HALLUCINATION MITIGATION VIA DIRECTIONAL SELF-GUIDED, SAMPLE-SELECTIVE CONTRASTIVE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Hallucination in large language models remains a major obstacle to reliable deployment. Existing fine-tuning and preference optimization methods broadly update the model, potentially perturbing otherwise reliable continuation preferences, while treating preference pairs largely uniformly. We instead formulate hallucination mitigation as self-guided selective contrastive post-training, suppressing a model-generated incorrect continuation only when it remains competitive with the correct one. The method requires no expert annotation or manual data curation and adaptively activates updates only on hallucination-active examples, with the selected fraction varying across training distributions. Despite this selective update scheme, it consistently outperforms/it is comparable to full-data SFT and standard preference-optimization baselines under cross-dataset evaluation, yielding a stronger repair–damage trade-off. These results suggest that targeting hallucination-active examples can be effective, while enabling a lightweight and scalable self-guided framework that automatically generates, filters, and corrects its own hallucination cases.

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong capabilities across language understanding and generation tasks, but hallucination remains a major obstacle to reliable real-world deployment. Devlin et al. (2019); Radford et al. (2019); Lewis et al. (2019) Models can produce fluent but unsupported, contradictory, or incorrect outputs Ji et al. (2023); Lin et al. (2022); Ravi et al. (2024), which is especially problematic in question-answering settings where factual correctness matters more than stylistic quality. A common remedy is to add stronger supervision through full-data fine-tuning, alignment objectives, preference optimization, or curated data (Lin et al., 2024; Ouyang et al., 2022). However, broad updates can overfit to distribution-specific facts, perturb reliable behaviors, and even increase hallucination on unfamiliar examples. (Gekhman et al., 2024; Kang et al., 2025; Si et al., 2025)

We take a different view: hallucinations can arise when an incorrect continuation remains overly competitive despite the model retaining support for the correct answer. This motivates selectively correcting such cases rather than broadly re-optimizing the model. Motivated by this view, we formulate hallucination mitigation as a self-guided selective contrastive learning problem. A frozen base model generates candidate negative continuations, and the correction objective is activated only when a negative is semantically distinct from the gold answer and remains insufficiently suppressed by the current model. Updates therefore focus on hallucination-active cases.

We use a lightweight, parameter-local adaptation that updates only selected projections within a single transformer layer, aiming to repair misaligned continuation preferences while preserving existing competence. (Azaria and Mitchell, 2023; Orgad et al., 2025) The method relies only on model-derived signals, requiring no expert annotation, prompt optimization, or manual data curation.

We ask whether hallucination can be reduced while preserving reliable base-model behavior by selectively correcting cases in which the model prefers a wrong continuation too strongly. Empirically, the proposed method improves the repair–damage trade-off across benchmarks while using only about 10–12% of the training stream. It outperforms or matches full-data SFT and controlled

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.

preference baselines including DPO, ORPO, and KTO, which are trained on the same type of base-generated hallucination pairs but optimize static preferences rather than online gated correction.

Our contributions are as follows:

- We formulate hallucination mitigation as a self-guided selective contrastive post-training problem, where model-generated negative branches are corrected only when they remain competitive against the gold answer.
- We introduce a model-only training framework that constructs corrective good–bad pairs without human annotation, external teachers, prompt optimization, or manual data curation.
- We propose a gated update objective that focuses on hallucination-active cases, reducing unnecessary parameter perturbation and improving the repair–damage trade-off.
- We show robustness across out-of-distribution benchmarks, cross-task HaluEval settings, random seeds, target layers, and judge choices.

2 Related work

2.1 Internal knowledge, misalignment, and model-centric signals

Recent studies argue that large language models can hallucinate even when they already possess the relevant parametric knowledge, because they fail to reliably elicit that knowledge at generation time Zhang et al. (2024); Jiang et al. (2024); Zhang et al. (2025). This can be understood as a form of misalignment in which a plausible correct continuation exists internally, yet an incorrect continuation is selected with relatively stronger support. However, at the level of individual samples, it is difficult to determine from the outside whether an error arises from a genuine lack of knowledge or from a failure of internal alignment despite the presence of the relevant knowledge. In response, a recent line of hallucination mitigation research has increasingly focused on reducing errors by leveraging the model's own internal capabilities, rather than injecting additional external criteria Zhang et al. (2024); Wen et al. (2025); Liang et al. (2024).

2.2 Selective fine-tuning for hallucination repair

Uniformly fine-tuning on all available data is not always a safe strategy for mitigating hallucinations. In fact, several studies report that fine-tuning on new knowledge or on examples unfamiliar to the model can even increase hallucination tendencies Lin et al. (2024); Gekhman et al. (2024); Dang et al. (2026); Huang et al. (2025); Wan et al. (2024). Even factually grounded knowledge data can disrupt the model's prior beliefs or weaken its confidence in previously acquired knowledge. These findings suggest that simply increasing the absolute amount of seemingly high-quality data, from a human perspective, may itself raise the risk of hallucination. Prior work on factuality alignment has introduced approaches such as factuality-aware preference learning Tian et al. (2023) and fine-grained supervision Huang and Chen (2024), but these methods generally rely on response-level preferences, factuality labels Chaduvula et al. (2026), or sentence-level masks Gu et al. (2025). They do not directly study a setting in which the loss is activated only when a bad branch produced by the base model is not sufficiently suppressed.

2.3 Abstention-based hallucination reduction and its limits

To reduce hallucinations, several studies have explored training language models to abstain from answering or to respond with statements such as "I don't know" Huang et al. (2025); Lin et al. (2024); Kim et al. (2025). While such approaches can lower hallucination rates, they also have an important limitation: because they explicitly train abstention behavior, the model may respond conservatively even in cases where it actually possesses the relevant knowledge. In other words, these methods do not fully solve the more fine-grained problem of teaching the model to distinguish between what it truly does not know and what it does know, nor do they calibrate how strongly that knowledge should be expressed.

2.4 Learning from good–bad pairs Without direct internal support comparison

Several prior methods learn from preferred–rejected response pairs Chaduvula et al. (2026); Tian et al. (2023), but they do not construct gold and bad branches from within the same model and directly compare the model's support for each. Moreover, a preferred response in such datasets does not necessarily correspond to the true answer, meaning that these methods do not directly transfer support from an incorrect answer to a correct one, but instead optimize relative comparisons among candidate responses. By contrast, our framework uses the difference between sample-specific good and bad pairs as a direct training signal.

