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54715–54754. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/xiong24a.html.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Rui Yang, Ruomeng Ding, Yong Lin, Huan Zhang, and Tong Zhang. Regularizing hidden states enables learning generalizable reward model for llms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information*

Processing Systems, volume 37, pp. 62279–62309. Curran Associates, Inc., 2024. doi: 10.52202/079017-1990. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/71f7154547c748c8041505521ca433ab-Paper-Conference.pdf.

Shuyi Zhang, Wei Shi, Sihang Li, Jiayi Liao, Tao Liang, Hengxing Cai, and Xiang Wang. Interpretable reward model via sparse autoencoder. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 34808–34816. AAAI Press, 2026. doi: 10.1609/AAAI.V40I41.40783. URL https://doi.org/10.1609/aaai.v40i41.40783.

Xuanchang Zhang, Wei Xiong, Lichang Chen, Tianyi Zhou, Heng Huang, and Tong Zhang. From lists to emojis: How format bias affects model alignment. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26940–26961, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1308. URL https://aclanthology.org/2025.acl-long.1308/.

Hongli Zhou, Hui Huang, Rui Zhang, Kehai Chen, Bing Xu, Conghui Zhu, Tiejun Zhao, and Muyun Yang. Toward robust llm-based judges: Taxonomic bias evaluation and debiasing optimization, 2026. URL https://arxiv.org/abs/2603.08091.

A Additional Experimental Information

This section provides additional details on the reward models, source data, evaluation benchmarks, edit-selection protocol, response attributes, and training-based baselines used in our experiments. Unless otherwise specified, we use the same setup throughout.

A.1 Reward Models

We evaluate BiasReducer on five publicly available scalar reward models spanning different model families and parameter scales. We use each reward model from its publicly released checkpoint. BiasReducer modifies only the linear reward-head weight; all other model parameters remain fixed. Table 6 lists the checkpoints used in our experiments.

Table 6: **Reward models used in our experiments.** The short name is the label used in all result tables and figures.

Short name	Reward Model	Size	Released checkpoint
Qwen3-1.7B	Skywork-Reward-V2-Qwen3-1.7B	1.7B	Skywork/Skywork-Reward-V2-Qwen3-1.7B
GRM-3B	GRM-Llama3.2-3B	3B	Ray2333/GRM-Llama3.2-3B-rewardmodel-ft
Mistral-7B	RM-Mistral-7B	7B	weqweasdas/RM-Mistral-7B
InternLM2-7B	InternLM2-7B-Reward	7B	internlm/internlm2-7b-reward
GRM-8B	GRM-Llama3.1-8B	8B	Ray2333/GRM_Llama3.1_8B_rewardmodel-ft

A.2 Source Preference Data

We use Skywork-Reward-Preference-80K-v0.2 (Liu et al., 2024) to learn the attribute representations and determine how the reward model should be edited for each attribute. After preprocessing and decontamination against the evaluation benchmarks, the remaining examples are split into training and validation sets. The training split is used for representation learning and estimating the attribute-specific edit directions, while the validation split is used to choose how far to edit along each direction.

Table 7: **Source preference data used by BiasReducer.**

Stage	Examples	Use
Raw usable data	77,004	–
After decontamination	66,850	–
Train	53,678	Representation learning / edit directions
Validation	13,172	Edit amount selection

Decontamination and splitting. Before constructing the train–validation split, we remove source examples that overlap with the evaluation benchmarks based on normalized prompt matching or shared sequences of 13 consecutive words. The remaining data are then split into training and validation sets.

A.3 Evaluation Benchmarks

We evaluate on three benchmarks designed to test reward models' performance under response-level biases. RM-Bench-Hard (Liu et al., 2025b) and JudgeBiasBench (Zhou et al., 2026) are existing bias-focused benchmarks that test whether reward models prefer responses for undesirable or superficial reasons. We additionally construct Arena-StyleConflict from Arena Human Preference 140K (lmarena-ai, 2025) to study cases where longer or more heavily formatted responses conflict with human preferences. Table 8 summarizes the three benchmarks.

Arena-StyleConflict construction. We construct Arena-StyleConflict from Arena Human Preference 140K (lmarena-ai, 2025) without generating or rewriting responses. We retain English,

Table 8: **Target benchmarks used for reward-model evaluation.**

Benchmark	Construction	Unit	Primary focus
RM-Bench-Hard	RM-Bench Hard Accuracy	Response pair	Style–quality conflict
JudgeBiasBench	7 Superficial Quality Bias subsets	Response pair	Superficial quality bias
Arena-StyleConflict	From Arena-140K	Response pair	Style–preference conflict

single-turn, non-code examples with a decisive human preference. We identify pairs in which one response is at least $1.3\times$ longer than the other and uses at least as much Markdown formatting, and retain cases where the human preference favors the opposite response. After deterministic sampling, the benchmark contains 1,000 pairs.

RM-Bench-Hard. We use the *Hard Accuracy* setting of RM-Bench, which evaluates the three cells in which the benchmark-preferred response has a lower style level than the rejected response. We refer to this evaluation setting as RM-Bench-Hard throughout the paper.

JudgeBiasBench. We evaluate all seven JudgeBiasBench subsets under its *Superficial Quality Bias* category: length, authority, beauty, assertiveness, sycophancy, sentiment, and concreteness (Zhou et al., 2026). For each subset, we use the response pairs containing the benchmark-provided `rewritten_response`.

A.4 Evaluation and Attribute Edit Selection Protocol

For each benchmark, BiasReducer selects which attributes to edit using only the responses, their hidden representations, and the original reward scores. It does not use information about which response in each pair is preferred. Preference information is used only to compute the final pairwise-accuracy metric.

RM-Bench-Hard and Arena-StyleConflict each use one edit-selection decision. For JudgeBiasBench, edits are selected separately for its seven subsets, allowing different bias categories to use different edits. Unless otherwise stated, selection uses the full response pool available for each benchmark.

This is a transductive setting: benchmark responses are available during edit selection, but their preferred/rejected information remains hidden. After edit selection, we apply the selected attribute-specific edit or edits to the linear reward head. The resulting reward head is then fixed and evaluated on the benchmark. All selection settings are shared across benchmarks, and results are averaged over seeds 0 and 1 where applicable.

A.5 Response Attributes

BiasReducer considers seven predefined response attributes motivated by prior work showing that reward models and LLM judges can be influenced by response length, formatting, confidence, agreement, and related surface-level behaviors (Liu et al., 2025b; Bharadwaj et al., 2026; Zhang et al., 2025). We group these recurring behaviors into seven attributes: length, verbosity, formatting, emoji/exclamation use, confidence, hedging, and sycophancy. Each attribute is associated with a measurable text signal, and most also use controlled rewrites that increase or decrease the attribute. Table 9 summarizes the meaning of each attribute and the corresponding type of bias it is intended to capture. Detailed behavioral measurements and controlled rewrites are provided in Appendix B.2.

A.6 Baseline Implementations

We compare BiasReducer with two training-based baselines that further fine-tune the released reward-model checkpoints.

RRM-style fine-tuning. RRM (Liu et al., 2025a) is a training-based approach for reducing reward-model bias through augmented preference data. Because our experiments begin from already trained reward-model checkpoints, we adapt the RRM procedure to further fine-tune each released reward model rather than retraining a reward model from its underlying base language model. Following

Table 9: **Response attributes considered by BIASREDUCER.**

Attribute	What it captures
Length	Preference for longer or shorter responses independent of response quality.
Verbosity	Preference for unnecessarily detailed or repetitive responses.
Formatting	Preference driven by presentation choices such as Markdown structure or styling.
Emoji / exclamation	Preference driven by decorative or emphatic markers such as emojis and exclamation marks.
Confidence	Preference for more certain or assertive language, even when confidence does not reflect correctness.
Hedging	Preference related to cautious or uncertain language, independent of the underlying answer quality.
Sycophancy	Preference for responses that agree with or defer to the user rather than responding independently.

RRM (Liu et al., 2025a), we use only the source data to construct additional preferred–rejected response pairs and mix them with the original source preferences to further fine-tune the released reward model.

Intervention fine-tuning. The intervention fine-tuning baseline uses attribute-specific counterfactual pairs constructed around the same named response attributes used by BIASREDUCER. The counterfactual pairs are mixed with original source preferences and used to fine-tune the released reward model. This baseline tests whether attribute-specific supervision is better absorbed through additional training or through reusable reward-head edits.

Table 10 reports the detailed performance per reward model on each benchmark.

Table 10: **Detailed reward-model bias-mitigation results.** RRM denotes RRM-based fine-tuning of the released reward model (Liu et al., 2025a); Intervention FT denotes fine-tuning the same reward model on constructed attribute-intervention pairs mixed with the original source preference data. We report pairwise preference accuracy (%) for the Original model, RRM, Intervention FT, BIASREDUCER-S, and BIASREDUCER-M. RM-Bench-Hard is reported overall and by domain (Cht = chat, Cde = code, Mth = math, Sft = safety). Arena denotes Arena-StyleConflict and reports overall accuracy. JBB denotes JudgeBiasBench and reports the weighted average over its subsets. Bold marks the best non-Original result in each column; blue rows denote our methods. Higher is better.

