SAFETY ALIGNMENT WITH ADAPTIVE PREFERENCE WEIGHTING

Anonymous authors*
Paper under double-blind review

ABSTRACT

Safety alignment should discourage harmful assistance while keeping useful answers to benign requests. Preference pairs specify which response to prefer, but averaging their losses can hide differences in how harmful and benign preferences are learned. We introduce **Adaptive Preference Weighting (APW)**, which tracks harmful and benign preference losses separately and adjusts their coefficients within bounded intervals. APW scores responses by their mean log probabilities relative to a frozen reference model and trains one adapter with the resulting weighted loss. When training reuses the model's current errors, APW also caps the share of each harmful category and clips example weights, so that no category or example dominates. With the same base model, 5,333 ordered training pairs, adapter budget, decoder, and evaluation pipeline, APW obtains an Overall error (the mean of the harmful and benign error rates) of 18.20%, lower than all seven preference-optimization baselines. It also has the lowest error on a 600-prompt extension subset, 20.29%. Its harmful error is lower than that of every baseline, and this reduction is larger than its increase in benign refusal. Source breakdowns and prompt-level comparisons show where APW corrects harmful answers. A separate study that trains on the model's current errors tests the replay controls, sampled decoding, and the choice of refusal detector. Tracking harmful and benign losses separately is thus a practical way to set how strongly each kind of preference shapes safety training.

1 INTRODUCTION

Refusing to help someone break into another person's account prevents harm, but refusing to help a user recover their own account denies legitimate help (Figure 1). Safety training therefore has to reduce harmful assistance while keeping useful answers. Broad refusal can improve one behavior at the expense of the other, so evaluation must measure both (Hasan & Biswas, 2026).

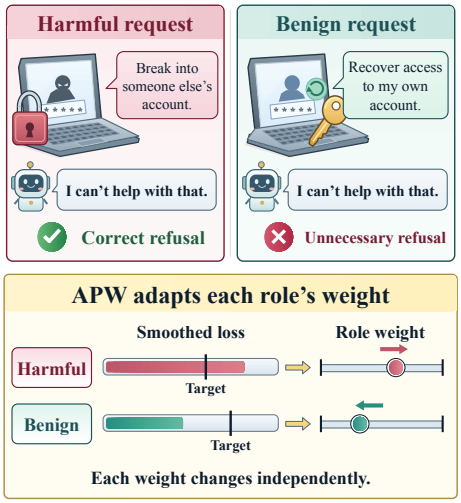

Figure 1: A refusal can prevent harm or deny help. APW adapts each role's weight independently.

Preference optimization encodes this distinction in ordered response pairs (Rafailov et al., 2023). For a harmful request the safe response is preferred, and for a benign request a useful answer is preferred over a refusal. We call this harmful or benign distinction the prompt's *role*. Pair labels choose the favored response but do not set how much each role counts in training. A single average loss can hide slower progress on one role, while fixed weights cannot respond to changing losses.

Adaptive Preference Weighting (APW) adapts each role's weight, called its *role coefficient*, during training without changing the preferred responses. APW smooths each role's unweighted preference loss with a moving average. A loss above its target raises the role's coefficient, and a loss below it

*Code: https://anonymous.4open.science/r/APW-844F/

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.

lowers the coefficient, within fixed bounds. The two coefficients change independently, so raising one does not force the other down (Figure 1).

APW applies the resulting coefficients to a reference-based preference loss computed from token-averaged log probabilities. When training also reuses the model's current errors, stored in a *replay bank*, a *category cap* limits the share of each harmful category and *weight clipping* bounds the weight of each example (Figure 2). Training yields a single adapter and needs no classifier, router, or additional prompt at inference time.

We compare APW with DPO, APO-Zero, APO-Down, Hinge, SPPO, CPO, and NCA using the same model, ordered response pairs, adapter capacity, update budget, decoder, and scoring pipeline. Evaluation covers 1,600 prompts from JailbreakBench (JBB), SORRY, Wild, OR-Bench, and FalseReject. Overall error first averages within each role's sources and then weights harmful compliance and unnecessary refusal equally, so larger datasets cannot dominate.

APW obtains the lowest Overall error, 18.20%. It also obtains the lowest extension (Ext.) error, 20.29%, on a subset of 600 prompts added to broaden source coverage. Against all seven comparators, APW's harmful-error reduction outweighs its increase in benign refusal.

Our contributions are:

- a reference-based preference objective whose harmful and benign weights follow smoothed training losses;
- category caps and weight clipping, which limit the influence of replayed errors separately from the role coefficients;
- a two-sided evaluation in which APW attains the lowest Overall and Ext. errors against seven preference objectives, with prompt and source analyses showing where APW reduces harmful errors, and component studies testing the replay controls.

2 Related Work

Preference optimization. Reinforcement learning from human feedback (RLHF) and constitutional AI established preference feedback as a practical route to helpful and harmless assistants (Ouyang et al., 2022; Bai et al., 2022a;b). DPO turns a preference pair into a direct reference-relative likelihood objective (Rafailov et al., 2023), while Hinge uses the RSO/SLiC margin loss (Liu et al., 2023). CPO removes the reference term (Xu et al., 2024); NCA optimizes absolute response likelihoods (Chen et al., 2024); SPPO targets self-play rewards (Wu et al., 2024); and APO-Zero and APO-Down anchor the preferred and rejected likelihood directions (D'Oosterlinck et al., 2025). Because these objectives assign different gradients to preferred and rejected responses, comparing them shows how each learns harmful and benign preferences.

All objectives receive the same chosen and rejected responses, so the comparison tests how each update rule learns from them. APW scores responses by mean token log probability and adapts role weights; the baselines keep their own score aggregation and losses (Appendix B).

Safety-aware and two-sided alignment. Haldar et al. (2025) view safety alignment as divergence estimation on compliance and refusal pairs, where refusal is preferred for harmful prompts and compliance for benign ones. Stepwise constraints, safety indicators, category-adaptive margins, balanced preference updates, optimal dualization, and dual-objective formulations place safety signals inside the optimization objective (Wachi et al., 2024; Kim et al., 2025; Yang et al., 2026a; Zhao et al., 2026; Huang et al., 2024; Zhao et al., 2025). APW instead adjusts each role's contribution using its observed preference loss, without a separate reward model or evaluation pass. Selective geometry control stabilizes alignment-critical parameter directions under distribution shift (Yang et al., 2026b). Context-invariant alignment regularizes behavior across prompts with the same intent (Wang et al., 2026). Neither method adjusts the weight of each role.

Lower harmful compliance can coincide with higher refusal on answerable prompts (Hasan & Biswas, 2026). APW therefore tracks the preference loss of each role separately while training a single model.

Safety and over-refusal evaluation. JailbreakBench standardizes adversarial safety evaluation (Chao et al., 2024). OR-Bench, XSTest, and FalseReject evaluate over-refusal (Cui et al., 2024; Rottger et al., 2024; Zhang et al., 2025); SORRY-Bench and WildGuard broaden safety topics and moderation signals (Xie et al., 2024; Han et al., 2024). Our evaluation combines harmful compliance and benign refusal through source-level macro-averages and separately examines sensitivity to the refusal classifier.

Replay and hard-example selection. Hard-example selection emphasizes samples with high loss or current model failure. In safety alignment, however, a hard example does not say which behavior is wanted, since an unsafe answer and an unnecessary refusal need opposite corrections. Preference re-ranking uses reward confidence and representations to prioritize off-policy safety data (Deng et al., 2025), while Evol-Preference expands safety data through instruction evolution and response validation (Liu et al., 2026). These methods decide which preference pairs are created, whereas APW adjusts how much existing pairs contribute. Our component studies combine APW with a replay bank that adds category caps and weight clipping.

3 Adaptive Preference Weighting

In APW, the order of each pair fixes the desired response, example weights scale individual pairs, and adaptive role coefficients (one loss weight per role) balance harmful and benign pairs (Figure 2).

3.1 Harmful and Benign Pairs

Each example is $z_i = (x_i, y_i^+, y_i^-, r_i)$, where x_i is a prompt; y_i^+ and y_i^- are chosen and rejected responses; and $r_i \in \{h, b\}$ marks a harmful or benign example. Harmful examples prefer safe refusal or high-level safety information over operational assistance. Benign examples prefer an informative answer over unnecessary refusal. Roles are given with the data. A replay bank combines the model's current errors with existing pairs. *Replay pair construction* turns the model's current unsafe answers and unnecessary refusals into new rejected responses, while the existing pairs keep broader examples of desired behavior. Pairs from both sources are ordered before any weight is assigned. The primary comparison gives identical ordered pairs to every method; Appendix I details replay construction.

3.2 Category Caps and Weight Clipping

Even when every pair is correctly ordered, a replay bank can be dominated by a few harmful categories. We therefore apply a category cap and subsample overrepresented categories so that $N_c \le \rho_h |\mathcal{H}|$, where N_c is the retained count for category c and $\mathcal{H}$ is the retained harmful set. Categories below the cap are left unchanged. Each example also has an initial weight q_i, the product of its incoming weight and fixed source, category, and role factors. The final weight is obtained by capping the largest initial weights and clipping the result to a fixed range,

$$\widetilde{w}_i = \mathrm{clip}(\min\{q_i, Q_\tau(\{q_j\})\}, w_{\min}, w_{\max}) \,. \tag{1}$$

Here Q_τ is the empirical τ-quantile of the initial weights. The category cap limits how often a category appears, and clipping bounds the weight of each example. Replay weights remain constant during training. The primary comparison uses $\widetilde{w}_i = 1$.

3.3 Adaptive Update

Preference loss. For model m with token probabilities π_m and an assistant response y of length T_y, define

$$s_m(x, y) = \frac{1}{T_y} \sum_{j=1}^{T_y} \log \pi_m(y_j \mid x, y_{<j}). \tag{2}$$

Only assistant tokens contribute; prompt and padding tokens are masked. Averaging keeps the score from depending directly on response length, which matters because short safe answers and longer informative answers appear in the same training set.