3 Method

3.1 Problem formulation

Let p_θ denote the trainable model and let p_{θ_0} denote a frozen copy of the initial base model. For each example i, the prompt x_i is constructed from the question and, when available, its reference context. We denote the dataset-provided gold answer as y_i^+ and the answer generated by the frozen base model as y_i^-. The adapted model scores both y_i^+ and y_i^- by assigning token-level likelihoods conditioned on the same prompt x_i. The goal is to update the model only when its own generated wrong answer remains insufficiently suppressed relative to the gold answer.

3.2 Self-generated counterfactual continuations

For each prompt x_i, we generate a candidate continuation using the frozen base model:

$$y_i^- = \mathrm{Decode}(p_{\theta_0}(\cdot \mid x_i)). \tag{1}$$

y_i^- reflects the base model's default prediction. Both branches are conditioned on the same prompt, so the comparison isolates the model's relative support for the gold answer and its own generated answer.

The gold and generated sequences are encoded as

$$\mathbf{z}_i^+ = [x_i; y_i^+], \qquad \mathbf{z}_i^- = [x_i; y_i^-]. \tag{2}$$

3.3 Semantic filtering of valid negatives

Each generated continuation is passed through a multi-stage filtering procedure before being used as a negative branch. First, we apply exact-match rules to remove continuations that are identical to the gold answer. Second, we apply numeric matching rules to catch answers that are numerically equivalent despite surface-form differences. Finally, for cases not resolved by these deterministic rules, we use a frozen semantic matcher to determine whether y_i^- and y_i^+ express the same final answer. The semantic matcher is a frozen LLM-based matcher prompted to output SAME if the gold answer and the generated answer are semantically equivalent or sufficiently similar, and DIFFERENT otherwise.

Let

$$s(x_i, y_i^+, y_i^-) \in \{\mathrm{SAME}, \mathrm{DIFFERENT}\} \tag{3}$$

denote this matching decision. A generated continuation is used as a bad branch only when

$$s(x_i, y_i^+, y_i^-) = \mathrm{DIFFERENT}. \tag{4}$$

If the generated continuation is judged equivalent to the gold answer, the example contributes no confidence-correction signal. The semantic matcher is used only to filter valid negative branches.

3.4 Answer-level cross-entropy comparison

For each retained pair (y_i^+, y_i^-), we compute token-level next-token cross-entropy under the adapted model:

$$\ell_{i,t}^+ = -\log p_\theta(z_{i,t}^+ \mid z_{i,<t}^+), \tag{5}$$

$$\ell_{i,t}^- = -\log p_\theta(z_{i,t}^- \mid z_{i,<t}^-). \tag{6}$$

Let $\mathcal{T}_i^+$ and $\mathcal{T}_i^-$ denote the answer-token positions used for the comparison.

We define the branch-level CE values as

$$\bar{\ell}_i^+ = \frac{1}{|\mathcal{T}_i^+|} \sum_{t \in \mathcal{T}_i^+} \ell_{i,t}^+, \tag{7}$$

$$\bar{\ell}_i^- = \frac{1}{|\mathcal{T}_i^-|} \sum_{t \in \mathcal{T}_i^-} \ell_{i,t}^-. \tag{8}$$

3.5 Selective confidence correction objective

We define the bad-branch suppression margin as

$$m_i = \bar{\ell}_i^- - \bar{\ell}_i^+. \tag{9}$$

A larger positive margin means that the model assigns higher loss to the base-generated bad continuation than to the gold continuation. A small or negative margin therefore indicates that the bad branch is insufficiently suppressed. The correction signal is applied only to examples that pass the semantic mismatch filter and remain below the hallucination-margin threshold:

$$a_i = \mathbf{1}\left[s(x_i, y_i^+, y_i^-) = \text{DIFFERENT}\right] \cdot \mathbf{1}[m_i < \tau_h], \tag{10}$$

where τ_h is the activation threshold.

For each active example, we penalize the remaining margin deficit

$$d_i = [\tau_p - m_i]_+, \tag{11}$$

where τ_p is the margin target hyperparameter. The deficit is mapped to a bounded correction weight

$$g_i = \frac{\sigma(d_i)}{\sigma(d_i) + c}, \tag{12}$$

where σ is a nonnegative activation and $c > 0$ controls the gate scale. In the implementation, σ is ReLU.

To avoid suppressing already unlikely bad branches, we further multiply this margin penalty by

$$b_i = \text{clip}\left(\frac{\left[B - \bar{\ell}_i^-\right]_+}{B}, 0, 1\right), \tag{13}$$

where B is the bad-branch cross-entropy target hyperparameter. The per-sample correction loss is

$$\mathcal{L}_i = a_i \cdot g_i \cdot b_i, \tag{14}$$

and the batch loss averages over active examples:

$$\mathcal{L} = \frac{\sum_i a_i \cdot g_i \cdot b_i}{\sum_i a_i}. \tag{15}$$

If no example activates the confidence signal, the loss is set to zero.

This loss has the intended optimization direction. In the active, unclipped ReLU case, minimizing $\mathcal{L}_i$ decreases $\bar{\ell}_i^+$ and increases $\bar{\ell}_i^-$, thereby increasing the margin $m_i = \bar{\ell}_i^- - \bar{\ell}_i^+$. Thus, g_i is not a detached heuristic weight but a differentiable bounded penalty on the remaining margin deficit, while b_i stops the update once the bad branch is already sufficiently unlikely.

3.6 Training dynamics and interpretation

The training dynamics are intentionally conservative. A base-generated contrastive branch is first filtered to ensure that it is semantically different from the gold answer. The correction term is then applied only when this branch remains insufficiently separated from the gold branch under the current model. Once the contrastive branch is already unlikely or sufficiently separated, the sample contributes little or no correction signal. Thus, the objective acts as a selective contrastive repair signal rather than a broad re-optimization objective over all training examples.