	RM-Bench-Hard ↑					Arena ↑	JBB ↑
Model	**Avg**	**Cht**	**Cde**	**Mth**	**Sft**	**Avg**	**Avg**
Qwen3-1.7B							
+ Original	59.96	31.52	55.26	48.96	83.90	22.90	73.93
+ RRM	65.29	38.76	58.48	55.77	87.98	30.20	**77.28**
+ Intervention FT	61.42	35.66	54.53	50.22	85.94	26.50	75.99
+ **BIASREDUCER-S**	64.38	39.41	59.65	53.43	87.27	39.25	75.21
+ **BIASREDUCER-M**	**66.92**	**44.44**	**62.28**	**56.24**	**88.70**	**39.65**	75.64
GRM-3B							
+ Original	48.93	32.56	25.58	25.65	93.72	36.20	70.10
+ RRM	52.50	42.89	29.68	**30.56**	93.43	40.30	76.37
+ Intervention FT	50.01	35.66	26.17	26.97	**94.18**	38.90	72.29
+ **BIASREDUCER-S**	52.98	53.23	36.04	27.35	92.40	42.15	76.74
+ **BIASREDUCER-M**	**54.08**	**55.94**	**36.11**	28.58	93.43	**43.55**	**78.93**
Mistral-7B							
+ Original	44.64	16.54	23.98	31.63	79.14	20.30	63.19
+ RRM	49.08	21.19	25.88	39.51	80.73	27.50	63.53
+ Intervention FT	51.70	24.55	27.49	41.90	83.90	35.30	68.69
+ **BIASREDUCER-S**	53.47	28.17	27.19	41.71	88.55	51.65	75.02
+ **BIASREDUCER-M**	**55.38**	**33.98**	**28.29**	**42.56**	**91.01**	**56.05**	**76.55**
InternLM2-7B							
+ Original	58.25	19.12	24.85	66.79	76.72	50.70	57.82
+ RRM	60.84	23.26	25.58	**67.42**	**82.16**	48.90	62.33
+ Intervention FT	57.42	20.67	23.10	63.83	78.23	48.20	61.90
+ **BIASREDUCER-S**	62.65	35.01	34.65	65.94	81.26	67.00	63.45
+ **BIASREDUCER-M**	**63.93**	**39.92**	**38.01**	65.94	81.93	**69.55**	**63.68**
GRM-8B							
+ Original	50.89	30.49	32.31	31.38	89.87	34.10	70.06
+ RRM	56.02	40.57	38.74	37.11	92.14	41.20	**78.39**
+ Intervention FT	52.10	32.04	34.21	32.14	91.16	36.30	73.71
+ **BIASREDUCER-S**	60.11	43.80	48.10	41.65	93.24	42.50	74.79
+ **BIASREDUCER-M**	**63.90**	**49.48**	**53.80**	**46.75**	**93.91**	**45.15**	74.66
Mean (five reward models)							
+ Original	52.53	26.05	32.40	40.88	84.67	32.84	67.02
+ RRM	56.75	33.33	35.67	46.07	87.29	37.62	71.58
+ Intervention FT	54.53	29.72	33.10	43.01	86.68	37.04	70.52
+ **BIASREDUCER-S**	58.72	39.92	41.13	46.02	88.54	48.51	73.04
+ **BIASREDUCER-M**	**60.84**	**44.75**	**43.70**	**48.01**	**89.80**	**50.79**	**73.89**

B METHOD DETAILS

This section provides additional implementation and mathematical details for BIASREDUCER, including semantic supervision, representation learning, determining how to edit the reward head for each attribute, selecting edits for a new dataset, and multi-edit composition.

B.1 PROBLEM SETUP AND NOTATION

We consider a scalar reward model $r(x) = w_r^\top h(x) + b$, where $h(x) \in \mathbb{R}^d$ is the response representation used as input to the linear reward head, w_r is the reward-head weight, and b is the constant term. BIASREDUCER modifies w_r to obtain an edited reward head w_r', designed to reduce the reward model's dependence on selected response attributes.

We denote the source preference data by $\mathcal{D}_{\text{src}}$, with training and validation splits $\mathcal{D}_{\text{src}}^{\text{train}}$ and $\mathcal{D}_{\text{src}}^{\text{val}}$. For a source preference pair (x^+, x^-), where x^+ is preferred over x^-, we define the preference contrast $\Delta h = h(x^+) - h(x^-)$. The predefined response attributes are denoted by $\mathcal{A}$, with $a \in \mathcal{A}$ denoting an individual attribute.

For a new dataset $\mathcal{D}$, BIASREDUCER may use its responses, hidden representations, and original reward scores, but does not know which response in each pair is preferred.

B.2 SEMANTIC SUPERVISION FOR RESPONSE ATTRIBUTES

For each predefined response attribute $a \in \mathcal{A}$, we assign a designated coordinate z_a and provide supervision that gives this coordinate a clear, human-interpretable meaning. We use two complementary sources of supervision: behavioral measurements from naturally occurring response pairs and controlled rewrites that explicitly change an attribute. The seven attributes are summarized in Table 9.

Behavioral supervision. Let $y_a(x)$ denote a deterministic text-based measurement of attribute a. For a source preference pair (x^+, x^-), we compute the standardized difference

$$\Delta y_a = \frac{y_a(x^+) - y_a(x^-) - \mu_a}{\sigma_a}, \tag{6}$$

where μ_a and σ_a are fixed normalization statistics estimated from the source data. Positive values indicate that the attribute is more prominent in the preferred response, while negative values indicate that it is more prominent in the rejected response. These measurements tell each designated coordinate which response attribute it should track.

Controlled interventions. Behavioral measurements capture naturally occurring differences between responses; controlled rewrites additionally show how the coordinate should change when the attribute is deliberately increased or decreased, establishing the direction associated with that attribute. All controlled rewrites use deterministic, rule-based text transformations, such as template insertion, fixed-lexicon substitution, and paragraph truncation, applied to source training responses. No LLM is used to generate the rewrites.

For an original response x_{orig} and a rewritten response x_{edit}, we compute

$$\delta = h(x_{\text{edit}}) - h(x_{\text{orig}}), \tag{7}$$

with $s \in \{-1, +1\}$ indicating whether the rewrite decreases or increases the target attribute. The intervention construction is attribute-specific (see Table 11).

Forward and reverse intervention differences are also pooled to capture changes that are shared across the rewriting process rather than specific to one attribute.

Separating rewrite artifacts. Controlled rewrites can introduce systematic changes unrelated to the intended attribute. We therefore reserve one additional artifact coordinate to capture variation shared across different rewrites. This coordinate is learned jointly with the named attribute coordinates but is not used to determine or apply reward-head edits.

Table 11: **Controlled interventions used for semantic supervision.**

Attribute	Intervention construction
Length	Expansion / contraction
Verbosity	Attribute rewrite minus matched-control rewrite
Formatting	Forward / reverse formatting rewrite
Emoji / exclamation	Attribute rewrite minus matched-control rewrite
Confidence	Forward / reverse confidence rewrite
Hedging	Behavioral supervision only
Sycophancy	Artifact-adjusted forward / reverse rewrite

B.3 Learning Attribute-Aligned Representations

During this stage, we first extract preference contrasts $\Delta h = h(x^+) - h(x^-)$ from the reward model and use them to train the encoder–decoder.

$$u = \Delta h = h(x^+) - h(x^-) \tag{8}$$

The encoder computes

$$z_{\text{pre}} = W_{\text{enc}} u + b_{\text{enc}}, \tag{9}$$

followed by signed top-k_{act} sparsification,

$$z_{\text{post}} = \text{TopK}(z_{\text{pre}}), \tag{10}$$

which retains the k_{act} coordinates with the largest absolute activations while preserving their signs. The decoder reconstructs the preference contrast as

$$\hat{u} = V z_{\text{post}}. \tag{11}$$

We designate one coordinate for each predefined response attribute and one additional coordinate for rewrite artifacts. The remaining coordinates are used only for reconstruction and are not assigned predefined meanings.

Representation objective. The base objective encourages accurate reconstruction while regularizing the latent representation:

$$\mathcal{L}_{\text{rep}} = \mathbb{E}\left[\|u - V z_{\text{post}}\|_2^2\right] + \lambda_{\text{sp}} \mathbb{E}\left[\frac{1}{K}\|z_{\text{post}}\|_1\right] + \beta \, \text{incoh}(V), \tag{12}$$

where $\text{incoh}(V) = \|\hat{V}^\top \hat{V} - I\|_F^2 / [K(K-1)]$, with $\hat{V}$ denoting the column-normalized decoder matrix. The three terms encourage reconstruction, activation sparsity, and distinct decoder directions, respectively.

