CARE: Stable Policy Optimization for Low-Entropy Sparse-Reward Regime

Anonymous ACL submission

Abstract

Reinforcement learning with verifiable rewards (RLVR) has become the predominant approach for post-training large language models on tasks where output correctness can be algorithmically verified. PPO-family optimizers (PPO, GRPO, DAPO, DR.GRPO) dominate this space. These algorithms were designed under the assumption of high-entropy action spaces; binary classification breaks that assumption. We study RLVR where the policy emits a binary label and the reward is sparse, and identify two breakdowns of the standard PPO-based objective. In early training, PPO's element-wise surrogate minimization selects unclipped, aggressive updates for negative-advantage high-ratio (NAHR) samples, destabilizing the policy. In late training, uniform batches cause mean-centered advantage to vanish, starving the policy of gradient signal. We propose **C**onservative **A**dvantage **R**atio **E**nhancement (CARE) algorithm with two targeted modifications to PPO-based objective. First, the elementwise minimum between clipped and unclipped surrogates is replaced by a single batch-aggregated comparison, committing to the less aggressive surrogate for the entire batch. Second, advantage normalization is applied conditionally, only when the batch exhibits reward variance; and raw rewards otherwise. We additionally demonstrate that the first modification enables entropy recovery in collapsed-policy regimes without explicit entropy bonuses. Trained on RAGTruth QA, CARE surpasses GRPO by +10.9 points macro-F1 on the in-distribution test set and exhibits robust transfer to out-of-distribution benchmarks: RAGTruth (Summarization, Data2Text); as well as out-of-domain benchmarks: HaluEval (QA, Dialogue, Summarization) and Internal QA (Easy, Hard).

1 Introduction

Reinforcement learning has become the preferred post-training paradigm for large language models, driving recent advances in instruction following, mathematical reasoning, and code synthesis (Schulman et al., 2017; Shao et al., 2024; Ouyang et al., 2022; Agrawal et al., 2025). Within this landscape, *Reinforcement Learning with Verifiable Rewards* (RLVR) (Guo et al., 2025) has emerged as the preferred training regime. By deriving the signal directly from a programmatic correctness check (Cobbe et al., 2021), RLVR sidesteps the learned reward model (Stiennon et al., 2020), removing a confounding source of approximation error. The optimizers in widespread use, namely PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), and DR.GRPO (Liu et al., 2025), share a single template: a clipped importance-weighted surrogate objective combined with a group-normalized advantage. They differ in clip bounds, advantage normalization, and batch filtering, as catalogued in Section 2.

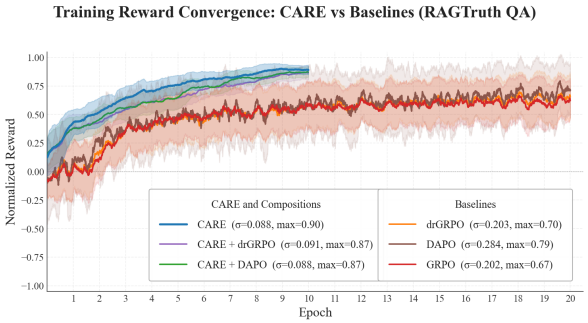


Figure 1: Training reward convergence on RAGTruth QA. CARE converges to ∼0.9 normalized reward in 10 epochs with narrow variance. Baselines (GRPO; DR.GRPO; DAPO) require 2× the training budget (20 epochs) and exhibit substantially wider reward variance, reflecting NAHR-driven instability (early spikes) and gradient starvation (slow late progress). CARE compositions (CARE+DAPO, CARE+DR.GRPO) match CARE's tight variance and convergence speed.

This template was designed for *high-entropy* tasks: multi-step math reasoning, long-horizon code generation, and open-ended chat. Even when

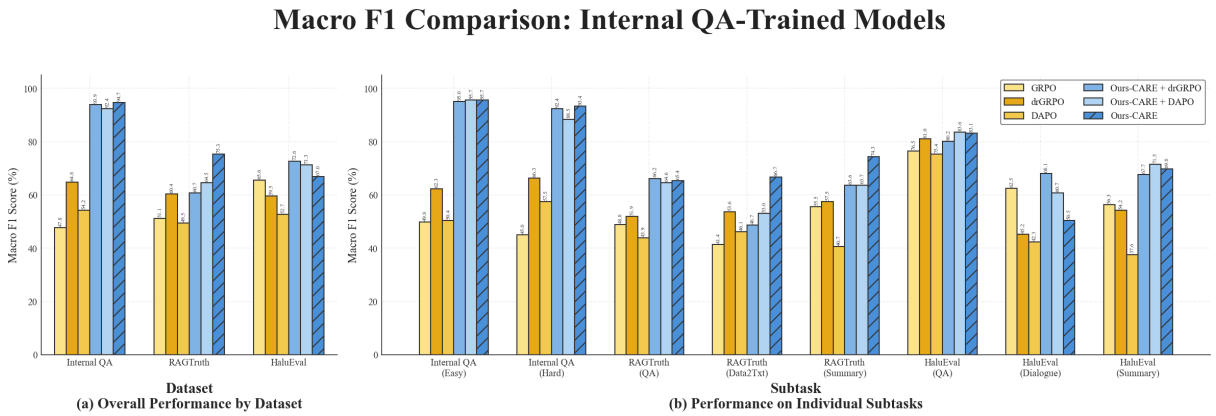


Figure 2: Macro-F1 (%) for models trained on the Internal QA dataset, evaluated across three benchmarks. (a) Performance averaged across subtasks within each benchmark. (b) Per-subtask breakdown. CARE and its compositions (CARE+ DAPO, CARE+ DR.GRPO) form a distinct performance tier above all baselines on every evaluation setting. The gap is largest on Internal QA dataset (+48.4 F1 over GRPO) and persists on out-of-domain benchmarks RAGTruth & HaluEval.

the final reward is binary, these tasks produce long, diverse output sequences i.e. any rollout batch contains varied trajectories and meaningful within-batch advantage dispersion. Binary classification breaks this assumption; the policy emits a single token, bounding output entropy and collapsing the diversity that high-entropy tasks provide implicitly. We term this as **low-entropy sparse-reward regime**: a combination of constrained output space and sparse reward ($R \in \{+1, -1\}$); see Section 3.1.