Table 1: Hallucination correction results across benchmarks. Imp. and Wors. denote examples corrected or degraded relative to the untrained base model under the same judge pipeline, and Net is computed as Imp. − Wors. Rate denotes the post-training hallucination rate, Data ratio denotes the fraction of training samples used for parameter updates, and Failure Count denotes the number of parsing failure. Parentheses in the Ours Rate column indicate the relative hallucination-rate reduction over the corresponding baseline, computed as $(\mathrm{Rate}_{\mathrm{baseline}} - \mathrm{Rate}_{\mathrm{ours}})/\mathrm{Rate}_{\mathrm{baseline}}$. Results are separated by training dataset, with HaluEval-QA in the upper block and DROP in the lower block.

Dataset	Baseline					Ours					Eval Size
	Imp.	Wors.	Net	Rate	Failure Count	Imp.	Wors.	Net	Rate	Failure Count	
Training Data: HaluEval-QA											
(a) DPO vs. Ours											
Data Ratio				8.50%					10.44%		
DROP (Dua et al., 2019)	472	330	142	43.23%	0	358	200	158	43.06% (+0.39%)	0	9,535
HotpotQA (fullwiki) (Yang et al., 2018)	345	315	30	51.05%	1	265	181	84	50.33% (+1.41%)	2	7,405
(b) SFT vs. Ours											
Data Ratio				100%					10.44%		
DROP	406	283	123	43.43%	0	358	200	158	43.06% (+0.85%)	0	9,535
HotpotQA (fullwiki)	282	259	23	51.15%	2	265	181	84	50.33% (+1.60%)	2	7,405
(c) ORPO vs. Ours											
Data Ratio				8.50%					10.44%		
DROP	559	498	61	44.08%	0	358	200	158	43.06% (+2.31%)	0	9,535
HotpotQA (fullwiki)	396	404	-8	51.57%	1	265	181	84	50.33% (+2.40%)	2	7,405
(d) KTO vs. Ours											
Data Ratio				8.50%					10.44%		
DROP	493	374	119	43.47%	0	358	200	158	43.06% (+0.94%)	0	9,535
HotpotQA (fullwiki)	368	349	19	51.20%	1	265	181	84	50.33% (+1.70%)	2	7,405
Training Data: DROP											
(e) SFT vs. Ours											
Data Ratio				100%					39.94%		
HaluEval-QA	216	227	-11	12.08%	1	320	62	258	9.39% (+22.27%)	1	10,000
HotpotQA (fullwiki)	236	252	-16	51.67%	1	244	138	106	50.03% (+3.17%)	1	7,405
(f) ORPO vs. Ours											
Data Ratio				32.96%					39.94%		
HaluEval-QA	323	250	73	11.24%	2	320	62	258	9.39% (+16.46%)	1	10,000
HotpotQA (fullwiki)	296	299	-3	51.50%	1	244	138	106	50.03% (+2.85%)	1	7,405
(g) DPO vs. Ours											
Data Ratio				32.96%					39.94%		
HaluEval-QA	288	320	-32	12.29%	1	320	62	258	9.39% (+23.60%)	1	10,000
HotpotQA (fullwiki)	300	302	-2	51.49%	2	244	138	106	50.03% (+2.84%)	1	7,405
(h) KTO vs. Ours											
Data Ratio				32.96%					39.94%		
HaluEval-QA	314	206	108	10.89%	2	320	62	258	9.39% (+13.77%)	1	10,000
HotpotQA (fullwiki)	326	245	81	50.36%	1	244	138	106	50.03% (+0.66%)	1	7,405

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We use Qwen2.5-7B-Instruct as the base model and consider two cross-dataset settings: HaluEval-QA→DROP/HotpotQA-FullWiki and DROP→HaluEval-QA/HotpotQA-FullWiki. These settings test transfer across training distributions.

Almost all model parameters are frozen, and gradients are enabled only for some output projections in layer 14. Training uses bfloat16, batch size 1, gradient accumulation 16, maximum sequence length 8192, maximum generation length 16, learning rate 1×10^{-5}, and 100 update steps. The method adds no inference-time module, retrieval component, verifier; all extra computation is confined to post-training.

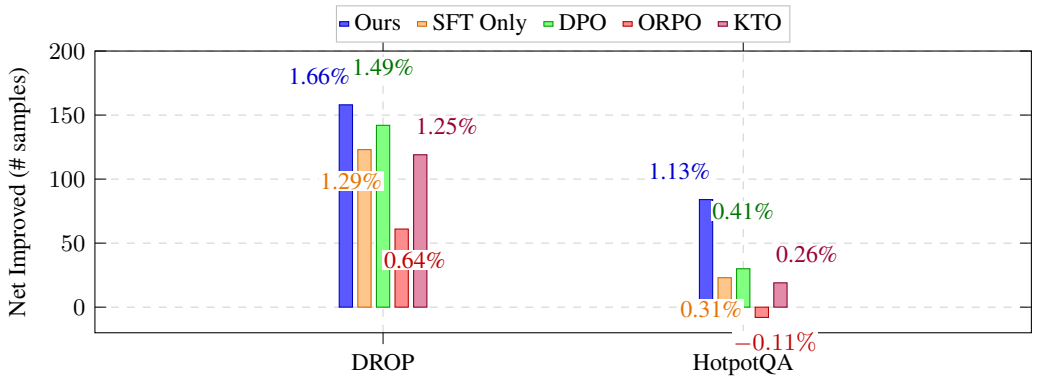


Figure 1: Net improved samples for Ours, SFT-only training, DPO, ORPO, and KTO across benchmarks. Percentages are computed with respect to the evaluation size of each benchmark.

Table 2: Perplexity, MMLU accuracy, SimpleQA accuracy, and FactScore comparison. Lower is better for PPL(wikitext-2, LAMBADA), while higher is better for MMLU, SimpleQA, FactScore.

Dataset	**Split**	**Model**	**Score**	**Δ vs. Baseline**
WikiText-2	test	Qwen2.5-7B-Instruct (baseline)	12.02	0.00
		Ours	**12.01**	−0.01
LAMBADA	validation	Qwen2.5-7B-Instruct (baseline)	24.10	0.00
		Ours	**24.08**	−0.02
MMLU	test	Qwen2.5-7B-Instruct (baseline)	**0.742344**	0.00
		Ours	0.742273	−0.000071
SimpleQA	test	Qwen2.5-7B-Instruct (baseline)	3.51%	0.00
		Ours	3.56%	+0.0005
FactScore	test	Qwen2.5-7B-Instruct (baseline)	0.1115	0.00
		Ours	**0.1147**	+0.0032

Table 3: Ablation study on the negative-branch CE target B and the target preference margin τ_p. Net is computed as Imp. − Wors.