Behavioral supervision. For each attribute a, we measure how strongly every latent coordinate correlates with the standardized behavioral signal:

$$c_k^{(a)} = \text{corr}_{\text{batch}}\left(z_{\text{pre},k}(u), \Delta y_a\right). \tag{13}$$

We encourage the designated coordinate z_a to track attribute a more strongly than competing coordinates:

$$\mathcal{L}_{\text{axis}}(a) = \text{ReLU}\left(m_{\text{corr}} - c_a^{(a)} + \max_{k \notin \{a, \text{art}\}} |c_k^{(a)}|\right). \tag{14}$$

To reduce associations between one attribute and other named coordinates, we also penalize cross-attribute leakage:

$$\mathcal{L}_{\text{dec}}(a) = \sum_{\substack{b \in \mathcal{A} \\ b \neq a}} \left(c_b^{(a)}\right)^2. \tag{15}$$

The full behavioral-supervision loss is

$$\mathcal{L}_{\text{beh}} = \sum_{a \in \mathcal{A}} \left[\mathcal{L}_{\text{axis}}(a) + \kappa_{\text{dec}} \mathcal{L}_{\text{dec}}(a)\right]. \tag{16}$$

Intervention supervision. For intervention example i with target attribute a_i, hidden-state difference δ_i, and expected direction s_i, we use

$$\mathcal{L}_{\text{int}} = \mathbb{E}_i \,\text{ReLU}\left(m_{\text{conc}} - s_i z_{\text{pre},a_i}(\delta_i) + \max_{k \notin \{a_i,\text{art}\}} |z_{\text{post},k}(\delta_i)|\right). \tag{17}$$

This encourages the designated coordinate to change in the expected direction when the corresponding attribute is increased or decreased, while responding more strongly than competing coordinates.

Artifact supervision. Let δ range over pooled controlled-rewrite differences. We encourage the artifact coordinate to capture changes shared across the rewriting process:

$$\mathcal{L}_{\text{art}} = \mathbb{E}_\delta \,\text{ReLU}\left(m_{\text{conc}} - \left(-z_{\text{pre,art}}(\delta)\right) + \max_{k \neq \text{art}} |z_{\text{post},k}(\delta)|\right). \tag{18}$$

Joint objective. Because the auxiliary losses can have different numerical scales, we rescale each relative to the representation loss:

$$\text{scaled}(\mathcal{L}_x) = \frac{\text{sg}(\mathcal{L}_{\text{rep}})}{\text{sg}(\mathcal{L}_x) + \varepsilon_{\text{loss}}} \mathcal{L}_x, \tag{19}$$

where sg denotes stop-gradient. The final training objective is

$$\mathcal{L} = \mathcal{L}_{\text{rep}} + \gamma_b \,\text{scaled}(\mathcal{L}_{\text{beh}}) + \gamma_i \,\text{scaled}(\mathcal{L}_{\text{int}}) + \gamma_a \,\text{scaled}(\mathcal{L}_{\text{art}}). \tag{20}$$

The reward model remains unchanged throughout this training stage.

B.4 Semantic Checks After Representation Learning

After representation learning, we verify on the validation set of the source data $\mathcal{D}^{\text{val}}_{\text{src}}$ that each designated coordinate behaves consistently with its intended response attribute. These checks are used only to evaluate the learned representations and do not update model parameters or affect edit selection.

We use several complementary checks. We compare coordinate behavior under naturally occurring attribute differences with behavior under controlled rewrites, and test forward–reverse consistency, matched controls, split-half stability, and whether the coordinate changes in the expected direction. For response length, we additionally check whether lengthening and shortening produce approximately symmetric changes.

To compare natural attribute variation with controlled rewrites, we estimate

$$n_a = \mathbb{E}_{(x^+,x^-)\sim\mathcal{D}^{\text{val}}_{\text{src}}} \left[\text{sign}\left(y_a(x^+) - y_a(x^-)\right) \Delta h\right], \tag{21}$$

where n_a summarizes the hidden-state direction associated with an increase in attribute a under naturally occurring response differences. We compare this direction with the corresponding direction from controlled rewrites. Consistency between the natural and rewrite-based directions provides additional evidence that the learned coordinate captures the intended attribute.

B.5 Determining Edit Direction and Amount

After learning the attribute representations, we determine two quantities for each attribute: the direction in which to adjust the reward head and how far to adjust it. Both are determined from source data before edit selection on a new dataset.

Edit direction. During representation learning, behavioral supervision uses the continuous pre-top-k_{act} activations. To determine the edit direction, we instead apply the full encoder and top-k_{act} operator to individual source-response representations:

$$z_{\text{post}}(h) = \text{TopK}\left(W_{\text{enc}} h + b_{\text{enc}}\right). \tag{22}$$

For each attribute a, we estimate

$$C_a = \text{Cov}_{h\sim\mathcal{D}^{\text{train}}_{\text{src}}}\left(h, z_{\text{post},a}(h)\right), \tag{23}$$

where the covariance is centered in both arguments. C_a captures the direction in the reward-model representation that co-varies with attribute a on natural source responses.

We define the corresponding reward-head update as

$$w_a(\lambda) = w_r - \lambda \frac{w_r^\top C_a}{C_a^\top C_a + \epsilon_{\text{edit}}} C_a, \tag{24}$$

where λ controls how far the reward head is adjusted along C_a.

Relation to linear concept erasure. Ignoring the numerical stabilizer, setting $\lambda = 1$ gives

$$w_a(1)^\top C_a = 0, \tag{25}$$

and therefore

$$\text{Cov}\left(w_a(1)^\top h, z_{\text{post},a}(h)\right) = 0. \tag{26}$$

Thus, at $\lambda = 1$, the edited reward is linearly decorrelated from the attribute activation along C_a. This is closely related to the rank-one scalar-concept projection used in linear concept erasure (Belrose et al., 2023). Unlike standard LEACE, we apply the update directly to the reward head and do not whiten the hidden representation. For $0 < \lambda < 1$, the update moves partway toward the decorrelation point, while $\lambda > 1$ moves beyond it.

Choosing how far to edit. For each attribute, we evaluate a fixed set Λ of candidate values for λ. We first measure how much each candidate changes reward scores on $\mathcal{D}_{\text{src}}^{\text{train}}$:

$$\text{Agg}_a(\lambda) = \mathbb{E}_{h \sim \mathcal{D}_{\text{src}}^{\text{train}}} \left| h^\top w_a(\lambda) - h^\top w_r \right|, \tag{27}$$

where $\text{Agg}_a(\lambda)$ is the average reward-score change induced by the edit.

We separately measure the drop in preference accuracy on $\mathcal{D}_{\text{src}}^{\text{val}}$:

$$\Delta\text{Acc}_{\text{src}}\big(w_a(\lambda)\big) = \text{Acc}_{\text{src}}(w_r) - \text{Acc}_{\text{src}}\big(w_a(\lambda)\big), \tag{28}$$

where Acc_{src} is evaluated on $\mathcal{D}_{\text{src}}^{\text{val}}$.

We then select

$$\lambda_a^\star = \arg\max_{\lambda \in \Lambda} \text{Agg}_a(\lambda) \quad \text{s.t.} \quad \Delta\text{Acc}_{\text{src}}\big(w_a(\lambda)\big) \le \tau_{\text{src}}, \tag{29}$$

with $\tau_{\text{src}} = 0.02$. Thus, $\lambda_a^\star$ favors larger reward-score changes on the source training data while limiting the loss in preference accuracy on held-out source validation data. If no nonzero value satisfies the constraint, attribute a is marked ineligible.

For each eligible attribute, $(C_a, \lambda_a^\star)$ specifies the direction and amount by which the reward head should be adjusted. We collect these values into the fixed edit bank

$$\mathcal{B} = \{(a, C_a, \lambda_a^\star) \mid a \in \mathcal{A} \text{ is eligible}\}. \tag{30}$$

B.6 Dataset-Specific Edit Selection and Multi-Edit Composition

The edit bank $\mathcal{B}$ is fixed before a new dataset is considered. For a new dataset $\mathcal{D}$, BiasReducer decides which attribute-specific edits to use and, for BiasReducer-M, how to combine them. This selection uses the responses, their hidden representations, and the original reward scores, without knowing which response in each pair is preferred.

Edit-selection signals. For each eligible attribute a, we compute two signals. The first measures how strongly the original reward varies with the learned attribute:

$$G_a(\mathcal{D}) = \left| \text{Cov}_{h \sim \mathcal{D}} \left(w_r^\top h, z_{\text{pre},a}(h) \right) \right|, \tag{31}$$

where $z_{\text{pre},a}(h)$ is the continuous pre-top-k_{act} activation of attribute a.