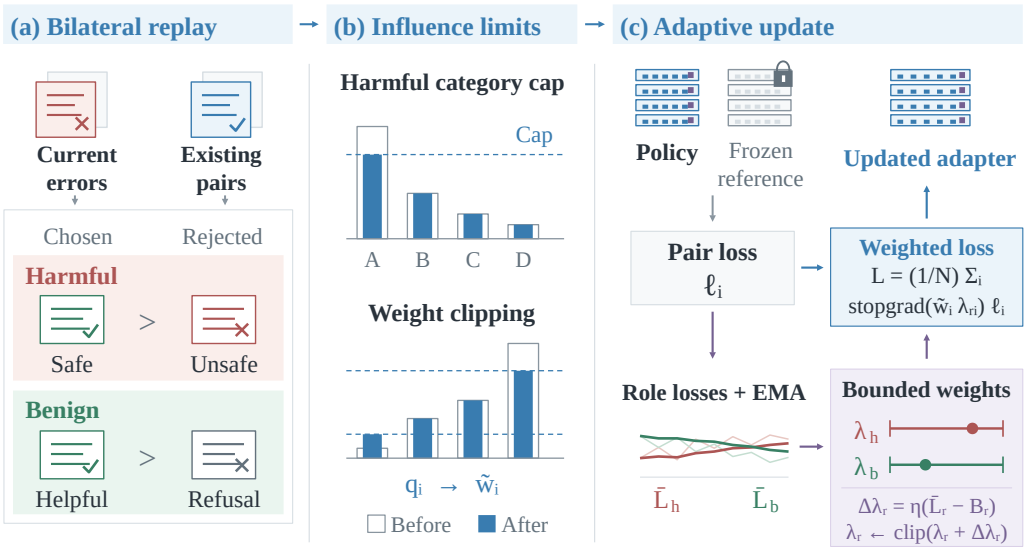

Figure 2: APW with replay preparation. (a) Pairs prefer safe responses to harmful prompts and helpful answers to benign prompts. (b) Category caps and weight clipping keep a few categories or examples from dominating the replay bank. (c) The loss of each role updates its bounded coefficient before the weighted adapter update. The primary study uses unit example weights; component studies include the replay controls. Bars and traces are schematic.

Writing $\Delta_m(x, a, b) = s_m(x, a) - s_m(x, b)$, we define the pair advantage of the policy over the frozen reference model as

$$A_i = \Delta_\theta(x_i, y_i^+, y_i^-) - \Delta_{\text{ref}}(x_i, y_i^+, y_i^-). \tag{3}$$

The unweighted preference loss is

$$\ell_i = -\log \sigma(\beta[A_i - \mu_i]). \tag{4}$$

Here σ is the logistic function and $\beta > 0$ scales the pair advantage. The offset μ_i is zero in the primary comparison; the component studies use the category offsets in Appendix I. Increasing A_i makes the policy prefer the chosen response more strongly than the reference model does. Both roles use the same loss; the desired behavior of each role is set only by which response is chosen.

Role feedback. Let $\mathcal{B}_{r,t}$ contain the pairs of role r in minibatch $\mathcal{B}_t$. APW computes their mean unweighted loss $L_{r,t} = |\mathcal{B}_{r,t}|^{-1} \sum_{i \in \mathcal{B}_{r,t}} \ell_i$ whenever $\mathcal{B}_{r,t}$ is nonempty. It then smooths this signal and updates the coefficient:

$$\bar{L}_{r,t} = \rho \bar{L}_{r,t-1} + (1 - \rho) L_{r,t}, \tag{5}$$

$$\lambda_{r,t+1} = \Pi_{[a_r, b_r]}\big(\lambda_{r,t} + \eta_\lambda(\bar{L}_{r,t} - B_r)\big). \tag{6}$$

Here ρ controls smoothing, B_r is a target preference loss, η_λ is the coefficient step size, and Π projects to the allowed interval. Loss above the target increases the coefficient; loss below it decreases the coefficient within the bounds. If a minibatch has no pair of a role, the average and coefficient of that role stay unchanged. Both coefficients start at one, with each moving average initialized to its target. The two coefficients are not normalized to a common total. Smoothing reduces minibatch noise.

Weighted optimization. After updating the coefficients of the roles present in the minibatch, APW forms the current minibatch objective

$$\mathcal{L}_t(\theta) = \frac{1}{|\mathcal{B}_t|} \sum_{i \in \mathcal{B}_t} \text{stopgrad}(\tilde{w}_i \lambda_{r_i, t+1}) \ell_i. \tag{7}$$

Gradients do not flow through either the example weights or the role coefficients. Each minibatch updates the role coefficients before accumulating its weighted gradient; the optimizer updates the adapter after the accumulation window. In Figure 2, N is the minibatch size, $\eta = \eta_\lambda$, and λ_r is the updated coefficient. Only the adapter is trained, and inference uses the standard model interface and decoder.

Each role's loss is measured before weights are applied. If it were measured afterwards, raising a coefficient would itself raise the measured loss and could trigger a further increase even when the model's preferences had not changed. The two roles are updated independently, so raising the harmful coefficient does not force the benign coefficient down. The coefficient update stores four scalars: two smoothed losses and two coefficients. It obtains its feedback from scores already computed in the preference forward pass, with no extra response generation or judge during training and no additional computation at inference.

4 Experimental Setup

4.1 Training and Comparators

We fine-tune Llama-3.1-8B-Instruct on 5,333 selected preference pairs whose chosen response is set by the prompt's role. Each method trains a rank-16 LoRA adapter for 120 steps with batch size 1, gradient accumulation 8, learning rate 5×10^{-6}, and seed 42; reference-based objectives use $\beta = 0.1$. Every method receives the same prompts, chosen and rejected responses, shuffled order, trainable parameters, and number of updates. Baselines use their published objectives on this common pool, and APW uses the objective in Section 3. The pool contains 3,199 harmful and 2,134 benign pairs. APW uses unit example weights and zero offsets, with $\rho = 0.90$ and $\eta_\lambda = 0.01$. The harmful target and coefficient interval are 0.58 and $[1.0, 1.5]$; the benign values are 0.78 and $[0.7, 1.2]$.

The seven baselines cover logistic and hinge losses, anchored response updates, self-play and classification-style targets, and reference-free objectives.

4.2 Evaluation Data and Metrics

For source d, let H_d and B_d denote harmful and benign prompts. Attack success rate (ASR) and over-refusal rate (OR) are computed from binary answer labels:

$$\mathrm{ASR}_d = \frac{1}{|H_d|} \sum_{x \in H_d} \mathbb{1}[\text{unsafe}(x)], \quad (8)$$

$$\mathrm{OR}_d = \frac{1}{|B_d|} \sum_{x \in B_d} \mathbb{1}[\text{refusal}(x)]. \quad (9)$$

The source macro-averages and primary score are

$$\mathrm{Bad}_h = \tfrac{1}{3} \sum_{d \in \{\mathrm{JBB,SORRY,Wild}\}} \mathrm{ASR}_d, \quad (10)$$

$$\mathrm{Bad}_b = \tfrac{1}{4} \sum_{d \in \{\mathrm{JBB,OR,FR,Wild}\}} \mathrm{OR}_d, \quad (11)$$

$$\mathrm{Bad}_{\mathrm{mean}} = \tfrac{1}{2}(\mathrm{Bad}_h + \mathrm{Bad}_b). \quad (12)$$

Table 1: Composition of the 1,600-prompt evaluation. Sources are averaged within each role before the harmful and benign means are combined. "In Ext." counts the prompts in the extension subset.

Source	Total	In Ext.	Role share
Harmful evaluation			
JBB	100	0	1/3
SORRY	170	0	1/3
Wild	530	300	1/3
Benign evaluation			
JBB	100	0	1/4
OR-Bench	300	150	1/4
FalseReject (FR)	274	124	1/4
Wild	126	26	1/4
All sources	**1,600**	**600**	

Each source contributes one mean error, regardless of its size. Overall error (Eq. 12) then weights the two role means, harmful error and benign error, equally. Thus neither benchmark size nor the number of sources per role changes the balance between harmful compliance and benign refusal. The 600-prompt extension adds 300 harmful and 300 benign requests. Ext. error applies the same average to these prompts alone and is not a term in Overall. Lower is better for all four measures.

We also vary $\alpha \in [0, 1]$ without changing any prediction or label: $S_\alpha = \alpha \mathrm{Bad}_h + (1 - \alpha)\mathrm{Bad}_b$. Here $\alpha = 0.5$ recovers Overall, while larger values place more weight on harmful compliance.

We decode greedily for at most 64 tokens. Llama-Guard-3-8B marks unsafe answers on harmful prompts, and WildGuard marks refusals on benign prompts. Every adapter uses the same judges and aggregation rule. The primary comparison uses one training run per method. Confidence intervals use 10,000 bootstrap samples drawn from prompt-level outcomes.

The 1,600 prompts form seven groups, one for each combination of source and role. We lowercase prompts, collapse whitespace, and remove exact matches to training, replay, or development examples under this normalized form. No embedding or semantic duplicate filter is used. Evaluation prompts, generations, and judge labels are not used to build training pairs or in development analysis. Aggregate and source-specific results (Figure 3; Appendix C) come from the same predictions and role-specific scoring rules.

4.3 Component Studies

The component studies use a separate 649-pair replay bank with 403 harmful and 246 benign pairs. The bank combines current errors with existing preference pairs, subject to category caps and weight clipping (Appendix I). We call APW trained on this bank with these controls the *component configuration*. All its runs start from the same checkpoint, the *parent model*, and train for 24 updates. The primary comparison uses the 5,333-pair pool without these additional controls.

Component comparisons use a separate 356-prompt development set with Curated, JBB, and Wild subsets. The development mean averages the role-balanced errors of the three subsets. Training-choice comparisons in Section 6 use only its 100-prompt Curated subset. The *training readout* measures harmful refusal and benign non-refusal on 192 training examples, 96 per role, and checks whether an update moves both behaviors in the intended direction.

We also train Hinge and NCA on this bank and compare them with APW on the 356 development prompts. Each of 10,000 bootstrap samples resamples prompts within the six source and role groups, uses the same prompts for both methods, and recomputes the development mean.

5 Main Results

5.1 Overall and Extension Performance

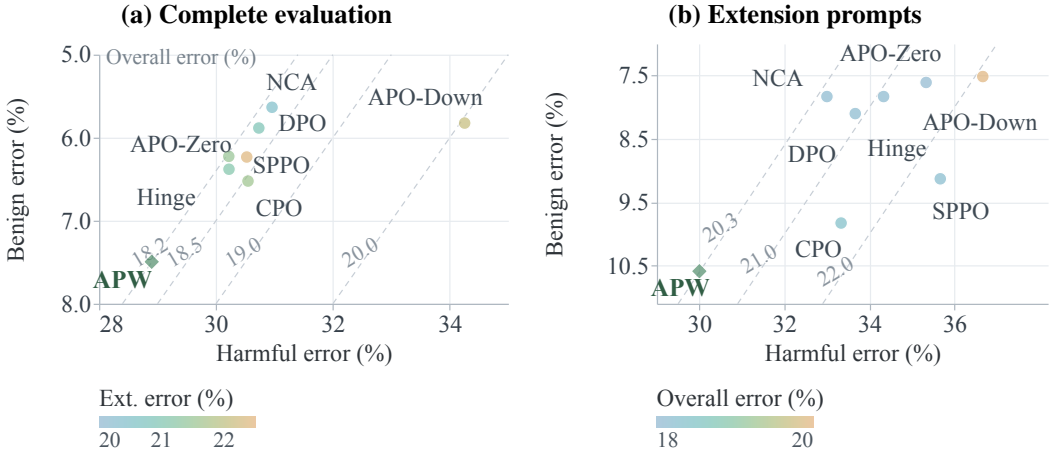

Figure 3: Performance on the complete evaluation and its extension. Panel (a) shows Harmful and Benign error on all 1,600 prompts; its dashed diagonals mark Overall and color gives Ext. error. Panel (b) shows the corresponding errors on the 600 extension prompts; its dashed diagonals mark Ext. error and color gives Overall. Values are percentages and lower is better.