B	τ_p	Step	DROP				HotpotQA (fullwiki)			
			Imp.	Wors.	Net	Rate	Imp.	Wors.	Net	Rate
1.1	0.0	100	365	176	189	42.89%	252	181	71	50.64%
	0.1	100	341	168	173	43.06%	240	168	72	50.63%
	0.2	100	356	177	179	42.99%	248	170	78	50.55%
1.15	0.0	100	350	170	180	42.98%	234	160	74	50.61%
	0.1	100	358	174	184	42.94%	247	174	73	50.62%
	0.2	100	363	180	183	42.95%	253	175	78	50.55%
1.2	0.0	100	346	167	179	42.99%	244	156	88	50.42%
	0.1	100	355	171	184	42.94%	246	153	93	50.35%
	0.2	100	355	172	183	42.95%	246	171	75	50.59%
1.22	0.0	100	353	174	179	42.99%	248	162	86	50.45%
	0.1	100	356	170	186	42.92%	246	161	85	50.46%

We compare with SFT and preference-optimization baselines DPO, ORPO, and KTO. SFT uses full-data gold-answer CE, while the preference baselines use gold/base-generated pairs retained after exact, numeric, and semantic filtering.

Each model is evaluated under the same decoding setup. A hallucination judge compares the prediction with the gold answer and labels it as NOT_HALLUCINATED if the prediction is semantically

Table 4: Judge robustness under two cross-dataset settings. (a) Judge ablation for HaluEval-QA→DROP/HotpotQA-FullWiki. Net Improved (%) is normalized by dataset size; Gap vs. Ours is the Hallucination Rate difference from Qwen2.5-7B-Instruct (pp). (b) Pairwise judge agreement for DROP→HaluEval-QA. Raw Agreement and Cohen's κ measure agreement; A-only/B-only denote samples labeled hallucinated by only one judge.

(a) Judge ablation: HaluEval-QA → OOD evaluation

Judge	Dataset	Improved	Worsened	Net Improved	Net Improved (%)	Hallucination Rate	Gap vs. Ours (pp)
Meta-Llama-3.1-8B	DROP	+343	-178	+165	+1.73%	30.91%	-11.98
	HotpotQA-FullWiki	+280	-145	+135	+1.82%	26.62%	-24.02
IBM-Granite-3.1-8B-Instruct	DROP	+324	-185	+139	+1.46%	36.59%	-6.30
	HotpotQA-FullWiki	+278	-150	+128	+1.73%	38.63%	-12.01
Qwen2.5-7B-Instruct	DROP	+365	-176	+189	+1.98%	42.89%	0.00
	HotpotQA-FullWiki	+252	-181	+71	+0.96%	50.64%	0.00

(b) Pairwise judge agreement: DROP → HaluEval-QA

Dataset	Prediction	Judge A	Judge B	N	Raw Agreement	Cohen's κ	A-only Hallucinated	B-only Hallucinated
HaluEval-QA	Base	Qwen2.5-7B-Instruct	Llama-3.1-8B-Instruct	1000	0.922	0.527	76	2
	Base	Qwen2.5-7B-Instruct	Mistral-7B-Instruct-v0.3	1000	0.898	0.291	102	0
	Base	Llama-3.1-8B-Instruct	Mistral-7B-Instruct-v0.3	1000	0.962	0.483	33	5
HaluEval-QA	Adapted	Qwen2.5-7B-Instruct	Llama-3.1-8B-Instruct	1000	0.941	0.549	57	2
	Adapted	Qwen2.5-7B-Instruct	Mistral-7B-Instruct-v0.3	1000	0.929	0.398	71	0
	Adapted	Llama-3.1-8B-Instruct	Mistral-7B-Instruct-v0.3	1000	0.976	0.635	20	4

equivalent to the gold answer in its final answer, allowing harmless paraphrases, and as HALLUCINATED otherwise. We report hallucination rate, with lower values indicating better factuality.

4.2 Evaluation protocol

We report sample-level changes relative to the frozen base model. An example is *improved* if a hallucinated base answer becomes non-hallucinated after adaptation, and *worsened* if a correct base answer becomes hallucinated. We define

$$\text{Net Improved} = \text{Improved} - \text{Worsened},$$

which directly measures the repair–damage trade-off. Differences in Net Improved reflect changes induced by the training objective. We additionally test evaluator sensitivity by replacing only the judge model in Table 4.

4.3 Preference data construction for DPO, ORPO, and KTO

For DPO, ORPO, and KTO, we first run the frozen base model on the training prompts. We discard empty predictions and predictions that are exact or numerically equivalent to the gold answer. For the remaining cases, a semantic judge keeps only predictions labeled as DIFFERENT. The gold answer and the retained base-generated incorrect answer form the preferred/desirable and dispreferred/undesirable responses. ORPO is a reference-free odds-ratio baseline over chosen/rejected responses, while KTO treats the same gold and negative responses as desirable/undesirable feedback.

This construction makes the preference baselines controlled: they are trained on the same type of base-generated hallucination cases targeted by our method. The retained preference-pair fraction varies with the training distribution. Within each setting, Ours shares the base model, prompt format, generation, filtering, and sequence length, but applies online gated correction instead of optimizing static preference supervision.

4.4 Cross-benchmark generalization

Ours improves the repair–damage trade-off in both cross-dataset settings using model-generated negative branches paired with dataset-provided gold answers, without human-annotated preference

Table 5: Layer ablation study on DROP. Improved, Worsened, Net Improved, and Hallucination Rate are measured for each layer at training step 100.

Layer	Step	Imp.	Wors.	Net	Rate (%)
10	100	351	211	140	43.25
11	100	369	211	158	43.06
12	100	364	199	165	42.99
13	100	356	194	162	43.02
14	100	343	164	179	42.99

Table 6: Seed robustness and significance analysis. (a) Seed ablation for HaluEval-QA→DROP. (b) Mean Hallucination Rate ± sample standard deviation across seeds for DROP→HaluEval-QA. (c) Paired bootstrap comparisons on HaluEval-QA for DROP-trained methods; effects and confidence intervals are in percentage points.