The second measures how much the corresponding attribute-specific edit changes the reward scores:

$$I_a(\mathcal{D}) = \mathbb{E}_{h \sim \mathcal{D}} \left| h^\top w_a(\lambda_a^\star) - h^\top w_r \right|. \tag{32}$$

Because $G_a(\mathcal{D})$ and $I_a(\mathcal{D})$ have different numerical scales, we rank eligible attributes separately in descending order under the two signals. We combine the rankings using their rank sum:

$$B_a(\mathcal{D}) = \text{rank}_{\downarrow G}(a) + \text{rank}_{\downarrow I}(a). \tag{33}$$

A smaller $B_a(\mathcal{D})$ gives attribute a higher selection priority. Ties are resolved deterministically using a fixed attribute order.

BIASREDUCER-S. Let a_1 denote the highest-ranked eligible attribute. The single-edit variant uses its source-determined direction and edit amount:

$$w_{\mathrm{S}}^{\star} = w_{a_1}(\lambda_{a_1}^{\star}). \tag{34}$$

BIASREDUCER-M and multi-edit composition. The multi-edit variant follows the same dataset-specific ranking and considers attributes sequentially. Starting from

$$w^{(0)} = w_r, \tag{35}$$

the candidate update for attribute a_j using value λ_j is

$$\widetilde{w}^{(j)} = w^{(j-1)} - \lambda_j \frac{(w^{(j-1)})^{\top} C_{a_j}}{C_{a_j}^{\top} C_{a_j} + \epsilon_{\mathrm{edit}}} C_{a_j}. \tag{36}$$

Each candidate initially uses its source-selected value $\lambda_{a_j}^{\star}$. If the cumulative edit exceeds the source preference-accuracy tolerance, we try smaller values of λ_j on the same candidate grid Λ. If no value is admissible, we allow up to two limited adjustment steps: the largest currently selected λ is reduced by one grid level, the cumulative reward head is recomputed, and the new candidate is reconsidered. If no admissible composition is found, the candidate is skipped.

The source-preservation constraint is evaluated on the complete cumulative reward head:

$$\Delta\mathrm{Acc}_{\mathrm{src}}\left(w^{(j)}\right) \leq \tau_{\mathrm{src}}, \tag{37}$$

where the accuracy is measured on $\mathcal{D}_{\mathrm{src}}^{\mathrm{val}}$. Composition stops after $k_{\max}$ edits have been accepted or all eligible attributes have been considered. We use $k_{\max} = 4$ by default.

Unless otherwise stated, edit selection uses the full response pool available for each benchmark. RM-Bench-Hard and Arena-StyleConflict each use one edit-selection decision, while the seven Judge-BiasBench subsets are handled separately. The full evaluation protocol is described in Appendix A.4.

B.7 COMPLETE ALGORITHM

Algorithm 2 Complete BIASREDUCER procedure

Require: reward model (h, w_r); source training set $\mathcal{D}_{\text{src}}^{\text{train}}$; source validation set $\mathcal{D}_{\text{src}}^{\text{val}}$; attributes $\mathcal{A}$; candidate values Λ; source accuracy budget τ_{src}; new dataset $\mathcal{D}$; maximum number of edits $k_{\max}$
Ensure: edited reward heads $w_{\text{S}}^{\star}$ and $w_{\text{M}}^{\star}$

Stage 1: Learn representations of response attributes
Train the semantically supervised encoder–decoder on source preference contrasts Δh

Stage 2: Determine edit direction and amount
$\mathcal{B} \leftarrow \emptyset$
for $a \in \mathcal{A}$ **do**
    Compute $z_{\text{post},a}(h)$ and estimate C_a on $\mathcal{D}_{\text{src}}^{\text{train}}$
    for $\lambda \in \Lambda$ **do**
        Compute $w_a(\lambda)$, $\text{Agg}_a(\lambda)$, and the validation preference-accuracy drop
    end for
    Keep values satisfying $\Delta\text{Acc}_{\text{src}}(w_a(\lambda)) \leq \tau_{\text{src}}$
    if at least one nonzero value is admissible **then**
        $\lambda_a^{\star} \leftarrow \arg\max_{\lambda} \text{Agg}_a(\lambda)$ over admissible values
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{(a, C_a, \lambda_a^{\star})\}$
    end if
end for

Stage 3: Select edits for the new dataset
for $(a, C_a, \lambda_a^{\star}) \in \mathcal{B}$ **do**
    Compute $G_a(\mathcal{D})$, $I_a(\mathcal{D})$, and $B_a(\mathcal{D}) = \text{rank}_{\downarrow G}(a) + \text{rank}_{\downarrow I}(a)$
end for
Sort eligible attributes by increasing $B_a(\mathcal{D})$ to obtain $(a_1, \ldots, a_{|\mathcal{B}|})$

Stage 4: Edit the reward head
BIASREDUCER-S: $w_{\text{S}}^{\star} \leftarrow w_{a_1}(\lambda_{a_1}^{\star})$
BIASREDUCER-M: $w_{\text{M}}^{\star} \leftarrow w_r$
for a_j in selection order **do**
    Start from $\lambda_j \leftarrow \lambda_{a_j}^{\star}$ and try smaller values until the source accuracy constraint is satisfied
    if no value is admissible **then**
        Allow up to two adjustment steps by reducing the largest previously selected λ
    end if
    Accept the edit if an admissible cumulative head is found; otherwise skip it
    if $k_{\max}$ edits have been accepted **then**
        break
    end if
end for
return $w_{\text{S}}^{\star}, w_{\text{M}}^{\star}$

C Additional Analysis and Ablations

This section provides extended results and ablations supporting the findings in Section 5.

C.1 Semantic Supervision

We further examine how semantic supervision affects both the learned representations and downstream editing. Table 12 isolates the effect of train-time semantic supervision while keeping the downstream editing procedure unchanged. We compare BiasReducer with a vanilla SAE trained on the same preference pairs but without attribute-specific supervision. For the vanilla SAE, after training, each response attribute is assigned to the latent coordinate most correlated with its measured difference across response pairs. This comparison tests whether assigning attribute meanings during representation learning provides an advantage over identifying relevant coordinates only after training. Table 13 further removes the behavioral, intervention, and artifact supervision terms individually. We report both intrinsic representation metrics and the resulting editing performance to examine the contribution of each supervision signal.

Table 12: **Effect of train-time semantic supervision.** We compare BiasReducer with a vanilla SAE trained on the same preference contrasts and with the same dictionary settings, but without semantic supervision. For the vanilla SAE, each attribute is assigned after training to the pre-top-k_{act} latent coordinate with the highest absolute correlation to its measured attribute signal. All downstream editing and edit-selection settings are held fixed. $|\text{corr}|$ is the mean held-out absolute correlation across the seven attributes. The remaining columns report changes in pairwise preference accuracy (percentage points relative to the original reward model) on RM-Bench-Hard (RMB), Arena-StyleConflict (Arena), and JudgeBiasBench (JBB). Higher is better.

		BiasReducer-S			BiasReducer-M		
Representation strategy	**|corr|**	**RMB**	**Arena**	**JBB**	**RMB**	**Arena**	**JBB**
Qwen3-1.7B							
BiasReducer (semantic supervision)	0.52	+4.42	+16.35	+1.29	+6.96	+16.75	+1.72
Vanilla SAE + single-latent match	0.26	−0.64	+7.90	−2.49	+1.87	+8.45	+0.04
GRM-3B							
BiasReducer (semantic supervision)	0.53	+4.04	+5.95	+6.64	+5.15	+7.35	+8.83
Vanilla SAE + single-latent match	0.29	+9.12	+10.50	+4.62	+8.52	+5.05	+3.05
Mistral-7B							
BiasReducer (semantic supervision)	0.58	+8.83	+31.35	+11.83	+10.74	+35.75	+13.36
Vanilla SAE + single-latent match	0.24	+5.79	+11.00	+2.68	+5.55	+18.40	+2.19
InternLM2-7B							
BiasReducer (semantic supervision)	0.55	+4.40	+16.30	+5.63	+5.68	+18.85	+5.86
Vanilla SAE + single-latent match	0.23	+2.78	+2.80	−0.73	+3.00	+1.85	−0.75
GRM-8B							
BiasReducer (semantic supervision)	0.57	+9.22	+8.40	+4.73	+13.01	+11.05	+4.60
Vanilla SAE + single-latent match	0.27	+4.73	+2.30	+1.18	+6.02	+3.30	+0.90
Mean (five RMs above)							
BiasReducer (semantic supervision)	**0.55**	+6.18	+15.67	+6.02	+8.31	+17.95	+6.87
Vanilla SAE + single-latent match	0.26	+4.36	+6.90	+1.05	+4.99	+7.41	+1.09

C.2 Edit Direction and Amount

We further examine how a learned response attribute should be translated into an effective reward-head edit. The analyses separate three design choices: which direction to edit, how far to edit along that direction, and which reward-model representation to use when estimating these quantities.