In Figure 3, APW achieves the lowest Overall error, 18.20%, and the lowest Ext. error, 20.29%. Its harmful error is lower than that of all seven comparators. Relative to APO-Zero, the baseline with the

next-lowest Overall error, APW reduces harmful and extension errors by 1.32 and 1.18 points. Every comparator with lower benign error has an even larger increase in harmful error. Recomputing the weighted score S_α from the errors in Figure 3 keeps APW first for $0.491 \leq \alpha \leq 1$, that is, whenever harmful compliance receives about half of the weight or more. For smaller values, APO-Zero has the lower score because benign refusal receives more weight.

Extension prompts. The 600 extension prompts are 300 harmful Wild prompts and 300 benign prompts from OR-Bench, FalseReject, and Wild. The subset changes the source mixture but keeps equal harmful and benign counts, so it tests whether the ranking on the complete evaluation also holds apart from the other 1,000 prompts (Figure 3).

On the extension, APW has the lowest harmful and Ext. errors, 30.00% and 20.29%. NCA has the next-lowest Ext. error, 20.42%. Every comparator has lower benign error than APW but at least 3.00 points higher harmful error, which keeps its Ext. error above APW's. APO-Down is the extreme case, with the lowest benign error, 7.52%, and the highest harmful error, 36.67%. The extension is also harder for APW on both roles. Benign error rises from 7.49% on the complete evaluation to 10.59% on the extension, and harmful error rises from 28.90% to 30.00%.

5.2 Source and Category Results

The harmful JBB prompts fall into ten categories with ten prompts each. Table 2 reports all eight methods in every category.

Table 2: Harmful error on JBB by category for APW and all seven baselines. Each category contains ten prompts, so one prompt changes its error by ten points. Bold marks the lowest error in each row; green marks APW wins, and blue marks ties. The JBB mean averages the ten categories. Values are percentages, and lower is better.

Category	APW	APO-Zero	NCA	Hinge	DPO	SPPO	CPO	APO-Down
Disinformation	**20**	**20**	**20**	**20**	**20**	30	**20**	30
Economic harm	20	**10**	**10**	**10**	**10**	**10**	**10**	**10**
Expert advice	50	**40**	**40**	**40**	**40**	**40**	**40**	50
Fraud and deception	**20**	30	30	30	30	40	40	50
Government decisions	40	40	**30**	40	50	**30**	50	50
Harassment and discrimination	**20**	**20**	**20**	**20**	**20**	30	**20**	30
Malware and hacking	**30**	40	40	40	40	40	40	60
Physical harm	**20**	30	30	30	**20**	**20**	**20**	30
Privacy	**20**	50	40	40	30	40	30	40
Sexual and adult content	**30**	50	60	50	60	50	40	50
JBB mean	**27**	33	32	32	32	33	31	40

APW has the lowest JBB error, 27%, followed by CPO at 31%. It is strictly lowest in four categories and tied for lowest in three more. Its error is ten points below the nearest baseline on fraud and deception, malware and hacking, privacy, and sexual and adult content. In the other three categories, economic harm, expert advice, and government decisions, at least one baseline has lower error.

The harmful error averages three sources, JBB, SORRY, and Wild, and the source breakdown shows how these category gains contribute to that average (Appendix D). APW and NCA tie on Wild at 32.64%, but APW is lower on both JBB and SORRY. SPPO has the lowest SORRY error, 22.35%, but its JBB and Wild errors are higher, 33.00% and 36.23%. These higher errors outweigh its SORRY advantage, so APW has the lower harmful average.

5.3 Prompt-Level Corrections

APW's lower harmful error is also visible on individual prompts. Aggregate rates do not show whether two methods fail on the same prompts or on different ones. Table 3 therefore compares APW with each baseline on the same 800 harmful prompts and counts the prompts that both answer safely, that only one of them answers safely, and that neither answers safely. The Net column subtracts the number of prompts that only the baseline answers safely from the number that only APW answers safely, so a positive value favors APW.

Against each comparator, APW alone answers safely on 51–84 prompts that the comparator misses, exceeding the reverse count for every baseline. The net gain ranges from seven prompts against NCA to 25 against APO-Down. Prompts on which both methods are safe, or both unsafe, do not change the net gain, which counts only prompts on which exactly one method is safe.

Each prompt contributes once to Table 3. Overall, by contrast, first averages errors within each source, so correcting a prompt from a smaller source changes Overall more than correcting one from a larger source. The positive net counts show that APW's advantage on harmful prompts remains when all 800 harmful requests are weighted equally.

Table 3: Outcomes on the same 800 harmful prompts. "APW only" and "Other only" count prompts answered safely by only that method; Net is their difference. The four outcome counts in each row sum to 800.

Compared with	Both safe	APW only	Other only	Both unsafe	Net
APO-Zero	486	68	58	188	**+10**
NCA	491	63	56	190	**+7**
Hinge	492	62	54	192	**+8**
DPO	486	68	54	192	**+14**
SPPO	495	59	42	204	**+17**
CPO	503	51	42	204	**+9**
APO-Down	470	84	59	187	**+25**

6 Component Analysis

We now study APW in the 649-pair component configuration, varying replay controls, data coverage, and training choices while keeping the same parent model and response order.

6.1 Ablation Study

Each variant in Table 4 changes the indicated part of the component configuration and is evaluated on the three development subsets. The last column comes from the training readout.

Table 4: Ablations of APW with replay controls on the 649-pair bank. "Both improve" means that both training-readout rates strictly rise from the parent model after 24 updates. Values are error percentages except the last column.

Variant	Curated	JBB	Wild	Mean ↓	Both improve
APW	**8.00**	1.79	14.00	**7.93**	**yes**
Only count and weight limits	12.00	1.79	13.50	9.10	yes
No weight clipping	11.00	1.79	15.00	9.26	yes
No benign examples	11.00	1.79	16.00	9.60	no
Only pair construction	13.00	1.79	14.50	9.76	no

The two partial variants keep only the replay pair construction or only the count and weight limits (category caps and weight clipping). Both have higher development means than the complete configuration. Removing weight clipping raises Wild error, and without the benign replay examples the training readout no longer improves on both roles. APW raises harmful refusal from 72.92% to 73.96% and benign non-refusal from 79.17% to 80.21% on the training readout.

Individual errors show where the gain comes from. On the Curated subset, the complete configuration corrects five harmful and one benign error made by at least one partial variant, without losing a prompt that both variants answer correctly. On Wild, it corrects four harmful errors but loses one harmful and one benign prompt that both variants answer correctly. On JBB, the error rates are identical (Appendix G).

In the bootstrap comparison on the 356 development prompts, APW has lower error than Hinge in 82.8% of the 10,000 draws and than NCA in 97.8%. The confidence interval of the difference lies above zero for NCA but includes zero for Hinge. The prompt-level corrections and full intervals are given in Appendix F.

6.2 Replay Coverage and Training Choices

Using fewer replay examples raises development error, even with the same number of updates. With all 649 examples, the development mean is 7.93%. A half-sized set gives 8.43%, a quarter-sized set

gives 9.43%, and retaining only half of the benign examples gives 8.10%. The smaller sets omit less frequent harmful topics and benign counterexamples. Category caps and weight clipping limit the influence of retained examples but cannot make up for missing ones.

We also vary the role coefficients, category cap, clipping threshold, and training length, using the Curated subset. APW obtains 8.00% error, compared with 9.00% for constant role coefficients, 10.00% with a looser 60% category cap, 10.00% with clipping at the 90th percentile, and 11.00% when training is doubled to 48 updates. With constant coefficients, the training readout still improves for both roles, but development error is higher than with adaptive weighting. A looser category cap or a lower clipping threshold improves only one of the two readout rates, whereas APW improves both.

6.3 Decoding and Refusal Scoring

With temperature 0.7 sampling, APW's development mean remains 1.02 points below APO-Zero's. Its harmful error is 2.53 points lower and its benign refusal 0.47 points higher. As with greedy decoding, the harmful reduction outweighs the benign increase.

The scoring comparison changes only how benign refusals are detected. Replacing WildGuard with an independent lexical detector, which looks for explicit refusal phrases, lowers APW's mean from 7.93% to 7.10%. The change comes from benign responses near the refusal boundary, which the two detectors label differently; the harmful labels stay the same.

Table 5: Generation and scoring choices for the component configuration. Sampling uses the same trained adapters; the detector comparison rescores the same APW responses. Values are error percentages.

Setting	Harm.	Benign	Mean ↓
Temperature 0.7 *decoding*			
APW	**13.52**	2.33	**7.93**
APO-Zero	16.05	**1.86**	8.95
Refusal scoring on the same APW responses			
WildGuard	14.19	1.67	7.93
Lexical	14.19	**0.00**	**7.10**

6.4 Training Seed Stability

Table 6 repeats APW training with seeds 42–49 in the 649-pair component configuration. The eight runs share the replay bank, training recipe, decoding, scoring, and 1,600 evaluation prompts, so differences come only from the training seed.

Overall error ranges from 16.95% to 17.97%, with a mean of 17.55% and a standard deviation of 0.31 points. Harmful and benign standard deviations are 0.62 and 0.48 points. The role errors show why Overall differs across seeds. Seed 42 has the lowest harmful error, 26.83%, but also the highest benign error, 8.04%. Seed 45 instead combines 27.41% harmful error with 6.48% benign error and obtains the lowest Overall score, 16.95%. Relative to seed 42, its 1.56-point reduction in benign refusal exceeds its 0.58-point increase in harmful error. Because changes in the two roles partly offset each other, Overall varies less than either role error.

Table 6: APW across eight training seeds in the 649-pair component configuration. Values are error percentages on the same 1,600 prompts; lower is better.

Training seed	Harm.	Benign	Overall
42	26.83	8.04	17.44
43	27.74	7.32	17.53
44	28.05	6.85	17.45
45	27.41	6.48	16.95
46	28.80	7.15	17.97
47	27.79	7.32	17.56
48	28.46	7.31	17.88
49	27.49	7.68	17.58
Mean	**27.82**	**7.27**	**17.55**
Std. dev.	0.62	0.48	0.31

7 Conclusion

APW adapts bounded role weights from smoothed losses without changing pair direction, training data, or inference. Category caps and weight clipping limit the influence of replayed examples. With Llama-3.1-8B, APW leads seven preference objectives in Overall (18.20%) and Ext. error, with harmful-error reductions outweighing increased benign refusal. Component studies show how replay coverage and example weights affect updates.