In this regime, the standard GRPO template exhibits two temporally distinct failure modes. **(i) Early-training instability (NAHR):** The elementwise surrogate selection permits samples with negative advantage and high importance ratio to bypass the clip in the destabilizing direction, injecting excessive gradient magnitude that produces reward spikes and policy oscillation.
(ii) Late-training gradient starvation: As accuracy rises, prompt groups become uniform (all rollouts receive the same reward), and mean-centered advantage yields zero gradient, starving the policy of any learning signal. In high-entropy dense reward tasks, output diversity naturally suppresses both these failures. In the low-entropy regime, the constrained output space removes this implicit regularization, and both failures manifest as structural properties of the template as formalized in Section 3.

Existing methods mitigate these limitations without addressing the root cause. DAPO (Yu et al., 2025) discards uniform prompt groups via dynamic sampling, wasting an increasing fraction of rollouts as training succeeds. DR.GRPO (Liu et al., 2025) removes the standard-deviation denominator, eliminating noise amplification but retaining the zero-gradient collapse from mean centering. Neither modifies the surrogate-level mechanism that drives early-training instability as shown in Figure 1.

We propose CARE (*Conservative Advantage Ratio Enhancement*), which address both failure modes with two targeted modifications to the GRPO template.
Mod 1 (batch-level surrogate selection) replaces the elementwise min between clipped and unclipped surrogates with a single conservative batch-level magnitude comparison, committing to whichever surrogate is less aggressive for the entire update-eliminating the per-sample escape route that permits destabilizing gradients.
Mod 2 (conditional advantage computation) shifts advantage normalization from group level to batch level, and bypasses normalization entirely when the batch is uniform , preserving gradient flow that mean centering would otherwise extinguish. The two modifications introduce no new hyperparameters and require only two conditional checks in the PPO-based objective function.

We study CARE through the lens of *hallucination detection* (Ji et al., 2023; Maynez et al., 2020), a natural testbed of the low-entropy sparse reward regime: given a context, a question, and a candidate answer, the model must emit HALLUCINATED or NOT_HALLUCINATED. The reward is read directly from the ground-truth label. Two considerations make this the right testbed. **(i) Practical.** Regulated industries such as finance, healthcare,

and legal cannot route documents through API-served frontier models due to latency, cost, and data-sovereignty constraints, leaving self-hosted 7-13B SLMs as the only deployable option for in-line factuality checking (Huang et al., 2025). The state-of-the-art compact detector still requires GPT-4-synthesized training data (Tang et al., 2024), a dependency RLVR removes. **(ii) Methodological.** The binary reward is clean, the failure modes are well-defined, and the absence of a learned reward model means any gains we observe is attributable to the optimizer rather than to reward-model artifacts.

2 Related Work

Recent RLVR optimizers share a single template: a clipped importance-weighted surrogate combined with a group-normalized advantage. They differ along three named axes: clip mechanism, advantage estimator, and batch composition. Appendix A gives the full loss formulation for each method. The paragraphs below walk through each axis in turn and describe the failure mode that the corresponding design choice exposes.

Axis 1: Clip mechanism. PPO (Schulman et al., 2017) applies a symmetric ratio clip, $\min\big(rA,\ \mathrm{clip}(r, 1{-}\varepsilon, 1{+}\varepsilon)\,A\big)$. The elementwise $\min$ makes the clip asymmetric in effect: when a sample combines a negative advantage and a ratio above $1{+}\varepsilon$, the $\min$ selects the unclipped rA term which escapes the upper bound, producing the NAHR artifact that destabilizes early training (Zheng et al., 2023). DAPO (Yu et al., 2025) replaces the symmetric clip with decoupled bounds $(\varepsilon_{\text{low}}, \varepsilon_{\text{high}})$ to preserve room for positive updates, but the elementwise $\min$ preserves the same per-sample escape route at the upper bound. GRPO (Shao et al., 2024) and DR.GRPO (Liu et al., 2025) both keep the same symmetric ratio clip as PPO and inherit its escape route. TruelyPPO (Wang et al., 2020) replaces the ratio-based clip trigger with a KL-based trust-region condition and, when the ratio leaves the trust region, applies a negative-slope rollback term $F^{\text{RB}}(r) = -\alpha r + (1{+}\alpha)(1{\pm}\varepsilon)$ that actively reverses the gradient on offending samples rather than zeroing it.

Axis 2: Advantage estimator. GRPO (Shao et al., 2024) normalizes advantage by the mean and standard deviation of rewards over the rollout group that shares a prompt. When the group's rewards are nearly uniform (small $\mathrm{std}(R)$) the std denominator amplifies noise in the advantage; in the limit where every sample in the group receives the same reward, mean centering drives the advantage to zero and the entire group contributes no gradient. DR.GRPO (Liu et al., 2025) drops both the std denominator from the advantage and the $1/|o|$ length normalization from the loss aggregator, keeping only mean centering. The std-induced noise amplification on near-uniform groups disappears, but the zero-gradient collapse on truly uniform groups remains, because mean centering alone still gives $A_i = 0$ whenever $R_i = \bar{R}$.

Axis 3: Batch composition. DAPO is the only optimizer in this family that filters at a granularity coarser than the individual sample. Its dynamic-sampling rule discards every *prompt group* whose rollouts all share the same accuracy ($\mathrm{acc} \in \{0, 1\}$), and over-samples additional prompts until each training batch is filled with informative groups. As training progresses in the low-entropy sparse reward regime, an increasing fraction of prompt groups become uniform and are filtered out, wasting an increasing fraction of generated rollouts and extending wall-clock time per effective gradient step.