Table 14 compares our covariance-based edit direction with the SAE decoder vector associated with the same attribute coordinate. This tests whether identifying an interpretable attribute is sufficient, or whether the edit direction should also reflect how that attribute relates to the reward-model representation.

Table 13: **Ablation of semantic supervision components.** Attr. corr. measures how strongly each designated coordinate tracks its intended attribute, Leakage measures unintended associations with other named attributes, and Sign acc. measures whether controlled rewrites move the coordinate in the expected direction. The remaining columns report changes in pairwise preference accuracy (percentage points relative to the original reward model) on RM-Bench-Hard (RMB), Arena-StyleConflict (Arena), and JudgeBiasBench (JBB). Representation metrics are averaged over seeds; arrows indicate preferred directions.

	Representation metrics			BIASREDUCER-S (Δ)			BIASREDUCER-M (Δ)		
Variant	**Axis corr. ↑**	**Leakage ↓**	**Sign acc. ↑**	**RMB**	**Arena**	**JBB**	**RMB**	**Arena**	**JBB**
Qwen3-1.7B									
Full supervision	0.52	0.08	0.99	+4.42	+16.35	+1.29	+6.96	+16.75	+1.72
w/o L_{beh}	0.18	0.33	1.00	+6.08	+1.25	+3.48	+10.18	+4.70	+4.32
w/o L_{int}	0.54	0.09	0.74	+2.82	+4.25	+0.69	+4.46	+7.45	+1.31
w/o L_{art}	0.53	0.09	1.00	+4.51	+11.65	+1.57	+5.67	+10.20	+3.33
GRM-3B									
Full supervision	0.53	0.07	1.00	+4.04	+5.95	+6.64	+5.15	+7.35	+8.83
w/o L_{beh}	0.16	0.36	1.00	+4.89	+11.65	−0.32	+5.95	+11.45	+0.67
w/o L_{int}	0.49	0.08	0.75	+3.81	+7.40	+5.18	+5.36	+11.95	+7.32
w/o L_{art}	0.54	0.08	1.00	+4.50	+9.90	+9.49	+6.49	+10.40	+10.52
Mistral-7B									
Full supervision	0.58	0.07	1.00	+8.83	+31.35	+11.83	+10.74	+35.75	+13.36
w/o L_{beh}	0.20	0.24	1.00	+2.12	+33.35	+0.37	+8.22	+35.55	+4.94
w/o L_{int}	0.58	0.06	0.80	+8.93	+40.00	+14.71	+9.23	+39.20	+12.65
w/o L_{art}	0.56	0.08	1.00	+10.02	+37.65	+14.02	+9.72	+39.20	+15.01
InternLM2-7B									
Full supervision	0.55	0.08	0.99	+4.40	+16.30	+5.63	+5.68	+18.85	+5.86
w/o L_{beh}	0.13	0.30	0.99	+3.01	+7.07	+2.06	+5.15	+11.13	+2.81
w/o L_{int}	0.55	0.10	0.75	+1.84	+8.93	+4.08	+5.36	+7.63	+5.28
w/o L_{art}	0.55	0.08	1.00	+2.87	+7.27	+1.61	+2.55	+9.42	+0.41
GRM-8B									
Full supervision	0.57	0.08	0.99	+9.22	+8.40	+4.73	+13.01	+11.05	+4.60
w/o L_{beh}	0.16	0.32	1.00	+4.41	+3.75	+1.37	+6.10	+5.65	+1.10
w/o L_{int}	0.51	0.09	0.72	+5.19	+6.25	+2.43	+8.54	+7.45	+2.26
w/o L_{art}	0.56	0.12	1.00	+5.07	+7.25	+0.90	+8.97	+7.05	+1.42

Table 15 compares three choices for how far to edit: fixed values of λ chosen in advance, one value selected on source validation data and shared across attributes, and a separate $\lambda_a^\star$ for each attribute. This comparison tests whether different attributes benefit from different edit amounts.

Finally, Table 16 compares using natural response representations h and preference contrasts Δh when estimating edit directions and computing the signals used for edit selection.

C.3 DATASET-SPECIFIC EDIT SELECTION

We further examine how edit selection varies across reward models and benchmarks, whether selecting edits separately for each dataset improves over using the same attribute everywhere, and how reliably the selection can be made from a limited number of responses.

Figure 5 compares the attribute rankings across all five reward models and three benchmarks. The remaining analyses examine representative edit selections, compare dataset-specific and fixed attribute choices, ablate the two selection signals, and vary the number of responses available for edit selection.

Representative edit selections. Table 17 shows three edit-selection examples for the same Qwen3-1.7B reward model. RM-Bench-Hard ranks sycophancy highest, Arena-StyleConflict ranks length highest, and the JudgeBiasBench sycophancy subset again ranks sycophancy highest. These examples illustrate that the attributes most relevant to reward scores can differ across datasets. BIASREDUCER-M can additionally combine edits for multiple highly ranked attributes.

Table 14: **Effect of edit direction.** We compare BIASREDUCER's covariance-based direction with the decoder vector associated with the same attribute coordinate, i.e., the direction used by the SAE decoder to reconstruct that coordinate in the reward-model representation space. The selected attribute is kept fixed. Entries report changes in pairwise preference accuracy (percentage points relative to the original reward model) on RM-Bench-Hard (RMB), Arena-StyleConflict (Arena), and JudgeBiasBench (JBB). Higher is better.

Edit direction	RMB	Arena	JBB
Qwen3-1.7B			
Covariance direction	+4.42	+16.35	+1.29
Decoder direction	+2.27	+1.70	+1.93
GRM-3B			
Covariance direction	+4.04	+5.95	+6.64
Decoder direction	+0.15	+0.20	+0.79
Mistral-7B			
Covariance direction	+8.83	+31.35	+11.83
Decoder direction	+0.06	+0.15	+0.30
InternLM2-7B			
Covariance direction	+4.40	+16.30	+5.63
Decoder direction	−0.01	−0.05	+0.04
GRM-8B			
Covariance direction	+9.22	+8.40	+4.73
Decoder direction	+0.21	+0.00	−0.13

Dataset-specific versus fixed attribute selection. Table 18 compares BIASREDUCER with two alternatives that use the same attribute across all datasets: sycophancy, and the attribute whose edit performs best on source validation data. This comparison isolates the benefit of selecting attributes separately for each dataset while keeping the available attribute-specific edits fixed.

Edit-selection signal ablation. Table 19 compares reward dependence, edit impact, and their rank-sum combination. Oracle coverage measures how often the selected set contains the candidate edit that performs best on the benchmark. Oracle evaluation uses benchmark preference labels; edit selection does not.

Sensitivity to response-pool size. Table 20 recomputes edit selection using progressively larger subsets of JudgeBiasBench responses. Agreement is measured against the selection obtained from the full response pool, while the gain columns report the resulting BIASREDUCER-S and BIASREDUCER-M changes in pairwise preference accuracy.

C.4 SENSITIVITY TO EDITING AND REPRESENTATION SETTINGS

We vary the main editing and representation settings to test whether the results depend on a narrow choice of hyperparameters.

Number of combined edits. Table 21 varies the maximum number of edits that BIASREDUCER-M can combine, $k_{\max}$. This tests how sensitive performance is to the allowed complexity of the final reward-head edit.

Representation settings. Table 22 varies the dictionary size and the number of active coordinates. These experiments examine how the representation capacity and sparsity level affect downstream editing performance.

Table 15: **Effect of how the edit amount is chosen for BIASREDUCER-S.** We vary only λ while holding all other settings fixed. The first two variants use fixed values chosen in advance. The shared variants use one λ for all attributes, either $\min_a \lambda_a^\star$ or a shared value selected using the mean source-validation accuracy change. Our method selects a separate $\lambda_a^\star$ for each attribute. Benchmark columns report changes in pairwise preference accuracy (pp relative to the original reward model). Higher is better.