AI USE STATEMENT

AI tools assisted with drafting and polishing the manuscript and with retrieving and discovering related literature. The authors reviewed and verified all resulting text and references and take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

Section 3 specifies the training objective and adaptive weights, and Section 4 describes the evaluation. The appendices provide implementation details, data composition, and additional results. Anonymous training, baseline, evaluation, and aggregation code accompanies this submission.

REFERENCES

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022a. URL https://arxiv.org/abs/2204.05862.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional AI: Harmlessness from AI feedback, 2022b. URL https://arxiv.org/abs/2212.08073.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramer, Hamed Hassani, and Eric Wong. JailbreakBench: An open robustness benchmark for jailbreaking large language models, 2024. URL https://arxiv.org/abs/2404.01318.

Huayu Chen, Guande He, Lifan Yuan, Ganqu Cui, Hang Su, and Jun Zhu. Noise contrastive alignment of language models with explicit rewards, 2024. URL https://arxiv.org/abs/2402.05369.

Justin Cui, Wei-Lin Chiang, Ion Stoica, and Cho-Jui Hsieh. OR-Bench: An over-refusal benchmark for large language models, 2024. URL https://arxiv.org/abs/2405.20947.

Qiyuan Deng, Xuefeng Bai, Kehai Chen, Yaowei Wang, Liqiang Nie, and Min Zhang. Efficient safety alignment via preference re-ranking. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 31156–31171, 2025. doi: 10.18653/v1/2025.acl-long.1504. URL https://aclanthology.org/2025.acl-long.1504/.

Karel D'Oosterlinck, Winnie Xu, Chris Develder, Thomas Demeester, Amanpreet Singh, Christopher Potts, Douwe Kiela, and Shikib Mehri. Anchored preference optimization and contrastive revisions: Addressing underspecification in alignment. *Transactions of the Association for Computational Linguistics*, 13:442–460, 2025. URL https://aclanthology.org/2025.tacl-1.22/.

Aaron Grattafiori et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Rajdeep Haldar, Ziyi Wang, Guang Lin, Yue Xing, and Qifan Song. LLM safety alignment is divergence estimation in disguise. In *Advances in Neural Information Processing Systems*, volume 38, pp. 94010–94040, 2025. doi: 10.52202/085713-3145. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/8729770ef7e55bdb4a54ff1f55ead1e8-Paper-Conference.pdf.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. WildGuard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of LLMs, 2024. URL https://arxiv.org/abs/2406.18495.

Alif Al Hasan and Sumon Biswas. The refusal-compliance tradeoff: A large-scale safety behavior audit of large language models, 2026. URL https://arxiv.org/abs/2605.05427.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Xinmeng Huang, Shuo Li, Edgar Dobriban, Osbert Bastani, Hamed Hassani, and Dongsheng Ding. One-shot safety alignment for large language models via optimal dualization. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2405.19544.

Geon-Hyeong Kim, Yu Jin Kim, Byoungjip Kim, Honglak Lee, Kyunghoon Bae, Youngsoo Jang, and Moontae Lee. SafeDPO: A simple approach to direct preference optimization with enhanced safety, 2025. URL https://arxiv.org/abs/2505.20065.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J. Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization, 2023. URL https://arxiv.org/abs/2309.06657.

Yun Liu, Youyou Huang, Shouqiang Liu, and Xiongwen Pang. Evol-preference: Automatic evolution of preference data for safety alignment. *Expert Systems with Applications*, 331:133366, 2026. doi: 10.1016/j.eswa.2026.133366. URL https://doi.org/10.1016/j.eswa.2026.133366.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. 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 Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023. URL https://arxiv.org/abs/2305.18290.

Paul Rottger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics*, 2024. URL https://aclanthology.org/2024.naacl-long.301/.

Akifumi Wachi, Thien Q. Tran, Rei Sato, Takumi Tanabe, and Youhei Akimoto. Stepwise alignment for constrained language model policy optimization, 2024. URL https://arxiv.org/abs/2404.11049.

Yixu Wang, Yang Yao, Xin Wang, Yifeng Gao, Yan Teng, Xingjun Ma, and Yingchun Wang. Towards context-invariant safety alignment for large language models, 2026. URL https://arxiv.org/abs/2605.20994.

Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment, 2024. URL https://arxiv.org/abs/2405.00675.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Madhushani Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, Ruoxi Jia, Bo Li, Kai Li, Danqi Chen, Peter Henderson, and Prateek Mittal. SORRY-Bench: Systematically evaluating large language model safety refusal, 2024. URL https://arxiv.org/abs/2406.14598.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. Contrastive preference optimization: Pushing the boundaries of LLM performance in machine translation, 2024. URL https://arxiv.org/abs/2401.08417.

Tiankai Yang, Yi Nian, Xinyuan Li, Ruiyao Xu, Kaize Ding, and Yue Zhao. Cat-DPO: Category-adaptive safety alignment, 2026a. URL https://arxiv.org/abs/2604.17299.

Yonghui Yang, Wenjian Tao, Jilong Liu, Xingyu Zhu, Junfeng Fang, Weibiao Huang, Le Wu, Richang Hong, and Tat-Seng Chua. Revisiting robustness for LLM safety alignment via selective geometry control, 2026b. URL https://arxiv.org/abs/2602.07340.

Zhehao Zhang, Weijie Xu, Fanyou Wu, and Chandan K. Reddy. FalseReject: A resource for improving contextual safety and mitigating over-refusals in LLMs via structured reasoning, 2025. URL https://arxiv.org/abs/2505.08054.

Shiji Zhao, Mengyang Wang, Shukun Xiong, Fangzhou Chen, Qihui Zhu, Shouwei Ruan, Yisong Xiao, Ranjie Duan, Xun Chen, and XingXing Wei. Improving safety alignment via balanced direct preference optimization, 2026. URL https://arxiv.org/abs/2603.22829.

Xuandong Zhao, Will Cai, Tianneng Shi, David Huang, Licong Lin, Song Mei, and Dawn Song. Improving LLM safety alignment with dual-objective optimization, 2025. URL https://arxiv.org/abs/2503.03710.

A Additional Component Studies

A.1 Replay Bank Size and Training Choices

We vary the 649-pair component-study replay bank and one training choice at a time (Table 7). Replay-size comparisons average three development subsets; training-choice comparisons use the Curated subset. Error rises when examples are removed, even if category proportions are preserved. Without benign examples, the training readout no longer improves on both roles. The training-choice comparisons give higher error with constant role coefficients, a looser category cap, a lower clipping threshold, or more updates. In both parts of the table, the complete replay configuration has the lowest error; it keeps harmful and benign coverage while limiting the contribution of individual categories and examples.

Table 7: Replay size and training choices. Errors are percentages; "Both improve" means that harmful refusal and benign non-refusal on the training readout both rise, as in Table 4.

Choice	Error ↓	Both improve
Replay examples (three development sets)		
All 649 examples	**7.93**	**yes**
Half the benign examples	8.10	no
Half the examples	8.43	no
Quarter of the examples	9.43	yes
Training choices (Curated subset)		
APW	**8.00**	**yes**
Fixed multipliers	9.00	yes
Category limit: 60%	10.00	no
Weight clip: 90th percentile	10.00	no
48 updates	11.00	yes

A.2 Repeatability and Training Length

Repeating greedy decoding gives identical responses, and therefore the same error rates and the same ordering on the development set. Increasing training from 24 to 48 updates raises error on the Curated subset from 8.00% to 11.00% with the same replay examples. Repeated decoding only checks that responses reproduce, whereas longer training changes the adapter; with this replay configuration, extra updates raise development error.

B Implementation Details

B.1 Training Setup

All objectives start from Llama-3.1-8B-Instruct (Grattafiori et al., 2024) and read an ordered set of 5,333 preference pairs with the same optimizer, batch schedule, LoRA adaptation (Hu et al., 2021), and length settings (Table 8). Objectives that require a reference model use the initial model; CPO does not require one. APW adds separate loss targets and bounded coefficients for harmful and benign roles.

Each method processes the same examples and updates the same adapter modules for the same number of optimizer steps. Only the loss assigned to those examples differs. For APW, the initial model supplies reference scores and the training pairs supply the role feedback; no additional examples are generated. The controller, the rule that updates the two role coefficients, keeps two smoothed losses and two coefficients. A role whose smoothed loss exceeds its target receives a larger coefficient, up to the specified bound. The other role continues to follow its own loss trajectory. Because the controller reads losses already computed for the preference objective, it needs neither an extra generation pass nor a judge during training. Its state persists between minibatches, with each role updated only when that role is represented.

Experiments run on NVIDIA A100-SXM4 GPUs with 80 GB memory. The host uses two AMD EPYC 7742 processors, 2 TiB system memory, and Ubuntu 22.04.5. The software environment contains Python 3.11.13, PyTorch 2.8.0 with CUDA 12.8, Transformers 4.55.2, TRL 0.21.0, PEFT 0.17.0, Datasets 3.6.0, and Accelerate 1.10.1.

For assistant response $y = (y_1, \ldots, y_T)$, APW forms both policy and reference scores by averaging log probabilities over assistant tokens; prompt tokens and padding are masked. The difference between chosen and rejected scores is computed before the log-sigmoid in Eq. 4. Primary training

uses zero offsets and unit example weights. The adaptive role coefficients are not differentiated through.

Let $\bar{L}_{r,t}$ denote the exponentially smoothed loss for role r and B_r its target value. We update the role coefficient as

$$\bar{L}_{r,t} = \rho \bar{L}_{r,t-1} + (1-\rho) L_{r,t}, \tag{13}$$

$$\lambda_{r,t+1} = \Pi_{[a_r,b_r]}\big(\lambda_{r,t} + \eta_\lambda(\bar{L}_{r,t} - B_r)\big), \tag{14}$$

with $\eta_\lambda = 0.01$, $[a_h, b_h] = [1.0, 1.5]$, and $[a_b, b_b] = [0.7, 1.2]$. Projection is applied after each additive update. Smoothing the role losses reduces the effect of any single small batch. An absent role retains its previous state. In the primary run the harmful and benign coefficients end at 1.5 and 0.7, respectively; these are the final adaptive states, not constant coefficients used throughout training.

The ablations use 24 training steps on the 649-pair replay bank used for the component studies. This setting compares pair construction, example weights, and whether the role coefficients are updated, with the same number of updates for every variant. The primary objective comparison uses the 120-step setup in Table 8. All variants in Tables 4 and 7 use the 24-step schedule unless training length is the variable under study.

Table 8: Common training settings, reference models, and APW role coefficients. All objectives use the same data and adapter budget. CPO is reference-free.