Hallucination detection. SelfCheckGPT (Manakul et al., 2023) detects hallucinations via multi-generation consistency but requires 5 to 20× inference calls, impractical for production. MiniCheck (Tang et al., 2024) achieves GPT-4(OpenAI, 2023) parity at 770M parameters but depends on GPT-4-synthesized training data. AlignScore (Zha et al., 2023) and RAGTruth (Wu et al., 2023) document the SFT ceiling for small supervised models. RL4HS (Su et al., 2025) applies GRPO with a span-level F1 reward and class-aware advantage scaling to hallucination span detection on RAGTruth, reporting 58.3% average span F1 on a 14B model.

3 Methodology

3.1 RLVR for Single-Token Classification

The model π_θ receives an input x and generates a single token a. The reward function is:

$$R(a, y) = \begin{cases} +1 & a = y \quad \text{(correct prediction)} \\ -1 & a \neq y \quad \text{(incorrect prediction)} \\ -2 & \text{a is empty or violates format} \end{cases} \tag{1}$$

Each training step samples a batch of N inputs; for each input q, the policy generates G completions, forming a *group*. GRPO computes advantage at

Table 1: Macro-F1 (%) for models trained on RAGTruth QA. Results on the in-distribution RAGTruth QA test set, out-of-domain RAGTruth subtasks (Summarization, Data-to-Text), out-of-distribution Internal QA (Easy, Hard), and out-of-distribution HaluEval (Dialogue, QA, Summarization).

	In-distribution	*Out-of-distribution*						
	RAGTruth	*RAGTruth*		*Internal QA*		*HaluEval*		
Method	**QA**	**Summ.**	**Data2Text**	**Easy**	**Hard**	**Dialogue**	**QA**	**Summ.**
GRPO	69.51	64.03	54.50	53.03	59.18	45.89	79.53	65.00
DR.GRPO	68.95	64.56	43.97	51.89	58.35	44.72	79.98	63.11
DAPO	71.68	67.05	50.91	54.52	59.73	43.61	79.96	63.37
Ours-CARE	**80.41**	**73.58**	52.63	**64.88**	**61.26**	49.44	82.00	**66.44**
Ours-CARE+DR.GRPO	79.75	72.77	**68.20**	58.98	54.54	55.49	**84.22**	64.66
Ours-CARE+DAPO	76.43	72.34	59.72	59.70	60.14	**57.89**	77.96	65.92

group level:

$$A_i = \frac{R_i - \text{mean}(R_q)}{\text{std}(R_q)} \tag{2}$$

where mean and standard deviation are computed over the G completions sharing input q. Importance-sampling (IS) ratios $r_i(\theta) = \pi_\theta(a_i|x_i)/\pi_{\theta_{\text{old}}}(a_i|x_i)$ enable data reuse across K inner optimization epochs per rollout (Queeney et al., 2021). The per sample GRPO clipped surrogate objective (Shao et al., 2024) is:

$$L_{\text{GRPO}}(\theta) = -\frac{1}{G}\sum_i \min\big(r_i(\theta)\cdot A_i, \text{clip}(r_i(\theta), 1-\varepsilon, 1+\varepsilon)\cdot A_i\big) \tag{3}$$

with $\varepsilon = 0.2$. The min is applied **element-wise before averaging**, this is the source of the first failure mode (Section 3.2). The group-level advantage is the source of the second (Section 3.3).

3.2 Failure Mode 1: Asymmetric Clipping

For a sample with advantage A_i and ratio r_i, let $\text{surr1}_i = r_i \cdot A_i$ and $\text{surr2}_i = \text{clip}(r_i, 1-\varepsilon, 1+\varepsilon)\cdot A_i$. When $A_i > 0$ and $r_i > 1+\varepsilon$: min selects surr2_i (clipped, conservative as intended). When $A_i < 0$ and $r_i > 1+\varepsilon$, $r_iA_i < (1+\varepsilon)A_i$ (both negative; r_iA_i has larger magnitude), so min selects r_iA_i i.e. **the unclipped, more aggressive update**.

Proposition 1 (Asymmetric Update)**.** *For* NAHR *samples ($A_i < 0$, $r_i > 1+\varepsilon$),* PPO *selects the unclipped surrogate with excess magnitude $\Delta_i = (r_i - (1+\varepsilon))\cdot|A_i| > 0$. The expected batch excess is $\mathbb{E}[\text{excess}] = f \cdot \mathbb{E}\big[(r-(1+\varepsilon))\cdot|A| \mid r>1+\varepsilon,\ A<0\big]$, where f is the* NAHR *fraction.*

NAHR is an early-training phenomenon. It requires both negative advantage $A_i < 0$ (incorrect prediction) and large IS ratios ($P(r_i > 1+\varepsilon)$, driven by rapid policy change). Both factors diminish as training progresses: the model makes fewer mistakes and the policy stabilises between reference updates. We formalize the proof in Appendix B and talk about NAHR dynamics in Appendix C .

As training progresses NAHR frequency decreases, however it is offset by an elevated advantage magnitude. As accuracy rises within this early phase, the remaining incorrect predictions become sparse outliers: their advantage magnitude grows as $|A_i| = 1 + \text{mean}(R_t) \to 2$ (Appendix C, Eq. C.4). Each surviving NAHR event thus injects more excess energy $\Delta_i = (r_i - (1+\varepsilon))|A_i|$.

In dense-reward settings, NAHR samples are transient and uncorrelated across batches, so their excess energy cancels in expectation. In sparse-reward regime, hard examples fail consistently across rollouts, causing excess energy to *compound* rather than cancel.