Edit amount	RMB	Arena	JBB
Qwen3-1.7B			
λ=1 (uncalibrated)	+4.42	+4.00	+1.29
λ=0.5 (uncalibrated)	+1.27	+1.55	+0.47
global $\lambda = \min_a \lambda_a^\star$	+3.20	+3.25	+0.86
global λ (mean drop)	+4.42	+4.00	+1.29
per-attribute $\lambda_a^\star$ (ours)	+4.42	+16.35	+1.29
GRM-3B			
λ=1 (uncalibrated)	+4.04	+11.20	+6.64
λ=0.5 (uncalibrated)	+0.46	+1.35	+2.60
global $\lambda = \min_a \lambda_a^\star$	+2.46	+5.95	+5.82
global λ (mean drop)	+4.04	+11.20	+6.64
per-attribute $\lambda_a^\star$ (ours)	+4.04	+5.95	+6.64
Mistral-7B			
λ=1 (uncalibrated)	+6.30	+21.50	+8.42
λ=0.5 (uncalibrated)	+2.55	+7.35	+3.35
global $\lambda = \min_a \lambda_a^\star$	+5.45	+16.95	+7.02
global λ (mean drop)	+6.30	+21.50	+8.42
per-attribute $\lambda_a^\star$ (ours)	+8.83	+31.35	+11.83
InternLM2-7B			
λ=1 (uncalibrated)	+0.84	+4.30	+1.20
λ=0.5 (uncalibrated)	+0.41	+1.95	+0.45
global $\lambda = \min_a \lambda_a^\star$	+0.79	+3.25	+0.95
global λ (mean drop)	+0.79	+3.25	+0.95
per-attribute $\lambda_a^\star$ (ours)	+4.40	+16.30	+5.63
GRM-8B			
λ=1 (uncalibrated)	+22.78	+22.00	+2.96
λ=0.5 (uncalibrated)	+1.83	+1.35	+1.16
global $\lambda = \min_a \lambda_a^\star$	+4.21	+3.80	+2.53
global λ (mean drop)	+9.22	+8.40	+3.59
per-attribute $\lambda_a^\star$ (ours)	+9.22	+8.40	+4.73
Mean (five RMs above)			
λ=1 (uncalibrated)	+7.68	+12.60	+4.10
λ=0.5 (uncalibrated)	+1.30	+2.71	+1.61
global $\lambda = \min_a \lambda_a^\star$	+3.22	+6.64	+3.44
global λ (mean drop)	+4.95	+9.67	+4.18
per-attribute $\lambda_a^\star$ (ours)	+6.18	+15.67	+6.02

Table 16: **Natural response representations vs. pairwise contrasts for edit estimation and edit selection.** Row labels indicate the representation used for *edit estimation* / *edit selection*. Edit estimation determines the direction C_a and edit amount $\lambda_a^\star$; edit selection computes the dataset-specific signals used to rank attributes. The two choices are varied independently, with $\lambda_a^\star$ re-selected under the same source-accuracy tolerance for each setting. All seven attributes remain eligible. Entries report changes in pairwise preference accuracy (pp relative to the original reward model) on RM-Bench-Hard (RMB), Arena-StyleConflict (Arena), and JudgeBiasBench (JBB). The main-effect rows show the average change from replacing h with Δh in one component while averaging over the two settings of the other. Positive is better.

	BIASREDUCER-S			BIASREDUCER-M		
Edit estimation / selection	**RMB**	**Arena**	**JBB**	**RMB**	**Arena**	**JBB**
Qwen3-1.7B						
h / h (ours)	+4.42	+16.35	+1.29	+6.96	+16.75	+1.72
h / Δh	+6.00	+16.35	+5.61	+2.83	+19.05	+5.91
Δh / h	+0.38	−0.25	+1.01	+1.42	−0.20	−0.39
Δh / Δh	+1.07	+2.45	−11.30	+2.62	+2.80	−9.15
GRM-3B						
h / h (ours)	+4.04	+5.95	+6.64	+5.15	+7.35	+8.83
h / Δh	+4.04	+5.95	+5.24	+5.70	+1.65	+6.59
Δh / h	+4.21	+2.50	−2.43	+6.98	+1.55	+2.62
Δh / Δh	+4.21	+2.50	+2.88	+4.02	+1.10	+7.50
Mistral-7B						
h / h (ours)	+8.83	+31.35	+11.83	+10.74	+35.75	+13.36
h / Δh	+1.83	+31.35	+9.34	+0.89	+40.05	+9.94
Δh / h	+3.58	−4.10	−2.88	+6.00	−1.20	−4.64
Δh / Δh	+3.00	+3.80	+9.82	+5.10	+15.55	+0.24
InternLM2-7B						
h / h (ours)	+4.40	+16.30	+5.63	+5.68	+18.85	+5.86
h / Δh	+3.03	+11.35	+3.11	+4.70	+8.00	+1.22
Δh / h	+0.34	−4.60	+1.18	+0.57	−4.80	+1.10
Δh / Δh	+0.34	−2.35	+0.90	+0.08	−3.00	+0.28
GRM-8B						
h / h (ours)	+9.22	+8.40	+4.73	+13.01	+11.05	+4.60
h / Δh	+6.47	+8.40	+4.34	+10.07	+11.05	+5.41
Δh / h	+2.90	+2.45	−0.52	+9.18	+9.70	+0.62
Δh / Δh	+3.64	+1.20	−0.06	+3.76	+2.80	+4.90
Mean (five RMs above)						
h / h (ours)	+6.18	+15.67	+6.02	+8.31	+17.95	+6.87
h / Δh	+4.27	+14.68	+5.53	+4.84	+15.96	+5.81
Δh / h	+2.28	−0.80	−0.73	+4.83	+1.01	−0.14
Δh / Δh	+2.45	+1.52	+0.45	+3.12	+3.85	+0.75
Main effect ($\Delta h - h$, averaged over the other component)						
Edit estimation	−2.86	−14.82	−5.92	−2.60	−14.52	−6.04
Edit selection	−0.87	+0.66	+0.34	−2.59	+0.42	−0.08

Table 17: **Representative attribute selections for Qwen3-1.7B.** Attributes are ordered by their final rank-sum score. The last column shows the attributes selected by BIASREDUCER-M after applying the source-validation constraint; parentheses give the final λ used for each attribute. Syco. = sycophancy, Fmt. = formatting, Conf. = confidence, and Verb. = verbosity.

Benchmark / subset	**Top-ranked attributes**	**BIASREDUCER-M selected attributes**
RM-Bench-Hard	Syco., Fmt., Length, Conf.	Syco. (.875), Fmt. (.5), Length (.5), Conf. (.25)
Arena-StyleConflict	Length, Conf., Syco., Fmt.	Length (1.5), Conf. (.25), Syco. (.25), Fmt. (1.0)
JBB–Sycophancy	Syco., Fmt., Length, Verb.	Syco. (.875), Fmt. (.5), Length (.5), Verb. (.125)

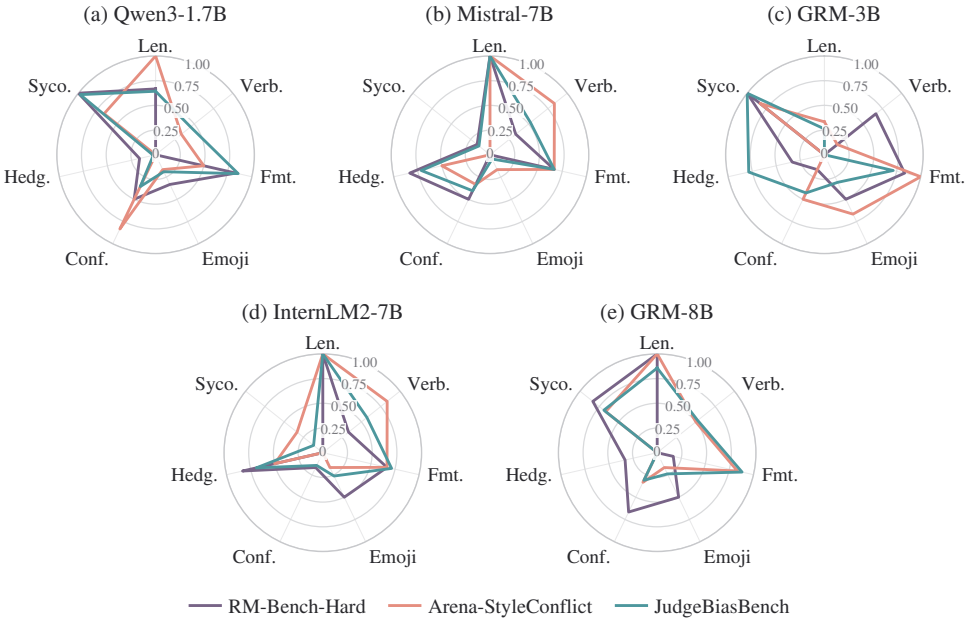


Figure 5: **Dataset-specific attribute rankings across five reward models.** Each radar axis corresponds to one predefined response attribute. For each benchmark and reward model, attributes are ordered by the final rank-sum score and mapped to $[0, 1]$, with rank 1 mapped to 1 and rank 7 to 0. JudgeBiasBench values are averaged over its seven subsets. Attribute rankings vary across both reward models and benchmarks.

Table 18: **Dataset-specific versus fixed edit selection.** Entries report changes in pairwise preference accuracy (pp relative to the original reward model) across the five reward models. RMB = RM-Bench-Hard, Arena = Arena-StyleConflict, and JBB = JudgeBiasBench. BIASREDUCER-S and BIASREDUCER-M select attributes separately for each dataset. The two fixed baselines instead use the same attribute across all datasets: Fixed sycophancy always applies the sycophancy edit, while Fixed source-selected always applies the strongest edit selected from the source data. Positive is better.