Setting	Value
Common training setup	
Initialization	Llama-3.1-8B-Instruct
Training pairs	5,333
Training seed	42
Updates	120
Batch / accumulation	1 / 8
Learning rate	5×10^{-6}
β	0.10
LoRA rank / α	16 / 32
LoRA dropout	0.05
LoRA targets	q, k, v, o projections
Token limit (prompt / answer)	256 / 256
Objective-specific reference	
Reference-based methods	Initial Llama model
CPO	Reference-free
APW role balancing	
Target loss (harm. / benign)	0.58 / 0.78
Harmful coefficient	[1.0, 1.5]
Benign coefficient	[0.7, 1.2]
Step size / smoothing	0.01 / 0.90
Example weight / offset	1 / 0

B.2 Preference Objectives

All methods receive the same prompts, chosen responses, and rejected responses, so every objective is shown the same desired behavior. The comparison therefore tests how each update rule learns from that behavior. APW scores each response by its mean token log probability and adapts the role coefficients; Table 9 gives the seven baseline losses.

For the baselines, let q_m^+ and q_m^- be the summed log probabilities that model m assigns to the retained assistant tokens of the chosen and rejected responses. Define the log ratios $a = q_\theta^+ - q_{\text{ref}}^+$ and $b = q_\theta^- - q_{\text{ref}}^-$, where the reference is the initial model. They measure how far the policy has moved the chosen and rejected log probabilities away from the initial model. Unlike APW's mean scores in Eq. 2, these scores are sums. Write σ for the logistic function and $\mathcal{L}_{\text{NLL}}^+$ for the mean negative log probability of the chosen response's assistant tokens.

Table 9: Losses used for the seven preference baselines. All use $\beta = 0.1$ and zero label smoothing. CPO has no reference model and adds the chosen-response likelihood term with coefficient one.

Method	Loss per response pair
DPO	$-\log \sigma(\beta(a-b))$
Hinge	$\max\{0, 1-\beta(a-b)\}$
APO-Zero	$1-\sigma(\beta a)+\sigma(\beta b)$
APO-Down	$\sigma(\beta a)+1-\sigma(\beta(a-b))$
SPPO	$(a-1/(2\beta))^2+(b+1/(2\beta))^2$
NCA	$-\log\sigma(\beta a)-\frac{1}{2}\log\sigma(-\beta a)-\frac{1}{2}\log\sigma(-\beta b)$
CPO	$-\log\sigma\big(\beta(q_\theta^+-q_\theta^-)\big)+\mathcal{L}_{\text{NLL}}^+$

DPO raises the chosen log ratio relative to the rejected one. Hinge stops penalizing a pair once $a-b$ reaches $1/\beta = 10$. APO-Zero increases the chosen likelihood and decreases the rejected likelihood;

APO-Down lowers both, with a larger decrease for the rejected response. SPPO uses hard preference targets and pulls the log ratios toward $a = 5$ and $b = -5$. NCA penalizes the chosen and rejected log ratios separately instead of through their difference. CPO adds a likelihood term for the chosen response to a pair loss computed from the policy alone. These losses can therefore give different gradients even though all methods see the same ordered pairs.

B.3 Baseline Implementations

Prompt tokens, padding, and tokens beyond the completion limit are excluded from every response score. APW averages the remaining assistant-token log probabilities according to Eq. 2. The other objectives retain their published score aggregation and loss transformation while using the same assistant-token mask. For reference-based objectives, the reference score is computed by the initial Llama model with gradients disabled, and policy and reference use the same tokenizer and chat template.

We implement the baselines with TRL 0.21.0. DPO, Hinge, APO-Zero, APO-Down, SPPO, and NCA use `DPOTrainer` with the `sigmoid`, `hinge`, `apo_zero`, `apo_down`, `sppo_hard`, and `nca_pair` losses, respectively. For SPPO, the chosen and rejected responses of each pair serve as winner and loser; no additional preference model scores them. CPO uses `CPOTrainer` with its sigmoid loss and $\alpha_{\text{CPO}} = 1$. All methods share the adapter capacity, learning rate, batch schedule, and update count in Table 8.

The implementation first masks tokens, then aggregates response scores, and finally applies the chosen objective. With the steps kept apart, each loss receives scores for the intended assistant response. APW takes the mean log probability over retained tokens; baselines use their published aggregation. Gradients flow through the policy scores into the LoRA modules, but not through reference scores. A reference forward pass adds a score to the calculation without adding trainable parameters.

With gradient accumulation, each optimizer update receives the same eight pairs in the same order for every objective. APW can change a role coefficient between those pairs because it updates the coefficients after every minibatch. The adapter itself changes only at the optimizer step. Response scoring, the loss, and these coefficients determine each pair's gradient contribution throughout the 120 updates in the primary comparison.

B.4 Response Pairs for the Primary Comparison

The responses in the 5,333 training pairs are generated. Llama-3.1-8B-Instruct writes candidate answers under instructions that ask for safe, helpful, cautious, direct, or ordinary responses. For harmful prompts, the direct and ordinary candidates come from Llama-2-7B instead. Each generation pass samples four answers per instruction with temperature 0.9, top-p 0.9, and a 192-token limit. These instructions are used only to generate candidates; each training pair contains the user prompt and the two selected responses.

For a harmful prompt, Llama-Guard-3-8B scores the candidate answers. The chosen response is the safe candidate with the lowest unsafe score, and the rejected response is the unsafe candidate with the highest unsafe score. For a benign prompt, WildGuard separates non-refusals from refusals. The chosen response is the longest non-refusal, with lower refusal score breaking ties; the rejected response is the refusal with the highest refusal score. We keep a pair only when both responses exist and differ. Each pair stores the candidate scores and the instruction type that produced each response. All 3,199 harmful and 2,134 benign chosen responses meet their selection rule. After removing six samples containing leftover dialogue markers, the pool contains 5,333 distinct prompts, and every objective receives the same resulting pairs.

B.5 Replay Bank for Component Studies

We draw harmful candidates from WildJailbreak training prompts and harmful preference examples. The initial model produces one greedy answer and two sampled answers with temperature 0.8, top-p 0.95, and a 64-token limit. A candidate must contain at least six words, be marked as a non-refusal by both WildGuard and a lexical detector, and be classified as unsafe by Llama-Guard-3-8B. Both

model thresholds are 0.5. For benign candidates, we remove prompts containing risk terms and retain pairs in which the chosen answer is not a refusal but the rejected answer is.

Table 10 summarizes the 649-pair component-study replay bank. Harmful chosen responses are overwhelmingly refusal-like and harmful rejected responses are almost always non-refusals; benign examples show the opposite pattern. The largest harmful category remains below 50%, and no evaluation prompt appears in this replay bank.

Table 10: Response direction in the replay bank used for component studies.

Measure	Harmful pairs	Benign pairs
Examples	403	246
Chosen refusal (%)	98.51	0.00
Rejected refusal (%)	0.99	97.97
Eval. overlap	0	0
Largest category (%)	49.13	n/a

The refusal patterns in the table check the intended direction of these pairs. For a harmful prompt, the chosen response refuses or redirects safely and the rejected response gives actionable assistance. For a benign prompt, the chosen response answers the request and the rejected response refuses. Refusal is therefore rewarded on harmful prompts and penalized on benign ones, as fixed by the prompt's role before the loss is computed. Category selection can remove an example and weighting can change its contribution, but neither swaps its chosen and rejected responses.

B.6 Training Batches

Pairs are shuffled once with seed 42 and consumed with micro-batch size one and gradient accumulation eight. Each optimizer step therefore accumulates gradients from eight ordered preference pairs. For every minibatch containing role r, APW computes the role loss, updates its exponential moving average, and applies the bounded coefficient update before weighting that minibatch loss. Role coefficients, and the additional example weights used in component studies, are not differentiated through.

The prompt and completion limits in Table 8 are applied before scoring. APW's completion limit and score both include the appended end-of-sequence token. All retained assistant tokens contribute; padding is masked, and no additional length penalty is applied. Training statistics are accumulated in full precision even when model operations use the lower-precision dtype. Token masking and batch formation are common across methods; response-score aggregation and the loss follow each objective's implementation. APW additionally updates its role coefficients.

B.7 Ablation Details

Variants in Tables 4 and 7 train on the component-study replay bank. Table 4 and the replay-size comparisons in Table 7 use all three development subsets; the training-choice comparisons use the 100-prompt Curated subset. The training readout, harmful refusal and benign non-refusal measured on 192 training examples (96 per role), shows whether both behaviors improve.

Keeping the role coefficients fixed, longer training, and reduced benign coverage all increase error. Without benign examples, harmful refusal and benign non-refusal no longer both improve. The training readout records the two changes separately, so a gain on one role cannot mask a loss on the other. Improving both training behaviors still does not guarantee lower development error; constant coefficients improve both roles but give a higher development error than adaptive weighting in this configuration (Appendix H).

C Evaluation Details

C.1 Metrics

Every prompt contributes one binary error label for its role. A harmful response is an error when the safety classifier identifies unsafe assistance. A benign response is an error when the refusal classifier identifies a refusal. Source-level ASR and OR are computed from their respective denominators; harmful and benign rates are macro-averaged over three and four sources, respectively, before receiving equal weight in Overall. Generation length, classifier confidence, and training loss do not enter the score.

Table 11 reports the numerator and denominator for each APW source rate. These integers yield harmful error $(27/100 + 46/170 + 173/530)/3 = 28.90\%$ and benign error $(1/100 + 15/300 + 57/274 + 4/126)/4 = 7.49\%$. Their mean is the 18.20% Overall error.

The denominators in the table apply to individual source rates. They do not set the weight of each source in Overall. Pooling all 1,600 binary labels would give the largest Wild subsets the greatest influence; averaging the seven source rates directly would instead give more weight to the benign role, which has four sources rather than three. We average within each role first to avoid both effects. Harm. measures unsafe assistance across the three harmful sources, and Benign measures unnecessary refusal across the four benign sources. Overall gives these two rates equal weight. A source can therefore supply more prompts, and a more precise estimate of its own error rate, without receiving a larger coefficient in the final score.

Table 11: APW errors and prompt counts by source. FR denotes FalseReject.

	Harmful		Benign	
Source	Prompts	Errors	Prompts	Errors
JBB	100	27	100	1
SORRY	170	46	n/a	n/a
OR-Bench	n/a	n/a	300	15
FR	n/a	n/a	274	57
Wild	530	173	126	4

C.2 Evaluation Data

Each method generates one response for each of the 1,600 distinct prompts across seven source and role groups. The 5,333 pairs used in the primary comparison and the 649-pair replay bank used for component studies are both disjoint from this evaluation set.

We use the per-prompt predictions for source breakdowns and comparisons of individual decisions. Applying the same aggregation rule gives the primary values and the source-level results in Table 12.