Modification 1: Batch-Level Conservative Update. We replace element-wise min with a batch-level magnitude comparison. Let $\overline{\text{surr}_1}$ and $\overline{\text{surr}_2}$ denote the batch-mean unclipped and clipped surrogates respectively:

$$\overline{\text{surr}_1} = \frac{1}{NG}\sum_i r_i(\theta)\cdot A_i \tag{4}$$

$$\overline{\text{surr}_2} = \frac{1}{NG}\sum_i \text{clip}\big(r_i(\theta), 1-\varepsilon, 1+\varepsilon\big)\cdot A_i \tag{5}$$

The CARE policy loss for Mod 1 is:

$$L_{\text{CARE}}(\theta) = \begin{cases} -\overline{\text{surr}_2} & \text{if } |\overline{\text{surr}_2}| < |\overline{\text{surr}_1}| \\ -\overline{\text{surr}_1} & \text{otherwise} \end{cases} \tag{6}$$

The absolute values in Eq. 6 serve strictly as a *selection criterion*: they determine which surrogate

is less aggressive. The signed value of the chosen surrogate is then used as the loss, preserving correct gradient direction.

By comparing batch means, CARE applies the same conservatism criterion regardless of advantage sign; eliminating the asymmetry in Proposition 1 and sharing TRPO's spirit of global conservatism (Schulman et al., 2015; Kakade and Langford, 2002) without second-order computation. In the negative advantage realm, CARE selects a smaller step in policy space, functioning as an implicit adaptive step-size conditioned on observed gradient energy. The batch-level decision also reduces the variance of the conservatism mechanism relative to PPO's per-sample decisions; we quantify this in Section 5.1.

3.3 Failure Mode 2: Vanishing Advantage

As training progresses and model accuracy improves, a growing proportion of input groups become *uniform*: i.e. all G completions sampled for a given input receive identical rewards. Under group-level normalization (Equation 2), the advantage within such uniform groups collapse to zero, resulting in a vanishing policy gradient signal.

Proposition 2 (Gradient Starvation)**.** *Let α_q denote the per-input accuracy for input q. Assume that the G completions per input are drawn independently, and that format violations are negligible such that rewards are effectively binary ($R_i \in \{+1, -1\}$). A group of G completions is uninformative under any mean-centered advantage ($A_i = 0\ \forall i$) with probability $P(\text{uniform}) = \alpha_q^G + (1-\alpha_q)^G$. The expected fraction of zero-gradient groups across a training step is $\mathbb{E}_q[\alpha_q^G + (1-\alpha_q)^G]$, which grows monotonically with model accuracy for $\alpha_q > 1/2$. For $G = 2$ and $\alpha_q = 0.9$, each group is uninformative with probability 0.82; at $\alpha_q = 0.95$, this rises to 0.905.*

This worsens monotonically with training success: the more accurate the model becomes, the fewer groups carry learning signal (Appendix B). Removing the standard-deviation denominator alone does not resolve this, Dr. GRPO retains mean centering ($A_i = R_i - \bar{R}_q$), so uniform groups still produce zero gradient. DAPO's dynamic sampling discards uniform groups entirely, but the fraction discarded grows with accuracy, wasting an increasing share of generated rollouts.

Modification 2
From group-level to batch-level normalization.

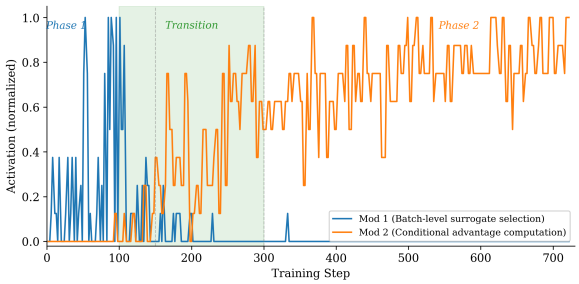


Figure 3: Temporal activation of CARE modifications on RAGTruth QA. Mod 1 peaks in initial training where NAHR events are most frequent, then decays to near-zero as the policy stabilises. Mod 2 activates as uniform batches become dominant ($\text{std}(R) = 0$) and remains high through the remainder of training. The green-shaded transition region shows both modifications simultaneously active, confirming temporal complementarity: Mod 1 absorbs early-phase instability while Mod 2 preserves late-phase gradient flow, with no training phase left unprotected.

The group-level formulation is acute in the binary regime because each group contains only G samples drawn from a Bernoulli distribution. With $G = 2$ (our experimental setting) and effectively binary rewards, the group reward standard deviation can take only two values: 0 (both completions receive the same reward) or 1 (one correct, one incorrect), since $\text{std}(R_q)$ is the population standard deviation computed over the G completions sharing input q. Group-level mean and variance estimated from two binary samples are noisy. The per-input calibration benefit of group normalization (adjusting for input difficulty); is minimal when all inputs share the same binary reward scale.

CARE therefore computes advantage at the *batch level*, pooling across all $N \cdot G$ samples. This has two benefits. First, individual uniform reward groups no longer produce zero gradient: their samples receive non-zero advantage from the batch-level mean and standard deviation as long as the batch contains any reward variation. Second, the batch-level mean and standard deviation is substantially more stable ($N \cdot G$ samples vs. G samples per group). The trade-off is loss of per-input difficulty calibration which is acceptable when rewards are binary and input difficulty variation is bounded. *For multi-step reasoning tasks where input difficulty varies by orders of magnitude, group-level normalization remains preferred.*

Conditional Advantage Computation Batch-level normalization collapses when reward variance vanishes across the batch; specifically, when all $N \cdot$

G samples are assigned the same reward. This occurs in late training phase when the model achieves near-perfect accuracy across a batch. CARE Mod 2 bypasses normalization entirely, using raw reward as advantage:

$$A_i = \begin{cases} \dfrac{R_i - \text{mean}(R)}{\text{std}(R)} & \text{std}(R) > 0 \\ R_i & \text{std}(R) = 0 \end{cases} \tag{7}$$

where $\text{mean}(R)$ and $\text{std}(R)$ are computed over the full batch of $N \cdot G$ samples.