(a) Seed ablation: HaluEval-QA → DROP

Seed	Net	Rate (%)
40	155	43.25
41	191	42.87
42	181	42.97
43	155	43.25
44	157	43.22

(b) Seed stability: DROP → HaluEval-QA

Method	Rate (%)
Ours	$\mathbf{9.3943 \pm 0.0379}$
DPO	12.2962 ± 0.2399
KTO	11.9153 ± 0.8716
ORPO	11.4323 ± 0.6452
SFT	12.1291 ± 0.4653

(c) Paired bootstrap: DROP → HaluEval-QA

Comparison	Effect (pp)	95% CI (pp)	p	Holm p	N
Ours vs. DPO	+2.9009	$[+2.4374, +3.3910]$	9.999×10^{-5}	0.00039996	9997
Ours vs. KTO	+2.5205	$[+1.5770, +3.3807]$	9.999×10^{-5}	0.00039996	9998
Ours vs. ORPO	+2.0371	$[+1.3669, +2.7840]$	9.999×10^{-5}	0.00039996	9998
Ours vs. SFT	+2.7342	$[+2.1640, +3.3444]$	9.999×10^{-5}	0.00039996	9997

pairs. Ours achieves positive Net Improved across both cross-dataset settings. The active-data fraction varies substantially across training sources, indicating that selectivity adapts to the training distribution.

Ours gates each retained bad branch by its current CE relative to the gold branch and its bad-branch CE, reducing pressure once the branch is already unlikely or sufficiently separated.

4.5 Ablation on Correction Targets

Table 3 ablates the two correction targets that determine the magnitude and stopping behavior of the confidence correction. The bad-branch CE target B tests absolute suppression: if the generated wrong answer already has high CE, the method should not keep pushing it away. The margin target τ_p tests relative separation: the wrong branch should have higher CE than the gold branch by at least the target margin.

4.6 Language modeling quality

Table 2 checks whether selective hallucination correction damages general model quality. Perplexity remains essentially unchanged on WikiText-2 and LAMBADA, MMLU is preserved, and SimpleQA and FactScore show small improvements.

4.7 Robustness ablations

Tables 5 and 6 evaluate robustness across layers, seeds, and statistical tests. For HaluEval-QA→DROP, all five seeds remain positive at 100 steps, with Net Improved ranging from 155 to 191. Under DROP→HaluEval-QA, Ours has the lowest mean hallucination rate and the smallest

Table 7: Backbone ablation results on the DROP dataset.

Backbone	Imp. ↑	Wors. ↓	Net ↑	Rate (%) ↓
Meta-Llama-3.1-8B-Instruct	790	621	169	44.86
IBM-Granite-3.1-8B-Instruct	324	241	83	55.64

Table 8: Selective-update ablation under both transfer directions. Ours achieves a lower hallucination rate while using fewer updates than the all-pair variant.

Train	Method	Hallucination Rate ↓	Update Ratio
HaluEval-QA	Ours (Selective)	**43.03%**	**164/1600 (10.25%)**
	All-pair (w/o Selectivity)	43.18%	184/1600 (11.50%)
DROP	Ours (Selective)	**9.39%**	**640/1600 (40.00%)**
	All-pair (w/o Selectivity)	12.81%	690/1600 (43.13%)

variation across seeds; paired bootstrap effects over all baselines are 2.04–2.90 percentage points, with all 95% confidence intervals excluding zero and Holm-adjusted p-values remaining significant.

The layer ablation shows that all tested layers 10–14 produce positive Net Improved scores, with layer 14 strongest at 100 steps. This shows that the gains persist across multiple choices of the local trainable subspace rather than depending on a single target layer.

Table 7 further shows positive Net Improved scores for both Meta-Llama-3.1-8B-Instruct and IBM-Granite-3.1-8B-Instruct on DROP, suggesting that the selective correction effect is not limited to the main Qwen2.5 backbone.

Judge robustness is also consistent: all three judges yield positive Net Improved on both DROP and HotpotQA-FullWiki despite different absolute hallucination rates. On DROP→HaluEval-QA, both Raw Agreement and Cohen's κ increase after adaptation for every judge pair, indicating that the repair trend is not tied to a single evaluator.

4.8 Selective-Update Ablation

Our method selectively updates samples using the hallucination-activation condition, margin-dependent gate, and bad-branch suppression term. To test whether this selectivity is beneficial, we compare against an all-pair variant that keeps the same semantic filtering but applies contrastive correction to every retained negative pair. And experiment shows less hallucination rate with less update ratio.

5 Conclusion

Instead of updating on all preference pairs, our method activates correction only when a base-generated incorrect continuation remains insufficiently suppressed relative to the gold continuation. Across benchmarks, the framework adapts the amount of training data that contributes to parameter updates according to the training distribution. When trained on HaluEval-QA, the active-data ratio remains around 10%, yielding performance comparable to or modestly better than the baselines with substantially fewer updates. When trained on DROP, the active ratio increases to nearly 40%, and the method generally outperforms the baselines across evaluation benchmarks. This suggests that the proposed selectivity does not enforce a fixed sparse-update regime, but instead adjusts the effective update budget to the training distribution, yielding a data-dependent trade-off between the fraction of samples used for correction and the resulting model performance.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, code making, code debugging and refactoring, and the organization and presentation of experimental and technical material. They were also used to help translate the authors' ideas into concrete implementations, explore implementation choices and experimental designs, and identify potentially useful conceptual connections among methods, formulations, and related ideas. Additionally, generative AI tools were used for general proofreading and to help ensure consistency with formatting and submission guidelines.

REPRODUCIBILITY STATEMENT

main code, run scripts, and the shared preference-pair cache are provided in the supplementary material. Experiments use `Qwen/Qwen2.5-7B-Instruct` with the configuration in Table 9 (layer 14, B=1.1, τ_p=τ_h=0, lr 10^{-5}, batch 1×16, 100 steps, seed 42); the final step-100 checkpoint is reported. Datasets are loaded from the Hugging Face Hub with the extraction rules in Appendix A.3; all prompts are given verbatim in Appendices A.4, A.5, and A.9, and all decoding is greedy. Baselines use official TRL trainers with the settings in Appendix A.12. Per-example predictions, judge labels, and the bootstrap script are released.