C.3 Generation and Scoring

All adapters use the same prompt order, chat template, loading procedure, dtype, greedy decoder, 64-token limit, classifier thresholds, judge prompt, and batch policy. Evaluation seeds 44, 45, 46, and 47 produce identical responses under greedy decoding. The 600 extension prompts are marked so that Ext. can be computed separately, while Harm., Benign, and Overall use all 1,600 predictions.

C.4 Classifier Application

The harmful classifier receives the user prompt and generated assistant response and predicts whether the response contains unsafe assistance. Safe refusals and high-level safety information are counted as correct; actionable harmful guidance is counted as an error. The benign classifier operates on the assistant response and predicts refusal. A direct, informative answer is correct even when it includes a brief safety qualification, whereas declining the benign request is an error. A safety qualification alone does not count as a refusal. The benign label records whether the model declines to answer, and the harmful label records the assistance actually given.

Classification is performed after generation and does not modify the response. Each prompt receives exactly one role-appropriate binary label: harmful examples use the safety label, and benign examples use the refusal label. Replacing the benign classifier with a lexical refusal detector, while keeping generated responses and harmful labels unchanged, shows how much the results depend on the benign refusal detector (Appendix H).

C.5 Aggregation

The calculation first forms one error rate per source. It then macro-averages the three harmful rates and the four benign rates:

$$\text{Harm.} = \frac{e_{\text{JBB}-\text{H}} + e_{\text{SORRY}-\text{H}} + e_{\text{Wild}-\text{H}}}{3}, \tag{15}$$

$$\text{Benign} = \frac{e_{\text{JBB}-\text{B}} + e_{\text{OR}-\text{B}} + e_{\text{FR}-\text{B}} + e_{\text{Wild}-\text{B}}}{4}. \tag{16}$$

Overall is their equally weighted mean. The 530-prompt Wild harmful block and the 100-prompt JBB harmful block thus have equal weight, despite the 5.3-fold difference in size. The 274-prompt FalseReject block likewise contributes one benign source rate. Prompt counts enter the denominators of the source rates, not the coefficients used to combine them.

The sources differ in how they collect prompts as well as in size. Some target adversarial attacks, some cover a broader range of harmful requests, and others collect answerable questions that often trigger refusal. Combining all their labels before computing rates would make the result depend heavily on how many prompts were collected by each source. Averaging source rates instead measures how a method performs across these collections. The final mean then weighs unsafe assistance against unnecessary refusal, so a gain on one large harmful collection cannot by itself outweigh changes across the benign collections.

Refusal counts alone would not recover these metrics. A harmful answer can be safe without explicitly refusing, for example by giving non-operational background or redirecting the request. Conversely, a benign answer can include a caution while still answering the question. The harmful label must identify unsafe assistance, and the benign label must identify a declined request. Using a common refusal rule would count some safe harmful responses as failures and some unnecessary benign refusals as successes. Role-specific labels avoid this.

Two methods with similar Overall errors can still get different prompts right, for instance when one avoids more harmful assistance but refuses more benign requests. Their source rates show where these changes occur. Comparing individual answers within a source then identifies which errors are shared and which are corrected by only one method.

Ext. restricts the calculation to the 600-prompt extension. Harm., Benign, and Overall use all 1,600 prompts, including that subset. Reporting Ext. separately lets us examine the added prompts while retaining the same seven source terms and weights in the complete evaluation.

C.6 Score Precision

We compute source rates from error counts, form the harmful and benign macro-averages, and round only the displayed values. Tables show percentages to two decimal places, so a displayed Overall value can differ by 0.01 points from an average of the rounded Harm. and Benign entries. Rankings and differences use unrounded values.

Ext. is calculated independently from its 300 harmful and 300 benign prompts. Its benign half contains three sources, which are macro-averaged before being combined with the harmful rate. Thus Ext. and Overall use the same role-balanced principle but different prompt compositions. Source counts and denominators in Table 11 provide the corresponding calculation for the 1,600-prompt result.

C.7 Role and Source Aggregation

Overall uses every source and role group; Ext. recomputes the score on the added prompts alone. An answer from the extension appears in both summaries, but enters Overall only through its source rate. Ext. is reported separately to show whether the ordering persists on this part of the evaluation.

Source and role labels determine which answers enter each rate. A source that contains both harmful and benign requests contributes to two separate groups; its labels are never pooled across roles. The harmful judge assesses unsafe assistance in one group, and the benign judge assesses refusal in the other. The two roles are combined only after the source rates are computed, so an error means the same thing throughout the calculation.

The source breakdown shows whether a change spans several datasets or comes from one, and equal source weights keep a large collection from dominating. Reporting the constituent rates alongside the aggregate shows where each method improves or loses accuracy; individual prompt comparisons then identify the answers responsible for those changes.

C.8 Individual Corrections

Source rates identify which prompt groups contribute to an aggregate difference and whether it occurs on the harmful or benign side (Appendix D). Within each group, shared correct and incorrect decisions cancel from the error difference. Only prompts corrected by one method and missed by the other contribute to the difference and its bootstrap variation (Appendix F).

D Results by Evaluation Source

We examine the 600-prompt extension first, then the seven source and prompt-type groups that make up the full evaluation. The extension changes the source mixture; the source breakdown identifies which harmful answers and benign refusals account for the aggregate differences.

D.1 Extension Results

The extension contains 300 harmful Wild prompts and 300 benign prompts drawn from OR-Bench, FalseReject, and Wild. Ext. is the equally weighted mean of harmful and benign errors, so a low error on one role cannot decide the result alone.

APW obtains 30.00% harmful error and 10.59% benign error, producing 20.29% Ext. APO-Down reaches 7.52% benign error but 36.67% harmful error. Averaging the harmful and benign extension errors therefore gives APW the lowest Ext. score.

The nearest extension result is NCA at 20.42%. Its benign error is 2.76 points below APW, but its harmful error is 3.00 points higher. DPO follows a similar pattern with 33.67% harmful and 8.10% benign error. Hinge and APO-Zero move farther toward benign performance and have larger harmful errors. In each case the harmful increase exceeds the benign reduction, leaving the combined error above APW's.

In this subset all harmful requests come from Wild and the benign half excludes JBB, yet APW still has the lowest harmful and combined errors, so its extension result does not rely on the JBB advantage seen in the complete evaluation. The benign cost remains, but is smaller than the harmful reduction against each comparator.

D.2 Results by Source and Prompt Type

Overall is computed from three harmful and four benign source rates (Table 12). Differences within these groups show which errors drive each method's aggregate.

Table 12: Error rates by source and prompt type. Bold marks column minima, including ties. FR denotes FalseReject.

	Harmful error (%)			Benign error (%)			
Method	JBB	SORRY	Wild	JBB	OR	FR	Wild
APW	**27.00**	27.06	**32.64**	1.00	5.00	20.80	3.17
APO-Zero	33.00	22.94	34.72	1.00	4.00	17.52	2.38
NCA	32.00	28.24	**32.64**	**0.00**	3.67	**15.69**	3.17
Hinge	32.00	24.71	33.96	**0.00**	3.67	17.88	3.97
DPO	32.00	25.29	34.91	**0.00**	**3.33**	18.61	**1.59**
SPPO	33.00	**22.35**	36.23	1.00	3.67	17.88	2.38
CPO	31.00	27.06	33.58	1.00	3.67	18.25	3.17
APO-Down	40.00	28.24	34.53	**0.00**	**3.33**	16.79	3.17

APW obtains 27.00%, 27.06%, and 32.64% error on the three harmful sources. Its four benign rates are 1.00%, 5.00%, 20.80%, and 3.17%. Macro-averaging these source rates yields 28.90% Harm. and 7.49% Benign, whose mean is the 18.20% Overall result. The larger Wild blocks therefore cannot dominate either role through their size.

Within-source comparisons show differences that Overall hides. NCA matches APW on harmful Wild but makes more errors on JBB and SORRY. CPO matches APW on harmful SORRY but makes more errors on JBB and Wild. SPPO's SORRY advantage comes with increases on both other harmful sources. On the benign side, NCA and DPO obtain lower rates on several collections. APW's combined advantage comes from the larger harmful reduction after these source-specific changes are averaged across both roles.

Among the harmful source minima, APW has 27.00% on JBB, SPPO has 22.35% on SORRY, and APW and NCA share 32.64% on Wild. JBB matters most here, since several objectives are

six percentage points worse than APW on it. APW combines its JBB lead with the shared Wild minimum and avoids the higher errors on those sources that offset SPPO's SORRY result.

The benign side looks different. NCA obtains the lowest FalseReject rate, DPO obtains the lowest Wild rate and shares the lowest OR-Bench rate, and four methods make no error on benign JBB. APW is not lowest on any single benign source, but its rates stay moderate on all four. APO-Down shows the reverse, with low benign rates and a much higher harmful JBB error.

To recover Overall from the table, we average the harmful columns and the benign columns separately, then take their mean. A lower error in any one column contributes according to that column's role, regardless of its prompt count. The largest number of corrected Wild prompts need not give the largest Overall reduction. For this reason we explain the aggregate with the source table and report raw correction counts separately.

D.3 Harmful and Benign Trade-offs

The four primary measures give the direction and size of each comparison. APO-Zero is the closest objective at 0.02 points higher Overall. Its 1.27-point advantage on Benign is outweighed by its 1.32-point increase on Harm. Its Ext. error is also 1.18 points higher.

NCA, Hinge, DPO, SPPO, CPO, and APO-Down follow the same pattern: lower Benign error but higher Ext., Harm., and Overall. APW's advantage comes from reducing harmful error by more than it increases benign refusal. This holds after macro-averaging the seven prompt groups, which weights each source equally regardless of size.

E Source-Level Comparison

Figure 4 decomposes the primary result across the seven source and role groups. Positive cells indicate lower error for APW. APW improves Overall and Ext. against every comparator, while the source columns show where each aggregate gain originates.

	JBB-H	SORRY-H	Wild-H	JBB-B	OR-B	FR-B	Wild-B	Overall	Ext.
APO-Zero	+6.0	-4.1	+2.1	+0.0	-1.0	-3.3	-0.8	**+0.02**	**+1.18**
NCA	+5.0	+1.2	+0.0	-1.0	-1.3	-5.1	+0.0	**+0.10**	**+0.12**
Hinge	+5.0	-2.4	+1.3	-1.0	-1.3	-2.9	+0.8	**+0.10**	**+0.79**
DPO	+5.0	-1.8	+2.3	-1.0	-1.7	-2.2	-1.6	**+0.11**	**+0.59**
SPPO	+6.0	-4.7	+3.6	+0.0	-1.3	-2.9	-0.8	**+0.18**	**+2.10**
CPO	+4.0	+0.0	+0.9	+0.0	-1.3	-2.6	+0.0	**+0.34**	**+1.29**
APO-Down	+13.0	+1.2	+1.9	-1.0	-1.7	-4.0	+0.0	**+1.84**	**+1.80**



Figure 4: Source-level error differences relative to APW. Each cell reports comparator error minus APW error in percentage points. Positive values favor APW; Overall and Ext. are the aggregate columns.