The conditional computation mechanism adds no additional hyperparameter. When all completions are correct, raw rewards still supply positive reinforcement that would be lost under mean centering. By symmetry, when all completions are incorrect ($R_i = -1$ uniformly), raw rewards deliver a corrective penalty ensuring the model continues to learn from these otherwise-ignored uniform-reward batches.

4 Experiments

4.1 Setup

We fine-tune Mistral 7B Instruct v0.3 (Jiang et al., 2023; AI, 2024) using LoRA (Hu et al., 2021) and report Macro-F1 (%) on all datasets.

Baselines. We compare against three PPO variants: GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), and DR.GRPO(Liu et al., 2025). Each baseline method is implemented using the default hyperparameters specified by its authors. To illustrate that CARE composes cleanly with existing methods, we integrate it with both DAPO and DR.GRPO, layering CARE's batch-level operations on top of each algorithm's own sample-level modifications.

Training. All CARE variants share the same optimization setup: AdamW (Loshchilov and Hutter, 2019) with learning rate 4×10^{-5}, cosine annealing, 5% warmup, 4 inner PPO epochs, and $G = 2$. Training runs for 10 outer epochs on 8 x H100s. See Appendix E for the complete configuration.

Datasets. We evaluate CARE across three hallucination detection benchmarks. **RAGTruth** (Wu et al., 2023; Lewis et al., 2020) is a multi-task corpus covering QA, Summarization, and Data-to-text generation, with outputs drawn from GPT-3.5, GPT-4, LLaMA-2(Touvron et al., 2023), and Mistral. **HaluEval** (Li et al., 2023; llm-semantic router, 2025) provides three subtasks: QA, Summary, and Dialogue. **Internal QA** is a proprietary benchmark featuring balanced training instances alongside two test splits: *Easy* and *Hard*, where the latter includes adversarially crafted subtle hallucinations. Training is conducted independently on RAGTruth and Internal QA, with evaluation spanning both in-distribution and cross-domain settings. We balance the RAGTruth training set via undersampling the majority class to match the minority (hallucinated) class count; details in Appendix E.

Table 2: Ablation on RAGTruth QA: individual and combined contributions of each modification.

Condition	Mod 1	Mod 2	Macro F1	Acc
GRPO baseline	×	×	69.51	77.83
+Mod 1 only	✓	×	74.97	84.57
+Mod 2 only	×	✓	79.49	86.06
CARE (both)	✓	✓	**80.41**	**86.86**

4.2 Results

CARE & its compositions outperform all baselines on hallucination detection tasks. Table 1 shows that CARE consistently outperforms pure RLVR baselines (GRPO ,DR.GRPO ,DAPO) across the majority of evaluation settings. On the in-distribution RAGTruth QA split, CARE leads with 80.4%, and this advantage carries over to out-of-domain scenarios including RAGTruth Summarization (73.6%), both Internal QA splits (64.9%/61.3%), and HaluEval QA (82.0%). CARE retains its competitive performance when composed with baselines: CARE+DR.GRPO yields the best results on Data-to-Text (68.2%), HaluEval Summarization (66.9%) and CARE+DAPO outperforms on HaluEval Dialogue score (57.9%).

Figure 2 presents the evaluation results across all benchmarks using the model trained on the Internal QA dataset. CARE and its compositional variants consistently outperform all baselines on both in-distribution and out-of-distribution datasets (full per-subtask breakdown in Table E.1, Appendix).

4.3 Ablation

Component contributions. To isolate each modification's contribution, we ablate on RAGTruth QA. Table 2 shows that their temporal coverage is truly complementary. Combining both the modifications yields additive gains, improving beyond what either one of them achieves alone.

Generation count and sample efficiency. We

ablate the group size $G \in 2, 4, 8$ on RAGTruth QA keeping other hyperparameters constant. Table E.2 confirms that CARE does not degrade with smaller group sizes. The batch-level mechanism draws on enough samples across the full batch to maintain a low-variance advantage estimate. This combined with the faster convergence (Figure 1), motivates our default choice of $G{=}2$. It minimizes the number of generations while keeping adequate signal for stable policy optimization.

5 Analysis

5.1 Training Stability and Additional Properties

We quantify stability via reward curve variance σ_R^2 (variance of the 100-step rolling mean reward). Figure 1 shows that CARE produces substantially smoother reward curves with fewer early-training spikes compared to baselines. We argue that this stabilization comes from the phased coordination of its two modifications whose activations are depicted in Figure 3: Mod1 is primarily active during the early training phase (steps 50–150), dampening initial reward spikes, while Mod2 engages from step $\approx$100 onwards as training transitions to uniform batches, yielding the stable optimization trajectory observed in practice.

The batch-level decision also reduces the variance of the conservatism mechanism relative to GRPO's N independent per-sample decisions; we explain this property in Appendix B. The connection between persistent hard examples and amplified NAHR energy is derived in Appendix C.

5.2 Entropy Recovery via Conservative Selection

Mod 1 prevents gradient overshoot due to NAHR samples (Section 3.2). We now show that the same mechanism produces a complementary benefit: entropy recovery in collapsed-policy regimes by activating gradients in a region where GRPO is clipped (Haarnoja et al., 2018).

Gradient-active regions: We partition the ratio advantage plane into four quadrants at the clip boundaries $[1{-}\varepsilon, 1{+}\varepsilon]$ and $A = 0$. From Proposition 1, GRPO is gradient-active in Q2 ($r > 1{+}\varepsilon$, $A < 0$ i.e. the NAHR zone) and Q4 (within clip bounds). It is clipped in Q1 ($r > 1{+}\varepsilon$, $A > 0$ intended conservatism) and Q3 ($r < 1{-}\varepsilon$, $A < 0$).