REFERENCES

Amos Azaria and Tom Mitchell. The internal state of an llm knows when it's lying, 2023.

Sindhuja Chaduvula, Ahmed Y. Radwan, Azib Farooq, Yani Ioannou, and Shaina Raza. Reducing hallucinations in llms via factuality-aware preference learning, 2026.

Renfei Dang, Peng Hu, Zhejian Lai, Changjiang Gao, Min Zhang, and Shujian Huang. Understanding new-knowledge-induced factual hallucinations in llms: Analysis and interpretation, 2026. URL https://arxiv.org/abs/2511.02626.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, 2019.

Zorik Gekhman, Gal Yona, Roee Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. Does fine-tuning LLMs on new knowledge encourage hallucinations? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7765–7784, 2024.

Yuzhe Gu, Wenwei Zhang, Chengqi Lyu, Dahua Lin, and Kai Chen. Mask-dpo: Generalizable fine-grained factuality alignment of llms, 2025.

Chao-Wei Huang and Yun-Nung Chen. Factalign: Long-form factuality alignment of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16363–16375, 2024.

Lei Huang, Xiaocheng Feng, Weitao Ma, Yuchun Fan, Xiachong Feng, Yuxuan Gu, Yangfan Ye, Liang Zhao, Weihong Zhong, Baoxin Wang, Dayong Wu, Guoping Hu, Lingpeng Kong, Tong Xiao, Ting Liu, and Bing Qin. Alleviating hallucinations from knowledge misalignment in large language models via selective abstention learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24564–24579, 2025.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023. ISSN 1557-7341. doi: 10.1145/3571730.

Che Jiang, Biqing Qi, Xiangyu Hong, Dayuan Fu, Yang Cheng, Fandong Meng, Mo Yu, Bowen Zhou, and Jie Zhou. On large language models' hallucination with regard to known facts. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1041–1053, 2024.

Katie Kang, Eric Wallace, Claire Tomlin, Aviral Kumar, and Sergey Levine. Unfamiliar finetuning examples control how language models hallucinate. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3600–3612, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.183.

Hyuhng Joon Kim, Youna Kim, Sang-goo Lee, and Taeuk Kim. When to speak, when to abstain: Contrastive decoding with abstention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9710–9730, 2025.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.

Yuxin Liang, Zhuoyang Song, Hao Wang, and Jiaxing Zhang. Learning to trust your feelings: Leveraging self-awareness in LLMs for hallucination mitigation. In *Proceedings of the 3rd Workshop on Knowledge Augmented Methods for NLP*, pages 44–58, 2024.

Sheng-Chieh Lin, Luyu Gao, Barlas Oguz, Wenhan Xiong, Jimmy Lin, Wen-tau Yih, and Xilun Chen. Flame: Factuality-aware alignment for large language models. In *38th Conference on Neural Information Processing Systems (NeurIPS 2024).*, 2024.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, 2022.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. Llms know more than they show: On the intrinsic representation of llm hallucinations, 2025. URL https://arxiv.org/abs/2410.02707.

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.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI, 2019.

Selvan Sunitha Ravi, Bartosz Mielczarek, Anand Kannappan, Douwe Kiela, and Rebecca Qian. Lynx: An open source hallucination evaluation model, 2024.

Shuzheng Si, Haozhe Zhao, Gang Chen, Cheng Gao, Yuzhuo Bai, Zhitong Wang, Kaikai An, Kangyang Luo, Chen Qian, Fanchao Qi, Baobao Chang, and Maosong Sun. Aligning large language models to follow instructions and hallucinate less via effective data filtering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16469–16488, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.804. URL https://aclanthology.org/2025.acl-long.804/.

Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D. Manning, and Chelsea Finn. Fine-tuning language models for factuality. In *Workshop on Neural Information Processing Systems (NeurIPS 2023)*., 2023.

Fanqi Wan, Xinting Huang, Leyang Cui, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge verification to nip hallucination in the bud. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2616–2633, 2024.

Xueru Wen, Jie Lou, Xinyu Lu, Yuqiu Ji, Xinyan Guan, Yaojie Lu, Hongyu Lin, Ben He, Xianpei Han, Debing Zhang, and Le Sun. On-policy self-alignment with fine-grained knowledge feedback for hallucination mitigation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5215–5231, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, 2018.

Xiaoying Zhang, Baolin Peng, Ye Tian, Jingyan Zhou, Lifeng Jin, Linfeng Song, Haitao Mi, and Helen Meng. Self-alignment for factuality: Mitigating hallucinations in LLMs via self-evaluation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1965, 2024.

Yuji Zhang, Sha Li, Cheng Qian, Jiateng Liu, Pengfei Yu, Chi Han, Yi R. Fung, Kathleen McKeown, ChengXiang Zhai, Manling Li, and Heng Ji. The law of knowledge overshadowing: Towards understanding, predicting and preventing LLM hallucination. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 23340–23358, 2025.

A APPENDIX

This appendix summarizes the computing setup, trainable-parameter selection, training-data construction, prompt templates, objective, evaluation protocol, selective-update accounting, and logged artifacts. The goal is to make the implementation details reproducible and to clarify how the proposed method differs from SFT, preference-optimization baselines, and the no-selectivity ablation.

A.1 COMPUTING RESOURCES AND HYPERPARAMETERS

The implementation is designed for single-device training. The model is loaded with `device_map=none`, and only a restricted set of parameters in one selected transformer layer is updated. The remaining model parameters are frozen. This makes the method substantially lighter than full-model fine-tuning.

We use Qwen2.5-7B-Instruct with `bfloat16` precision, gradient checkpointing, batch size 1, gradient accumulation 16, maximum sequence length 8192, and maximum generated tokens 16. The learning rate is 1×10^{-5} with no weight decay. Unless otherwise stated, the reported main setting uses target layer 14, random row selection with seed 42, trainable ratio 1.0, $B = 1.1$, $\tau_p = 0$, $\tau_h = 0$, gate scale $c = 1$, full-answer CE comparison (`method_use_full_seq=true`), and 100 training steps. The no-selectivity ablation is disabled in the main method (`method_all_pair_contrastive=false`).