F Statistics and Robustness

F.1 Prompt-Level Comparison

The component-study adapters for Hinge, NCA, and APW have predictions on the same 356-prompt development set. To characterize their differences, we compute

$$\Delta_m = \mathrm{Bad}_{\mathrm{control}}(m) - \mathrm{Bad}_{\mathrm{control}}(\mathrm{APW}). \tag{17}$$

Here $\mathrm{Bad}_{\mathrm{control}}$ is the development mean, the average of the role-balanced errors on the three development subsets. Positive differences favor APW. For each of 10,000 bootstrap draws, we resample

prompts separately within six combinations of source and prompt type and use the same sampled prompts for APW and comparator m.

Table 13 reports the differences and confidence intervals. Table 14 summarizes the prompts that determine them.

Table 13: Development-set error differences. Difference is comparator error minus APW error; differences and 95% confidence intervals are in percentage points. APW lower is the percentage of 10,000 bootstrap draws with a positive difference.

Objective	Difference	95% CI	APW lower (%)
Hinge	+0.83	[−0.83, 2.50]	82.8
NCA	+1.67	**[0.17, 3.33]**	**97.8**

NCA makes 12 errors that APW corrects, while APW makes five errors that NCA corrects, giving seven net corrections for APW and a 95% interval above zero. Hinge has ten errors corrected only by APW and six corrected only by Hinge; its point estimate favors APW, but the interval crosses zero. Both methods answer more than 300 prompts correctly. The remaining cases, where only one method is correct, decide the difference, and the bootstrap keeps each prompt's outcomes for both methods together.

Within each source and role group, prompts that both methods answer correctly or both answer incorrectly cancel from the difference, leaving only the cases on which their decisions diverge. Resampling the same prompt for both methods preserves this cancellation in every bootstrap draw. Stratifying by source and role keeps the number of sampled prompts in each group unchanged. A draw can include a prompt on which the methods disagree more or less often, but a prompt always keeps its role and source. The interval measures how the error difference varies when these observed cases are resampled.

Table 14: Corrections on development prompts. Net is the APW-only count minus the comparator-only count.

	Exclusive corrections		
Method	APW	Other	Net
Hinge	10	6	+4
NCA	**12**	5	**+7**

F.2 Metric-Level Comparison

The four primary measures show how each objective arrives at its Overall score. APW is lower on Ext., Harm., and Overall than APO-Zero, NCA, Hinge, DPO, SPPO, CPO, and APO-Down; those methods are lower on Benign.

APO-Zero is nearest: its lower Benign error is offset by higher Harm., leaving Overall 0.02 points higher; its Ext. error is also higher. The remaining objectives follow the same pattern. Across all seven comparisons, APW has the lower extension and Overall error.

The sign of each role's difference explains the small Overall gap. Reducing harmful compliance moves the mean in APW's favor; reducing benign refusal moves it toward the comparator. These changes partly cancel when the two roles are averaged, so we report both rates to keep the behavioral differences visible when combined scores are close.

F.3 Changing Category Labels

This analysis changes only category labels in the component-study replay bank. At each noise level, we keep the examples, harmful or benign labels, initial offsets, and model responses unchanged. We randomly replace the specified fraction of category labels, then recompute the category offsets and the 50% category cap. Table 15 reports the change in the weighted mean offset and the fraction of examples whose offset decreases.

Table 15: Sensitivity to category-label changes.

Relabeled (%)	Mean offset change	Lower offset (%)	Cap exceeded
0	0.0000	0.0	No
5	0.0006	4.7	No
10	0.0012	9.8	No
20	0.0025	19.1	No
30	0.0038	28.9	No
40	0.0050	38.6	No

When 40% of labels are replaced, the weighted average offset changes by 0.005, 38.6% of examples receive a smaller offset, and no category ex-

ceeds the 50% cap. The average thus barely moves even though many individual examples receive a different offset.

G Component-Study Replay Bank and Ablations

G.1 Examples and Weights by Category

The replay bank used for component studies contains 649 examples with total weight 965.66. The category cap applies to the number of examples, while training uses their weights. Table 16 reports the corresponding shares; Table 17 adds the counts, category offsets, and mean example weights.

Table 16: Examples and training weight in the component-study replay bank.

Category	Examples (%)	Weight (%)
Benign trigger	37.9	23.7
General	30.5	37.5
Violence / self-harm	11.7	14.7
Cyber / privacy	10.3	12.3
Fraud / crime	4.3	4.7
Civic / finance	3.1	4.2
Weapons / explosives	1.5	1.9
Bio / chemical	0.5	0.7
Infrastructure	0.2	0.3

Table 17: Category offsets and mean example weights.

Category	n	Offset	Weight
Benign trigger	246	0.000	0.931
General	198	0.035	1.830
Violence/self-harm	76	0.050	1.865
Cyber/privacy	67	0.060	1.770
Fraud/crime	28	0.045	1.611
Civic/finance	20	0.040	2.005
Weapons/explosives	10	0.055	1.870
Bio/chemical	3	0.055	2.385
Infrastructure	1	0.045	3.000

General safety accounts for 30.5% of examples but 37.5% of total weight. Benign triggers, the category of benign examples, account for 37.9% of examples and 23.7% of total weight. No other harmful category exceeds 14.7% of the weight. This configuration gives more weight to difficult harmful examples while keeping the largest category below half of the total example weight.

G.2 Sources of Replay Examples

The component-study replay bank combines existing training pairs with errors produced by the current model. Table 18 shows the category counts for each source of examples.

Current harmful non-refusals contribute 111 examples, including 60 general-safety, 19 violence/self-harm, and 15 cyber/privacy examples. Current unsafe answers add 62 examples, and selected and earlier harmful failures add 88 and 30. The Existing pairs and Harmful pairs rows together provide 299 training examples (251 and 48, respectively), while 59 benign refusal examples preserve coverage of benign requests.

Existing pairs provide most benign triggers and cover several harmful categories. Current-model errors add prompts on which the adapter still gives unsafe answers or fails to refuse. Earlier and selected failures retain cases that may not recur in the latest generation. Keeping them means a difficult example is not dropped just because it was missing from the latest sample, while current errors point training at failures that persist.

Table 18: Replay sources cover different categories.

Source	Benign trigger	General	Violence	Cyber	Fraud	Civic	Weapon	Bio	Infra.	Harm. total
Existing pairs	187	31	13	11	5	2	2	0	0	64
Current non-refusals	0	60	19	15	6	7	2	2	0	111
Selected failures	0	41	15	19	4	5	3	1	0	88
Unsafe answers	0	30	14	9	3	3	2	0	1	62
Benign refusals	59	0	0	0	0	0	0	0	0	0
Harmful pairs	0	22	10	8	7	1	0	0	0	48
Earlier failures	0	14	5	5	3	2	1	0	0	30

The table also explains why the category cap is applied after sources are merged. General safety appears in several harmful sources. Capping each source independently would still allow that topic to accumulate across the combined bank. We instead count its retained examples across sources before applying the cap. Benign refusals provide the opposite response direction, and existing pairs preserve categories that occur less often among current errors. The resulting bank contains recent failures and broader coverage, and no frequently observed topic dominates by count.

G.3 Weight Distribution

Capping weights at the 95th percentile and clipping them to $[0.20, 3.00]$ limits the upper tail of the example weights. The maximum weight is 3.0. The effective number of examples is 476.9; it equals the raw count when all weights are equal and falls as weight concentrates on fewer examples. The top 1%, 5%, and 10% of examples carry 1.86%, 9.94%, and 19.88% of the total weight. Table 19 reports these weights for each source.

Table 19: Replay weight distribution by source.

Source	Median	90th pct.	Effective n
Existing pairs	0.977	0.977	250.5
Current non-refusals	2.883	3.000	111.0
Selected failures	1.050	1.176	87.6
Unsafe answers	3.000	3.000	62.0
Benign refusals	0.782	0.782	59.0
Harmful pairs	0.786	0.847	47.8
Earlier failures	1.155	1.294	29.9

Current unsafe answers and harmful non-refusals receive the largest weights, but their effective counts remain 62.0 and 111.0. The 251 existing harmful/benign pairs have an effective count of 250.5. Weighting thus shifts influence between sources without concentrating it on a few examples.

A source with many examples could still contribute through only a few of them if their weights were much larger than the rest. Its effective count would then be far below its raw count. Here the main sources have effective counts close to their raw counts even though their average weights differ, so the larger weights for current harmful failures are spread across those failures. The upper-tail shares give a second check, measuring how much weight the largest examples carry after clipping, independently of category frequency.

G.4 Number of Replay Examples

We vary replay size under the same 24 updates, keeping all replay examples disjoint from evaluation prompts. Development error measures generalization (Table 7), while category coverage and role changes characterize which examples remain and how the update affects each behavior (Table 20).

Table 20: Replay size, category coverage, and training behavior.

Examples	Largest (%)	Harm. Δ	Benign Δ
All (649)	49.1	+1.04	+1.04
25% (172)	46.4	+1.04	+1.61
50% (330)	48.1	+2.08	−1.04
Half benign (527)	49.1	0.00	0.00

The 25% subset improves both behaviors but raises development error by 1.5 points. The 50% subset worsens benign acceptance and raises error by 0.5 points. Keeping only half of the benign examples raises error by 0.2 points and improves neither behavior. Only the complete 649-pair bank combines the lowest error with improvement in both.

The smaller banks omit less frequent harmful categories and benign counterexamples. The full bank retains these cases, while the category cap restricts how often the largest harmful group appears. Every row uses the same number of optimizer steps, so the lower development error with the full bank does not come from a longer schedule. Category caps and weight clipping act only on the examples that remain and cannot restore an error type removed by subsampling. In Table 20, Largest is the largest harmful category's share. Δ measures changes from the parent model in percentage points; increased harmful refusal and benign non-refusal are improvements.

G.5 Errors Corrected by APW

Aggregate results do not show whether pair construction and example weighting correct the same prompts. Table 21 compares a model using only the replay pair construction, a model using only

the count and weight limits (category caps and weight clipping), and APW. The Gains column counts prompts missed by at least one ablation but answered correctly by APW. Losses are prompts answered correctly by both ablations but incorrectly by APW.

Table 21: Component errors and APW corrections. Pairs, Weights, and APW report error percentages; Gains and Losses report numbers of prompts.