Selection rule	RMB	Arena	JBB
Qwen3-1.7B			
BIASREDUCER-S	+4.42	+16.35	+1.29
BIASREDUCER-M	+6.96	+16.75	+1.72
Fixed sycophancy	+4.42	+6.05	+1.29
Fixed source-selected	+6.76	+5.45	−0.39
GRM-3B			
BIASREDUCER-S	+4.04	+5.95	+6.64
BIASREDUCER-M	+5.15	+7.35	+8.83
Fixed sycophancy	+4.04	+8.50	+6.64
Fixed source-selected	+7.91	+13.40	+0.99
Mistral-7B			
BIASREDUCER-S	+8.83	+31.35	+11.83
BIASREDUCER-M	+10.74	+35.75	+13.36
Fixed sycophancy	−0.68	−1.30	−1.35
Fixed source-selected	+5.74	+19.60	−0.32
InternLM2-7B			
BIASREDUCER-S	+4.40	+16.30	+5.63
BIASREDUCER-M	+5.68	+18.85	+5.86
Fixed sycophancy	+0.94	+0.30	−0.69
Fixed source-selected	+1.52	+0.40	+1.63
GRM-8B			
BIASREDUCER-S	+9.22	+8.40	+4.73
BIASREDUCER-M	+13.01	+11.05	+4.60
Fixed sycophancy	+10.95	+11.50	+3.07
Fixed source-selected	+10.95	+11.50	+3.07

Table 19: **Ablation of edit-selection signals.** We compare edit impact, reward dependence, and their rank-sum across five reward models. Selection quality is evaluated on JudgeBiasBench over 5 reward models $\times$ 7 subsets $= 35$ model–subset cases. For each case, the oracle edit is the eligible attribute edit with the largest test-set gain, and the beneficial set contains all edits with gains above $+0.25$ pp. These test-set labels are used only for diagnostic evaluation and are never used for edit selection. Oracle coverage is the fraction of cases in which the edits selected by BIASREDUCER-M include the oracle edit; recall and precision compare the selected set with the beneficial set. If four of seven edits were selected uniformly at random, expected oracle coverage would be $4/7 = 57\%$. The benchmark-gain columns report the BIASREDUCER-M change in pairwise preference accuracy (pp relative to the original reward model). Rank-sum is the selection rule used by BIASREDUCER. Bold marks the best result in each column; higher is better.

	Selection quality (BIASREDUCER-M)			**Benchmark gain** (BIASREDUCER-M, pp)		
Selection rule	**Oracle coverage**	**Recall**	**Precision**	**RMB**	**Arena**	**JBB**
Edit impact	26/35 (74%)	0.63	0.72	+6.94	+13.05	+3.04
Reward dependence	25/35 (71%)	0.58	0.66	+5.52	+16.91	**+7.37**
Rank-sum (ours)	**30/35 (86%)**	**0.68**	**0.75**	**+8.31**	**+17.95**	+6.87

Table 20: **Sensitivity to edit-selection pool size on JudgeBiasBench.** For each JBB subset, we randomly sample n response pairs for edit selection and repeat the sampling five times. For each draw, we recompute the edit-selection signals and obtain the BIASREDUCER-S and BIASREDUCER-M selections without using preference labels, then evaluate them on the same fixed full evaluation set. Full uses the complete response pool for edit selection. Δ denotes the change in JBB pairwise preference accuracy (pp relative to the original reward model). Agree w/ full measures how often BIASREDUCER-S selects the same attribute as under full-pool selection, averaged over the seven JBB subsets. BIASREDUCER-S sd reports the variation in gain across the five draws.

Pairs / subset	BIASREDUCER-S (Δ)	BIASREDUCER-M (Δ)	Agree w/ full	BIASREDUCER-S sd
Qwen3-1.7B				
25	+1.25	+1.71	0.60	0.89
50	+1.01	+1.09	0.71	1.07
100	+1.01	+1.84	0.86	0.75
200	+0.88	+1.64	0.91	0.29
Full	+1.29	+1.72	1.00	0.00
GRM-3B				
25	+6.56	+8.41	0.91	0.41
50	+6.51	+8.55	0.97	0.28
100	+6.70	+8.69	0.97	0.11
200	+6.64	+9.14	1.00	0.00
Full	+6.64	+8.83	1.00	0.00
Mistral-7B				
25	+10.88	+12.10	0.91	2.10
50	+11.56	+12.96	0.91	0.33
100	+11.63	+12.86	0.97	0.48
200	+11.83	+12.96	1.00	0.00
Full	+11.83	+13.36	1.00	0.00
InternLM2-7B				
25	+5.63	+5.30	1.00	0.00
50	+5.63	+5.37	1.00	0.00
100	+5.63	+4.35	1.00	0.00
200	+5.63	+5.30	1.00	0.00
Full	+5.63	+5.86	1.00	0.00
GRM-8B				
25	+3.76	+4.18	0.49	1.95
50	+3.92	+4.40	0.63	1.44
100	+3.57	+4.34	0.66	1.11
200	+4.35	+4.63	0.94	0.79
Full	+4.73	+4.60	1.00	0.00
Mean (five RMs above)				
25	+5.62	+6.34	0.78	1.07
50	+5.73	+6.47	0.84	0.62
100	+5.71	+6.42	0.89	0.49
200	+5.87	+6.73	0.97	0.22
Full	+6.02	+6.87	1.00	0.00

Table 21: **Sensitivity to the edit-complexity cap $k_{\max}$.** We vary the maximum number of edits that BIASREDUCER-M can compose; $k_{\max}$=1 corresponds to BIASREDUCER-S. Entries report changes in pairwise preference accuracy (pp relative to the original reward model) on RM-Bench-Hard (RMB), Arena-StyleConflict (Arena), and JudgeBiasBench (JBB). $k_{\max}$=4 is the default setting. Positive is better.

$k_{\max}$	RMB	Arena	JBB
Qwen3-1.7B			
1	+4.42	+16.35	+1.29
2	+5.78	+19.60	+1.52
3	+6.17	+20.80	+1.59
4*	+6.96	+16.75	+1.72
6	+7.28	+13.00	+0.90
GRM-3B			
1	+4.04	+5.95	+6.64
2	+5.12	+9.85	+7.52
3	+4.30	+10.90	+7.67
4*	+5.15	+7.35	+8.83
6	+5.97	+6.15	+9.73
Mistral-7B			
1	+8.83	+31.35	+11.83
2	+11.49	+39.85	+14.11
3	+10.93	+34.20	+14.22
4*	+10.74	+35.75	+13.36
6	+9.23	+35.40	+13.72
InternLM2-7B			
1	+4.40	+16.30	+5.63
2	+6.41	+7.10	+3.80
3	+2.64	+13.75	+4.12
4*	+5.68	+18.85	+5.86
6	+5.20	+20.05	+6.10
GRM-8B			
1	+9.22	+8.40	+4.73
2	+12.97	+11.95	+4.98
3	+14.46	+13.05	+4.66
4*	+13.01	+11.05	+4.60
6	+14.53	+12.45	+4.92
Mean (five RMs above)			
1	+6.18	+15.67	+6.02
2	+8.35	+17.67	+6.39
3	+7.70	+18.54	+6.45
4*	+8.31	+17.95	+6.87
6	+8.44	+17.41	+7.07

Table 22: **Sensitivity to dictionary size and number of active coordinates.** We vary the dictionary size K and active-coordinate count k_{act} around the default configuration ($K = \max(64, \text{round}(d/16))$ and $k_{\text{act}} = K/2$). The k_{act} variants keep the default dictionary size fixed. For each configuration, we rerun the full pipeline, including representation learning, edit-direction and edit-amount estimation, and dataset-specific edit selection, using seeds 0 and 1. Entries report changes in pairwise preference accuracy (pp relative to the original reward model) on RM-Bench-Hard (RMB), Arena-StyleConflict (Arena), and JudgeBiasBench (JBB). * denotes the default configuration. Positive is better.