Table 9: Main training and evaluation configuration.

Item	**Setting**
Base model	Qwen2.5-7B-Instruct
Training source	HaluEval-QA , Drop
Precision	bfloat16
Device mapping	none
Gradient checkpointing	enabled
Train batch size	1
Evaluation batch size	32
Gradient accumulation	16
Maximum sequence length	8192
Maximum new tokens	16
Learning rate	1×10^{-5}
Weight decay	0.0
Main target layer	14
Layer ablation	{10, 11, 12, 13, 14}
Row selection	Random, seed 42
Trainable ratio	1.0
Semantic match mode	LLM semantic matching
Hallucination-activation threshold τ_h	0
Margin target τ_p	0
Gate activation σ	ReLU
Gate scale c	1
Bad-branch CE target B	1.1 in the reported main setting
Sequence scope	Full answer (`method_use_full_seq=true`)
All-pair contrastive ablation	Disabled in Ours

A.2 TRAINABLE PARAMETER SELECTION

Let h be the hidden size and let $M \in \{0,1\}^h$ be the row mask. All model parameters are first frozen. We then select one transformer layer and enable gradients only for linear projections whose output dimension equals the hidden size. During backpropagation, gradients are multiplied by the row mask:

$$\nabla W \leftarrow M[:, None] \odot \nabla W, \qquad \nabla b \leftarrow M \odot \nabla b.$$

Thus, only selected output rows of the target-layer linear maps are updated. In the main reported runs, the trainable ratio is 1.0, so all rows in the selected projections are active. For smaller trainable ratios, only the sampled rows receive nonzero gradients.

A.3 Training Data and Baselines

The gold answer is used as the correct branch. The frozen base model generates a candidate negative branch. A generated branch is treated as a base-model hallucination only when it differs from the gold answer under exact, numeric, or semantic answer matching.

SFT and the proposed method use the same training source but different training signals. SFT is a CE-only baseline: it trains on the gold answer and disables the confidence correction. The proposed method sets `method_sft_coef=0` and `method_conf_coef=1`, and trains only from the confidence-correction term on retained base-generated hallucination cases.

Table 10: Implementation-level comparison between the SFT baseline and the proposed method.

Component	**SFT**	**Ours**
Correct branch	Gold answer	Gold answer
Base-generated branch	Not used	Used
Branch filtering	Not used	Exact + numeric + semantic
`train_use_clean`	Not used	false
`train_use_halluci`	Not used	true
`method_sft_coef`	1.0	0.0
`method_conf_coef`	0.0	1.0
`method_all_pair_contrastive`	Not used	false
Training signal	Gold CE	Selective confidence correction

For DPO, ORPO, and KTO, preference supervision is constructed from the same retained base-generated negative pool. For each training source, the frozen base model generates an answer with deterministic decoding; empty predictions and answers that are exact, numerically equivalent, or semantically equivalent to the gold answer are removed. The remaining DIFFERENT examples pair the gold answer with the retained base-generated incorrect answer. DPO and ORPO use these as chosen/rejected responses, while KTO uses the same responses as desirable/undesirable feedback.

A.4 Prompt Templates

For both training and evaluation generation, we use answer-only prompts. When a reference is available, the model is instructed to answer using only the reference. Prompts are formatted with the tokenizer chat template when available.

Answer-only system prompt with reference

You are a precise question-answering assistant. Answer the user's question using only the provided reference. Do not explain your reasoning. Do not restate the question. Return only the final answer text. If the answer is a number, return only the number.

Answer-only user prompt with reference

```
REFERENCE: {reference}
QUESTION: {question}
Return only the final answer.
```

Answer-only system prompt without reference

You are a precise question-answering assistant. Answer the user's question as accurately as possible. Do not explain your reasoning. Do not restate the question. Return only the final answer text. If the answer is a number, return only the number.

Answer-only user prompt without reference

```
QUESTION: {question}
Return only the final answer.
```

A.5 Semantic Answer Matching

We first apply exact and numeric matching between the gold answer and the base-generated candidate answer. If the rule-based checks do not identify the answers as matching, we use an LLM semantic matcher. The matcher outputs SAME when the two answers have the same final meaning and DIFFERENT otherwise. A base-generated answer is used as an incorrect branch only when it is judged different from the gold answer.

Semantic answer matching prompt

Judge whether GOLD_ANSWER and CANDIDATE_ANSWER mean the same final answer to QUESTION. Ignore wording differences, abbreviations, aliases, extra names, and formatting if they refer to the same answer. Output only SAME or DIFFERENT.

```
QUESTION: {question}
GOLD_ANSWER: {gold_answer}
CANDIDATE_ANSWER: {candidate_answer}
LABEL:
```

A.6 Selective Correction and No-Selectivity Ablation

For each retained DIFFERENT pair, let

$$m_i = \bar{\ell}_i^- - \bar{\ell}_i^+,$$

where the reported main setting computes both branch losses over the full answer sequence. The selective method first applies the binary activation

$$a_i = \mathbf{1}[m_i < \tau_h],$$

then forms the margin deficit and bounded gate

$$d_i = [\tau_p - m_i]_+, \qquad g_i = \frac{\mathrm{ReLU}(d_i)}{\mathrm{ReLU}(d_i) + c},$$

and the bad-branch suppression term

$$b_i = \mathrm{clip}\left(\frac{[B - \bar{\ell}_i^-]_+}{B}, 0, 1\right).$$

The per-sample confidence correction is therefore

$$C_i = a_i g_i b_i.$$

In the implementation, a retained sample is counted as a confidence-loss contributor only when this correction is non-zero. The confidence loss is normalized by the number of such contributing samples, so zero-correction examples do not dilute the loss average.

For the no-selectivity ablation, we set `method_all_pair_contrastive=true`. Semantic filtering is kept unchanged, but the activation threshold, margin gate, and bad-branch stopping term are bypassed. Every retained DIFFERENT pair contributes

$$C_i^{\mathrm{all}} = \bar{\ell}_i^+ - \bar{\ell}_i^- = -m_i.$$

Thus, the ablation compares selective repair against applying contrastive correction to every semantically valid base-generated negative.