CARE's batch-level selection produces the entropy-recovering pattern. In Q3, the unclipped term $r \cdot A$ has *smaller* magnitude than the clipped term $(1{-}\varepsilon) \cdot A$ (since $r < 1{-}\varepsilon$ and both products are negative). The conservative criterion $|\overline{\text{surr}_1}| < |\overline{\text{surr}_2}|$ is satisfied, and CARE selects the unclipped surrogate keeping the gradient active. In Q2, the unclipped surrogate has larger magnitude, so CARE selects the clipped surrogate, zeroing the gradient. CARE is thus active in Q3 and Q4, clipped in Q1 and Q2 (Figure 5).

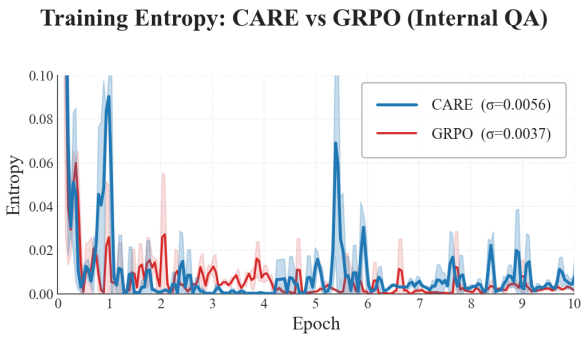
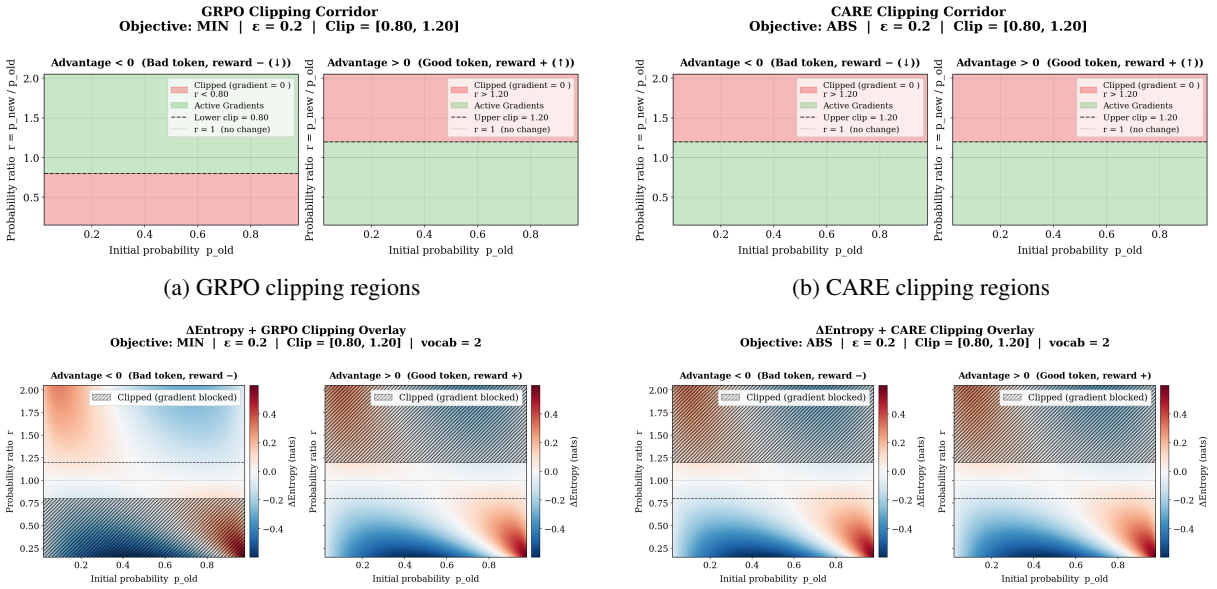


Figure 4: Training entropy for CARE vs. GRPO on Internal QA (rolling mean, window = 5, shaded = $\pm 1\sigma$). CARE sustains higher entropy throughout training, consistent with Q3 gradient activation redistributing probability mass away from collapsed-policy tokens.

How Q3 gradient activation recovers entropy: In the low-entropy sparse-reward regime, the policy tends to collapse to a near-deterministic distribution i.e. it assigns high probability p_{old} to the same token. The corrective signal comes from negative-advantage tokens i.e. misclassified datapoints. These high-p_{old}, negative advantage tokens begin each inner epoch at $r = 1$ and tend to move toward Q3 if the optimizer successfully suppresses their probability.

In Q3, GRPO clips the gradient at $r = 1{-}\varepsilon$, imposing a floor on its suppression and thus preserving probability mass on incorrect outputs. It restricts the optimizer's potential to mitigate policy collapse. CARE continues suppressing across inner epochs, redistributes the probability mass, and recovers entropy. Figure 5 shows that CARE's active Q3 region corresponds to positive ΔEntropy for high-p_{old} negative tokens (red region), precisely where GRPO's clip blocks recovery. Figure 4 provides empirical confirmation: both methods start from identical entropy (0.277), but GRPO collapses to near-zero by step 200 and remains trapped, while CARE recovers entropy in training phases where the batch-level decision selects the unclipped surrogate (corresponding to Q3 activity). The recov-



(a) GRPO clipping regions

(b) CARE clipping regions

(c) GRPO ΔEntropy overlay

(d) CARE ΔEntropy overlay

Figure 5: Gradient-active regions and entropy dynamics for GRPO (left column) vs. CARE (right column). **Top row:** Clipping corridors in the ratio-advantage plane. GRPO is active in Q2 (NAHR zone) and clipped in Q3; CARE is active in Q3 (entropy-recovering), clipped in Q2. **Bottom row:** ΔEntropy heatmap with clipping overlay. GRPO blocks gradients where suppression would increase entropy; CARE permits them, enabling entropy recovery without explicit bonuses.

ery is intermittent rather than monotonic, consistent with the batch-level mechanism activating only when $|\overline{\text{surr}_1}| < |\overline{\text{surr}_2}|$ i.e., when the batch contains sufficient negative-advantage samples to make the unclipped surrogate less aggressive overall.