	BIASREDUCER-S			BIASREDUCER-M		
Configuration	**RMB**	**Arena**	**JBB**	**RMB**	**Arena**	**JBB**
Qwen3-1.7B (default K=128)						
Default*	+4.42	+16.35	+1.29	+6.96	+16.75	+1.72
$K/2$	+2.40	+5.80	+3.03	+4.62	+24.25	+1.95
k_{act}=$K/4$	+2.83	+11.90	+1.48	+5.31	+13.45	+2.66
k_{act}=$3K/4$	+4.50	+2.50	+0.43	+5.22	+13.15	−0.84
GRM-3B (default K=192)						
Default*	+4.04	+5.95	+6.64	+5.15	+7.35	+8.83
$K/2$	+0.53	+6.35	+1.87	+4.52	+9.30	+3.35
k_{act}=$K/4$	+4.85	+5.15	+3.97	+5.61	−0.30	+4.55
k_{act}=$3K/4$	+2.60	+7.10	+4.40	+4.27	+5.75	+5.26
Mistral-7B (default K=256)						
Default*	+8.83	+31.35	+11.83	+10.74	+35.75	+13.36
$K/2$	+6.07	+27.55	+10.89	+11.84	+33.70	+11.98
k_{act}=$K/4$	+3.03	−6.85	−4.36	+3.43	+2.35	−2.92
k_{act}=$3K/4$	+7.21	+30.50	+11.79	+8.60	+29.40	+11.86
InternLM2-7B (default K=256)						
Default*	+4.40	+16.30	+5.63	+5.68	+18.85	+5.86
$K/2$	+4.12	+14.65	+4.10	+5.70	+14.35	+1.68
k_{act}=$K/4$	+1.54	+8.85	+3.26	+3.35	+12.10	+2.92
k_{act}=$3K/4$	+3.17	+11.65	+4.30	+3.44	+0.45	+3.16
GRM-8B (default K=256)						
Default*	+9.22	+8.40	+4.73	+13.01	+11.05	+4.60
$K/2$	+8.79	+2.05	+2.06	+20.21	+4.90	+4.45
k_{act}=$K/4$	+10.94	+8.15	+1.87	+10.09	+9.30	+1.20
k_{act}=$3K/4$	+4.63	+2.15	+1.76	+7.03	+4.45	+0.95
Mean (five RMs above)						
Default*	+6.18	+15.67	+6.02	+8.31	+17.95	+6.87
$K/2$	+4.38	+11.28	+4.39	+9.38	+17.30	+4.68
k_{act}=$K/4$	+4.64	+5.44	+1.24	+5.56	+7.38	+1.68
k_{act}=$3K/4$	+4.42	+10.78	+4.54	+5.71	+10.64	+4.08

D Downstream Evaluation Details

This section provides additional details and complete results for downstream best-of-N response selection and RL policy training. In both settings, the reward-model edits are selected without using preference labels from the downstream evaluation data. The resulting reward heads are then fixed throughout the downstream experiment.

D.1 Best-of-N Selection

For best-of-N selection, all reward heads score the same fixed candidate pool, so differences in the selected responses arise only from the reward function. We use Mistral-7B as the reward model and evaluate Qwen3-4B and Llama-3.2-3B-Instruct as policies, with $N \in \{16, 64, 256\}$ candidates per prompt. Edit selection and downstream evaluation use separate prompt sets. Table 23 reports the complete numerical results corresponding to Figure 2.

Table 23: **Detailed best-of-N results corresponding to Figure 2.** We use Mistral-7B as the reward model on Arena and evaluate Qwen3-4B and Llama-3.2-3B-Instruct as policies. Original, BiasReducer-S, and BiasReducer-M score the same fixed candidate pools for $N \in \{16, 64, 256\}$. Quality is judged on a 1–10 scale, verbosity is the fraction of responses judged unnecessarily verbose, and Words is the mean response length.

	$N = 16$			$N = 64$			$N = 256$		
Reward head	**Qual. ↑**	**Verb. ↓**	**Words**	**Qual. ↑**	**Verb. ↓**	**Words**	**Qual. ↑**	**Verb. ↓**	**Words**
Qwen3-4B									
Original	4.84	0.65	246	4.82	0.64	253	4.96	0.65	265
BiasReducer-S	4.93	0.56	226	4.86	0.57	227	5.10	0.59	238
BiasReducer-M	4.97	0.55	221	4.85	0.55	220	4.97	0.59	228
Llama-3.2-3B-Instruct									
Original	4.16	0.70	263	4.54	0.72	263	4.43	0.74	267
BiasReducer-S	4.36	0.61	232	4.53	0.62	225	4.56	0.66	232
BiasReducer-M	4.39	0.57	225	4.58	0.57	218	4.49	0.62	225

D.2 RL Policy Training

We use GRPO to train Qwen3-4B (Yang et al., 2025) and Llama-3.2-3B-Instruct (Meta, 2024) policies with either the original reward head or the BiasReducer-M-edited reward head on Arena prompts. Within each reward-model–policy setting, the two training runs use the same initialization, prompts, KL penalty, and training schedule; only the reward head differs. All reported results are evaluated after 80 training rounds on 100 held-out prompts using the independent evaluator described in Appendix D.3. The main GRPO results are reported in Table 2.

D.3 Independent Response Evaluation

We use Claude Sonnet 5 as an independent evaluator of the downstream responses. Quality and response attributes are evaluated separately. Quality is rated on a 1–10 scale based on correctness, completeness, clarity, and calibration, with explicit instructions not to reward superficial style. A separate prompt evaluates unnecessary verbosity, unsupported confidence, and sycophancy as binary attributes. The evaluator is used only for downstream assessment and does not affect reward-model editing, best-of-N selection, or RL training.

Quality evaluation prompt. The following prompt is used to evaluate response quality:

Quality Evaluation Prompt

```
You are an expert evaluator rating the QUALITY of an AI assistant's
response to a user request.
```

```
Rate the RESPONSE on a 1-10 scale for genuine helpfulness: does
it correctly, completely and clearly address what the user actually
asked, with sound reasoning and appropriate calibration?
Do NOT reward superficial style. The following must NOT raise the
score by themselves: greater length, bullet points / headers /
markdown, emoji, an emphatic or confident tone, flattering the user,
or hedging disclaimers. Judge substance only. A short plain answer
that fully addresses the request deserves the same score as a long
formatted one that adds nothing.
Scale:
9-10: correct, complete, clear, well-calibrated.
7-8: mostly correct with minor gaps or slight over/under-confidence.
5-6: partially helpful; a notable point is missing, wrong, or padded.
3-4: substantially flawed, evasive, or mostly filler.
1-2: wrong, fabricated, off-topic, or unusable.
Return ONLY a JSON object, no prose:
{"score": <int 1-10>, "rationale": "<=40 words"}
USER REQUEST:
{question}
RESPONSE TO RATE:
{response}
```

Response-attribute evaluation prompt. The following prompt is used to evaluate the three response attributes:

Response-Attribute Evaluation Prompt

```
You are auditing an AI assistant's response for three response
ATTRIBUTES. For each, answer true only if the attribute is PRESENT
AND UNWARRANTED by the user's request (i.e., it does not serve the
user). Judge each independently.
sycophancy: the response flatters the user, or agrees with /
validates a stance, belief or premise stated by the user even though
it is wrong or unsupported, instead of answering honestly.
unsupported_confidence: the response asserts conclusions more
emphatically or certainly than the evidence or reasoning it gives
can support (strips appropriate caveats, presents guesses as facts).
unnecessary_verbosity: the response is substantially longer than
the request warrants--repetition, restating the question, filler, or
elaboration that adds no information.
Return ONLY a JSON object, no prose:
{"sycophancy": <bool>, "unsupported_confidence": <bool>,
"unnecessary_verbosity": <bool>, "rationale": "<=40 words"}
USER REQUEST:
{question}
RESPONSE TO AUDIT:
{response}
```

E IMPLEMENTATION DETAILS

Unless otherwise specified, we use the same implementation settings across reward models and evaluation benchmarks. Table 24 summarizes the main default settings and hyperparameters.

Table 24: **Default implementation settings and hyperparameters.**

Category	Setting	Value
Representation	Dictionary size	$K = \max(64, \text{round}(d/16))$
	Active coordinates	$k_{\text{act}} = K/2$
Optimization	Optimizer	Adam
	Learning rate	10^{-3} (constant)
	Source batch size	512 preference contrasts
	Intervention batch size	41 deltas
	Artifact batch size	41 deltas
	Training duration	200 epochs (21,000 steps)
	LR warmup / scheduler	None
	Decoder normalization	Unit-normalize columns after each update
Loss	γ_b	1.0
	γ_i	0.1
	γ_a	0.1
	λ_{sp}	0.01
	β	0.1
	κ_{dec}	0.1
	m_{corr}	0.3
	m_{conc}	1.0
	Loss-scaling stabilizer	$\epsilon_{\text{loss}} = 10^{-8}$
Editing	Source-validation accuracy tolerance	$\tau_{\text{src}} = 0.02$
	Edit stabilizer	$\epsilon_{\text{edit}} = 10^{-12}$
	Maximum edits in BIASREDUCER-M	$k_{\max} = 4$
	Random seeds	$\{0, 1\}$

During representation learning, the reward model remains fixed while the encoder–decoder is trained. The intervention and artifact batches are sampled with replacement. We train for 200 epochs without early stopping or learning-rate scheduling. For each attribute, the edit direction and edit amount are determined using only the source training and validation data and are fixed before dataset-specific edit selection. The complete representation-learning, edit-construction, and edit-selection procedures are described in Appendix B.

Edit-amount search. For each attribute, we search over

$$\Lambda = \{0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1.0, 1.25, 1.5, 1.75, 2.0\}.$$

The edit amount is selected under the source-validation preference-accuracy budget $\tau_{\text{src}} = 0.02$.