A.7 Self-Gated Sample Usage

The implementation distinguishes a semantically valid base mismatch from a sample that actually contributes to optimization. For Ours, `used_conf` is true only when the example is a valid mismatch, passes the selective objective, and has a non-zero confidence correction. In the all-pair ablation, every retained DIFFERENT pair is marked as a confidence contributor. We log CE usage, confidence usage, their union, and the number of base mismatches for every training window:

$$\text{UsageRatio} = \frac{\#\text{used union}}{\#\text{seen examples}}, \qquad \text{MismatchRatio} = \frac{\#\text{base mismatches}}{\#\text{seen examples}}.$$

Accordingly, the mismatch count records the size of the semantic negative pool, whereas the usage count records the examples that actually contribute to the training objective. Under the all-pair ablation, confidence usage equals the retained mismatch count.

A.8 Evaluation Metrics and Judge

Training is performed on HaluEval-QA. Evaluation is performed on held-out and out-of-distribution benchmarks:

$$\{\text{HaluEval dialogue, HaluEval summarization, DROP, HotpotQA fullwiki}\}.$$

All methods are compared against the frozen Qwen2.5-7B-Instruct baseline.

A sample is counted as improved if the baseline is judged `HALLUCINATED` and the adapted model is judged `NOT_HALLUCINATED`. It is counted as worsened in the opposite case:

$$\text{Net} = \text{Improved} - \text{Worsened}.$$

The hallucination rate is

$$\text{HallucinationRate} = \frac{\#\texttt{HALLUCINATED}}{\#\texttt{total examples}}.$$

Generation randomness is disabled during evaluation.

A.9 Hallucination Judge Prompt

The hallucination label is assigned by a deterministic judge prompt. The judge receives the question, gold answer, optional reference, and model prediction, and is required to output exactly one of two labels.

Hallucination judge prompt

You are a strict hallucination judge.

Given QUESTION, GOLD_ANSWER, optional REFERENCE, and MODEL_PREDICTION, decide whether MODEL_PREDICTION is hallucinated.

Return NOT_HALLUCINATED if the prediction gives the same final answer as GOLD_ANSWER, allowing harmless paraphrases.

Return HALLUCINATED if the prediction contradicts GOLD_ANSWER, fails to answer the question, or adds incorrect factual claims.

Output only one label: NOT_HALLUCINATED or HALLUCINATED.

```
QUESTION: {question}
GOLD_ANSWER: {gold_answer}
REFERENCE: {reference}
MODEL_PREDICTION: {prediction}
LABEL:
```

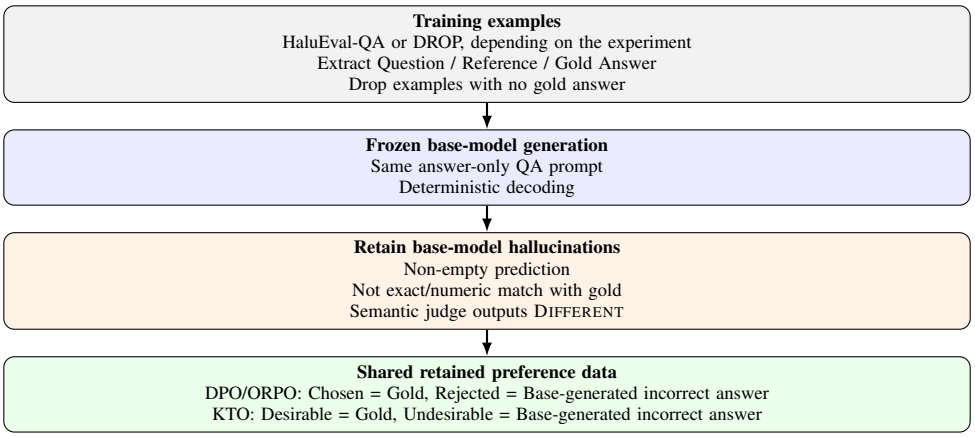


Figure 2: Preference-data construction shared by DPO, ORPO, and KTO.

A.10 Generation, Padding, and Truncation

All evaluation generations are deterministic: sampling is disabled (`do_sample=false`), temperature is set to 0, and `top_p` is set to 0. We use left padding for decoder-only generation. If a prompt exceeds the maximum sequence length, the prompt is truncated to fit the length budget. The raw prompt length, used prompt length, and truncation indicator are logged.

For teacher-forced CE computation, prompt tokens are masked with label value -100, and only answer tokens contribute to the CE loss.

A.11 Logged Artifacts

Each run saves the command-line configuration, parameter report, training metrics, evaluation metrics, usage metrics, and evaluation comparisons. When sample tracing is enabled, the implementation additionally saves per-sample JSON traces containing the question, reference, gold answer, prompt text, base-generated answer, matching result, token-level CE statistics, objective terms, the all-pair-ablation flag, per-sample CE/confidence usage flags, effective loss terms, balance scales, and gradient statistics. These artifacts make the selective and all-pair update behavior auditable.

A.12 DPO, ORPO, and KTO Data Construction

For DPO, ORPO, and KTO, we construct supervision from the same training source used in the corresponding experiment (HaluEval-QA or DROP). The frozen base model first generates an answer using the same answer-only QA prompt with deterministic decoding. We discard empty generations and generations that match the gold answer exactly or numerically; the remaining candidates are retained only when the semantic matcher outputs DIFFERENT. The gold answer and the retained base-generated incorrect answer therefore define the same underlying preference example for all three baselines.

The implementation stores these retained examples in a shared pair cache and reuses that cache across DPO, ORPO, and KTO within a matched experimental setting. DPO and ORPO consume each retained example as a chosen/rejected pair, with the gold answer as chosen and the base-generated incorrect answer as rejected. KTO uses the same two responses as desirable and undesirable feedback, respectively. Thus, the three preference baselines differ in their optimization objectives rather than in how the base-generated training examples are selected.

This construction also matches the comparison protocol used for Ours: the base model, training source, answer-only prompt, deterministic generation, and exact/numeric/semantic filtering are held fixed within each setting. The key distinction is that DPO, ORPO, and KTO optimize static retained preference data, whereas Ours applies online selective correction only when a retained base-generated negative remains hallucination-active under the current model.