Self-attenuation and stability. The absence of a clipping floor raises a stability concern. CARE provides implicit self-limitation: as $r \to 0$ in Q3, the gradient $r \cdot A \to 0$, since the selected surrogate is proportional to r. Suppression is self-attenuating for tokens whose probability has already been reduced. Empirically, CARE selects the unclipped surrogate in >85% of steps during the entropy-recovery phase (steps 0-200), confirming batch-level decisions align with per-token Q3 activity.

6 Conclusion

We identified two failure modes of the GRPO objective in the low-entropy sparse-reward regime: NAHR-driven instability in early training and vanishing advantage in late training. We propose CARE, a pair of targeted modifications that resolve both without introducing any new hyperparameters.

Mod 1 replaces the elementwise min between clipped and unclipped surrogates with a single conservative batch-level magnitude comparison, preventing NAHR samples from slipping past the clip, producing smoother reward trajectories and enabling $K{=}4$ inner optimization epochs per rollout. Conditional advantage computation (Mod 2) bypasses normalization when batch rewards become uniform, preserving gradient flow that mean-centering would otherwise suppress. The two modifications jointly yield both faster convergence and higher sample efficiency: CARE reaches 0.9 normalized reward in 10 epochs where baselines require 20, and maintains stable performance at $G{=}2$ generations per input where GRPO demands larger groups to suppress advantage variance. The conservative batch-level mechanism in Mod 1 enables entropy recovery without explicit bonus by redirecting the gradient activity from the destabilizing NAHR quadrant (Q2) to the entropy-recovering region of the ratio-advantage plane (Q3).

We validated this on hallucination detection tasks: CARE achieves +10.9 macro-F1 over GRPO on RAGTruth QA and +48.4 on Internal QA Hard, with consistent gains across out-of-distribution datasets. CARE composes with both DR.GRPO and DAPO inheriting their sample-level modifications while retaining its own stability and convergence benefits. These results position CARE as a strong candidate for replacing GRPO in low-entropy sparse-reward regime.

7 Limitations & Future Work

CARE is designed and evaluated for low entropy sparse-reward RLVR. Its advantage remains to be tested in high-entropy settings (multi-class or continuous-reward tasks) where batches naturally maintain diverse advantages and neither failure mode surfaces as often.

CARE 's conservatism guarantee holds at the batch level rather than per sample, unlike rollback methods such as Truly PPO (Wang et al., 2020) which enforce sample-level bounds.We view this as an acceptable trade-off given the empirical stability gains (Section 5.1). Exploration on other models across additional datasets and training configurations remains to be investigated.

The number of inner optimisation epochs K is currently fixed at 4. An entropy-aware schedule i.e. increasing K when policy entropy has collapsed (to maximise Mod 1's entropy-recovery effect across additional inner epochs) and reducing K when entropy is already healthy (to focus on exploitation rather than further redistribution) could improve sample efficiency while preserving Mod 1's stability guarantees.

Our experiments assume reliable binary labels. In practice, annotations contain noise from annotator disagreement or automated labelling pipelines (Gao et al., 2023). Evaluating CARE under controlled reward noise (e.g., label-flip rates of 5 to 15%) would establish whether batch-level conservatism provides inherent robustness to noisy rewards or whether additional safeguards are needed.

A training-phase-dependent hybrid approach i.e applying group-level normalization during the early stages of training when groups are informative ($\mathrm{std}(R_q) > 0$), and transitioning to batch-level normalization (Mod 2) in later stages could preserve per-prompt calibration early on while avoiding gradient starvation as training progresses. This would extend CARE's applicability to settings with heterogeneous prompt difficulty, such as curriculum-structured RL pipelines.

References

Pulkit Agrawal, Rukma Talwadker, Aditya Pareek, and Tridib Mukherjee. 2025. Fast-q: Fast-track exploration with adversarially balanced state representations for counterfactual action estimation in offline reinforcement learning. page 85–94.

Mistral AI. 2024. Mistral-7b-instruct-v0.3 large language model. https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

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

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Sham Kakade and John Langford. 2002. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 267–274. Morgan Kaufmann.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020.

Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. Halueval: A large-scale hallucination evaluation benchmark for large language models.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding r1-zero-like training: A critical perspective. *Preprint*, arXiv:2503.20783.

llm-semantic router. 2025. Halueval span-level dataset (RAGTruth-normalized). Prompts normalized to RAGTruth format for cross-dataset compatibility.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744.

James Queeney, Razvan Pascanu, and Doina Precup. 2021. Generalized proximal policy optimization with sample reuse. In *Advances in Neural Information Processing Systems*, volume 34, pages 11909–11919.

John Schulman, Sergey Levine, Philipp Moritz, Michael I Jordan, and Pieter Abbeel. 2015. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*.

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

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

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021.

Hsuan Su, Ting-Yao Hu, Hema Swetha Koppula, Kundan Krishna, Hadi Pouransari, Cheng-Yu Hsieh, Cem Koc, Joseph Yitan Cheng, Oncel Tuzel, and Raviteja Vemulapalli. 2025. Learning to reason for hallucination span detection. *arXiv preprint arXiv:2510.02173*.

Liyan Tang, Philippe Laban, and Greg Durrett. 2024. Minicheck: Efficient fact-checking of llms on grounding documents.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Biber, Jessica Birber, Lukas Bischof, Rory Brown, Sharan Choudhary, Josh Chudnovsky, Marta R Costa-jussà, Yann Crego, Abhishek Daultani, and Sankar Deb. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. TRL: Transformers Reinforcement Learning.

Yuhui Wang, Hao He, Chao Wen, and Xiaoyang Tan. 2020. Truly proximal policy optimization. *Preprint*, arXiv:1903.07940.