Subset	Pairs	Weights	APW	Gains	Losses
Curated harm.	22.0	20.0	**14.0**	5	0
Curated benign	4.0	4.0	**2.0**	1	0
JBB harm.	3.6	3.6	3.6	0	0
JBB benign	0.0	0.0	0.0	0	0
Wild harm.	27.0	**25.0**	**25.0**	4	1
Wild benign	**2.0**	**2.0**	3.0	0	1

APW corrects six Curated-subset errors made by at least one single-change ablation and introduces no new error on this set. On Wild prompts it corrects four harmful errors but introduces one harmful and one benign error; JBB is unchanged. The aggregate gain can thus be traced to specific prompts.

The two ablations leave partially different errors. APW resolves some of each, so the complete configuration improves on both without changing every prediction.

H Additional Analyses

H.1 Sampling

We evaluate the component-study adapters with temperature 0.7, top-p 0.90, and top-k 50, while retaining the 64-token limit and the same trained adapters (Table 5). This tests whether the development-set ordering holds under stochastic decoding.

APW scores 7.93%, compared with 8.95% for APO-Zero. APW has lower harmful ASR, 13.52% versus 16.05%, while APO-Zero has lower benign refusal, 1.86% versus 2.33%. As in the primary result, the harmful-error reduction outweighs the benign difference.

The source results explain the sampled-decoding mean. APO-Zero is lower on Curated, but APW is lower on JBB and Wild. Across these sources, APW's harmful reduction exceeds its increase in benign refusal, as under greedy decoding. Since only the response sampling differs, APW's lower mean does not depend on greedy decoding.

H.2 Alternative Refusal Detector

We rescore the same generated responses with an independent lexical refusal detector in place of WildGuard for benign examples, keeping harmful labels unchanged (Table 5).

Development error changes from 7.93% with WildGuard to 7.10% with lexical labels. Harmful ASR remains 14.19% because those labels are not changed. The lower mean comes entirely from benign answers whose refusal labels differ between detectors. Both scores are low; the lexical rule simply counts fewer of these answers as refusals.

The lexical detector looks for explicit refusal phrases. WildGuard can also recognize wording that implies the request is being declined, so the detectors can disagree on answers near the refusal boundary. Rescoring the same answers shows how much the reported benign error depends on this decision rule; the generated text and the harmful labels stay the same.

H.3 Hyperparameter Sensitivity

We change one training choice at a time on the same Curated prompts: constant or updated role coefficients, the category cap, the clipping percentile, and the number of updates (Table 7).

Keeping the role coefficients constant raises development error while still improving both training roles. A looser category cap and a lower clipping threshold improve only one of the two readout rates. In the component configuration, adaptive coefficients with moderate replay controls work best.

The training readout and the development set measure different things. Constant coefficients and longer training can improve both training roles, yet yield higher development error. The training readout checks that an update moves both behaviors as intended, and held-out prompts show which adapter generalizes better. The complete replay configuration passes the first check and has the lowest development error.

H.4 Scope and Extensions

The evaluation uses short, single-turn English prompts. In dialogue, a high-level answer may become actionable after later context, and long responses may contain both safe explanation and unsafe detail. Extending APW to these settings requires preference pairs over conversation states or response segments rather than one comparison for the entire exchange.

Harmful and benign errors are identified by model-based classifiers, with an additional lexical analysis for benign refusal. Human judgments could better distinguish concise but adequate answers from non-refusing yet unhelpful ones, and safe educational context from borderline operational detail. Such labels would refine both evaluation and pair construction.

The category names are broad, and rare categories contain little evidence. Finer topics, overlapping labels, or continuous similarity could represent mixed requests more accurately. They would still need to limit how much weight any narrow group of prompts receives, so that no such group controls the update. Groups of prompts that ask for the same thing in different words would also test whether improvements in both roles persist when only the wording changes.

The development-set confidence intervals cover Hinge and NCA. The primary harmful evaluation instead provides individual correction counts for all seven objectives, while the eight-seed study summarizes APW's optimization variation in the replay-bank setting. Each of these analyses covers a different source of variation.

H.5 Reproducibility

Training starts from the common initial model and ordered preference pairs; evaluation loads the resulting adapters and records one prediction per prompt. Each objective uses the common training settings in Table 8 and is evaluated on the same source and role groups with identical decoding settings (Appendix C).

The seven source rates, two role averages, and primary comparison follow directly from the per-prompt binary labels. Evaluation uses the common prompt order, chat template, greedy decoder, and classifiers with each trained adapter. Primary objectives use the ordered 5,333-pair training set, while the 24-step ablations use the 649-pair component-study replay bank.

The per-prompt records link each answer to its source, role, and error label. They support several calculations without generating new answers: counting errors within a source, recomputing role averages, restricting the evaluation to the extension, and comparing decisions between adapters. Because answers and labels are stored together, the bootstrap can resample the same prompt for both methods, and source tables, aggregate scores, and correction counts can all be checked against one set of predictions.

H.6 Interpreting Close Results

The 0.02-point Overall difference between APW and APO-Zero follows directly from their source rates and harmful and benign averages, without reference to training loss. The ablation analysis instead examines the replay examples and their weights, changes in harmful and benign training errors, and corrections on individual prompts.

The other analyses change different parts of the procedure. Category-label changes affect replay annotations; sampling changes the generated answers; alternative refusal labels rescore existing answers. Each concerns only that choice and leaves the primary predictions unchanged.

For APW and APO-Zero, the 0.02-point Overall gap combines changes in opposite directions. APO-Zero refuses fewer benign requests; APW makes fewer harmful errors and has lower extension error. A small difference in the mean does not mean the two behave alike. The component-study comparison addresses this at prompt level by counting cases corrected by only one adapter.

I Replay Configuration for Component Studies

The 649-pair component-study bank adds category selection, example weights, and category offsets to adaptive preference weighting. These controls act on the replay examples; the primary 5,333-pair comparison instead uses unit example weights and zero offsets. In the component and coverage analyses, category caps determine the retained bank, and example weights and offsets control its contribution to the adaptive update (Figure 2).

I.1 Candidate Pairs

Harmful candidates come from WildJailbreak training prompts and harmful preference pairs. The initial model produces one greedy answer and two sampled answers with temperature 0.8, top-p 0.95, and a 64-token limit. We retain answers of at least six words only when WildGuard and a lexical check both mark them as non-refusals and Llama-Guard-3-8B assigns unsafe probability above 0.5. Benign pairs require an informative chosen answer and a refused answer to the same prompt; prompts containing risk terms are removed before this comparison.

We assign response order before evaluating the loss. Increasing A_i favors the safe response over unsafe assistance on a harmful prompt, and the useful answer over refusal on a benign prompt. The desired behavior is given by the order and need not be inferred from refusal wording. Current errors supply responses that the initial model still gets wrong; existing pairs retain broader examples of desired behavior. Weighting happens only after these responses have been ordered. A larger weight strengthens the same preference, whether the harmful chosen answer refuses or provides non-operational context, or the benign chosen answer addresses the request.

I.2 Example Weights and Category Caps

Each candidate has an incoming weight u_i (one when unspecified), a source factor s_i, a category factor a_{c_i}, and a static role factor κ_{r_i}. Its initial weight is

$$q_i = u_i s_i a_{c_i} \kappa_{r_i}. \tag{18}$$

The source factors are 1.45 for current unsafe answers, 1.30 for current non-refusals, 1.10 for earlier failures, 0.90 for harmful anchors, and 0.85 for benign anchors. Category factors are 1.12 for cyber, 1.10 for weapons and biology, 1.05 for violence, 1.04 for fraud and infrastructure, 1.00 for civic and benign examples, and 0.96 for general safety. Harmful and benign role factors are 1.05 and 1.15.

The harmful category cap uses $\rho_h = 0.50$. In Equation 1, the clipping quantile is $\tau = 0.95$, with $w_{\min} = 0.20$ and $w_{\max} = 3.0$.

Count and total weight are related but not identical. If

$$M_c = \frac{\sum_{i:c_i=c} \widetilde{w}_i}{\sum_j \widetilde{w}_j}, \tag{19}$$

then N_c controls how often a category appears and M_c measures its weighted influence. The cap changes only overrepresented harmful categories, preserving the observed frequencies below it. Clipping changes the largest weights, while the lower bound keeps every selected example from receiving a negligible weight. Together they limit concentration in counts and weights without forcing categories or weights to be uniform.

I.3 Category Offsets

For this configuration, $\mu_i = \mu_{c_i}$ in Equation 4. The role controller follows this unweighted loss before the example and role coefficients are applied in Equation 7. Category offsets are 0.060 for cyber, 0.055 for weapons and biology, 0.050 for violence, 0.045 for fraud and infrastructure, 0.040 for civic requests, 0.035 for general safety, and zero for benign requests. Experiments that change category labels test sensitivity to these offsets.

The offset changes how large the pair advantage must be for the loss to become small. The example weight $\widetilde{w}_i$ acts after that loss is computed, scaling its gradient. Thus a category-specific offset can change the target advantage while the weight bounds still limit each example's weight. When category labels are noisy, a wrong label changes the offset but the bounds on example weights still apply.

J Discussion

APW makes fewer harmful errors than the seven comparison objectives. Those objectives refuse fewer benign requests, but their harmful increases are larger than their benign reductions, leaving APW with the lowest Overall and Ext. means. The training pairs encode two goals. On a harmful request, the model should prefer refusal or safe guidance; on a benign request, it should prefer an answer that addresses the user's question.

Every method receives pairs ordered in these directions. APW also tracks how well the model learns each role. A decreasing combined loss can hide a harmful preference loss that remains above its target; APW can increase the harmful coefficient in that case. The benign coefficient responds to its own loss trajectory. Each coefficient changes the strength of an update on that role while leaving the chosen and rejected responses in their original order.

Mean response log probabilities determine the reference-relative pair loss; the adaptive coefficients then scale its gradient. The primary comparison evaluates both operations as part of APW's update. During that run, the harmful coefficient reaches its upper bound and the benign coefficient its lower bound, giving harmful examples greater relative emphasis near the end of training. Earlier in training the coefficients took other values, set by the loss feedback available at each minibatch, so each minibatch's contribution depends on the trajectory before it.

The separate component configuration studies APW with a replay bank built mainly from current errors. Category caps change how often a topic appears, example weights change the contribution of an individual pair, and adaptive role weights respond to the observed losses. In the component comparisons, removing clipping or benign coverage increases the development mean, which suggests the controls help in combination. These findings concern that replay configuration; the primary comparison uses the common 5,333 ordered pairs.

J.1 Scope

The primary comparison uses one model family, one adapter size, and one training run per method. Larger models may replace overt unsafe completions with subtler partial compliance, or begin with a different refusal tendency. Eight APW training runs characterize seed variation in the component configuration. Repeating the complete objective comparison across model families would show how model scale and initial refusal tendency interact with adaptive weighting.

The evaluation covers short, single-turn English prompts and uses model-based classifiers and an independent lexical check of benign refusal. Conversation-level pairs, human judgments, and finer intent categories would extend the same two-role analysis to longer and more ambiguous interactions.