Yuanhao Wu, Juno Zhu, Siliang Xu, Kashun Shum, Cheng Niu, Randy Zhong, Juntong Song, and Tong Zhang. 2023. RAGTruth: A hallucination corpus for developing trustworthy retrieval-augmented language models.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. Alignscore: Evaluating factual consistency with a unified alignment function.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, Limao Xiong, Lu Chen, Zhiheng Xi, Nuo Xu, Wenbin Lai, Minghao Zhu, Fei Huang, Tao Gui, Qi Zhang, and Xipeng Qiu. 2023. Secrets of RLHF in large language models part I: PPO. *arXiv preprint arXiv:2307.04964*.

A Algorithm Comparison in the Low-Entropy Sparse-Reward Regime

Table A.1 compares each method's behaviour across five dimensions in the binary single-token classification setting ($a \in \{\texttt{True}, \texttt{False}\}$, $R \in \{+1, -1, -2\}$). Loss formulations follow.

GRPO (baseline): Advantages are computed at the group level (over the G completions sharing a prompt) with mean and standard deviation normalization:

$$A_i = \frac{R_i - \text{mean}(R_q)}{\text{std}(R_q)} \tag{A.1}$$

The loss applies the elementwise $\min$ between clipped and unclipped surrogates:

$$L_{\text{GRPO}} = -\frac{1}{N}\sum_i \min\big(r_i A_i, \text{clip}(r_i, 1-\varepsilon, 1+\varepsilon)\cdot A_i\big) \tag{A.2}$$

Failure: For $A_i < 0$, $r_i > 1+\varepsilon$: $\min$ selects $r_i A_i$ (unclipped), producing excess energy $\Delta_i = (r_i - 1 - \varepsilon)|A_i|$. For uniform groups ($\text{std}(R_q) = 0$): $A_i = 0$ for all samples, yielding zero gradient.

DR.GRPO: Removes the standard-deviation denominator and per-token length normalization, retaining only mean centering at the group level:

$$A_i = R_i - \text{mean}(R_q) \tag{A.3}$$

$$L_{\text{DrGRPO}} = -\frac{1}{N}\sum_i \min\big(r_i A_i, \text{clip}(r_i, 1-\varepsilon, 1+\varepsilon)\cdot A_i\big) \tag{A.4}$$

Removing the denominator eliminates noise amplification in dense-reward settings where $\text{std}(R_q)$ can be arbitrarily small. In the binary regime with integer rewards, this is less relevant—the failure is the numerator vanishing on uniform groups ($R_i - \bar{R}_q = 0$ when all R_i are equal), which mean centering preserves regardless of the denominator.

Truly PPO: Replaces the $\min$ operator with a per-sample rollback mechanism on the KL divergence. When the policy exits the trust region ($D^s_{KL}(\pi_{\text{old}}, \pi) \geq \delta$) and the surrogate objective is simultaneously improved ($r_{s,a}(\pi)A_{s,a} \geq r_{s,a}(\pi_{\text{old}})A_{s,a}$), a penalty proportional to the KL divergence is applied:

$$L^{\text{truly}}_{s,a}(\pi) = r_{s,a}(\pi)\, A_{s,a} - \begin{cases} \alpha\, D^s_{KL}(\pi_{\text{old}}, \pi) & C_1 \wedge C_2 \\ \delta & \text{otherwise} \end{cases} \tag{A.5}$$

where $C_1 := D^s_{KL}(\pi_{\text{old}}, \pi) \geq \delta$ and $C_2 := r_{s,a}(\pi)A_{s,a} \geq r_{s,a}(\pi_{\text{old}})A_{s,a}$.

The constant δ in the second case is retained to ensure continuity of the objective. This formulation generates a negative incentive on the KL divergence only when the policy is both outside the trust region *and* the likelihood term is improving, thereby preventing unbounded policy updates without unnecessarily penalising samples inside the trust region. Unlike the ratio-clipping of standard PPO, the trigger condition here is KL-divergence-based, which is theoretically justified by the monotonic improvement bound (Theorem 1). Theorem 6 of the paper further establishes that, under appropriate choice of α and δ, optimising this objective guarantees:

$$\eta\big(\pi^{\text{truly}}_{\text{new}}\big) \;\geq\; \eta(\pi_{\text{old}}) \tag{A.6}$$

where $\eta(\cdot)$ denotes the expected discounted return of a policy, $\alpha > 0$ controls the strength of the rollback penalty, and $\delta > 0$ defines the trust region radius in KL-divergence space.

DAPO: Uses decoupled asymmetric clip bounds and discards uniform groups:

$$L_{\text{DAPO}} = -\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \min\big(r_i A_i, \text{clip}(r_i, 1-\varepsilon_{\text{low}}, 1+\varepsilon_{\text{high}})\cdot A_i\big) \tag{A.7}$$

where $\mathcal{I} = \{i : \text{prompt group of } i \text{ has } 0 < \text{acc}_q < 1\}$. The wider upper clip ($\varepsilon_{\text{high}} = 0.28$) affects only positive-advantage samples (Q1); NAHR behaviour in Q2 is identical to standard GRPO. Dynamic sampling resolves gradient starvation by discarding uniform groups, at the cost of wasting generated rollouts—the fraction discarded grows monotonically with model accuracy (Proposition 2).

CARE (ours): Computes advantages at the batch level and bypasses normalization when the batch is uniform:

$$A_i = \begin{cases} \dfrac{R_i - \text{mean}(R)}{\text{std}(R)} & \text{if } \text{std}(R) > 0 \\ R_i & \text{if } \text{std}(R) = 0 \end{cases} \tag{A.8}$$

Table A.1: Algorithm comparison in the binary sparse-reward regime. Each row describes how the method handles importance sampling, normalization on degenerate batches, NAHR samples, and uniform batches—the two failure modes identified in Section 3.

Method	IS handling	Normalization	NAHR behaviour	Uniform batch	Net effect (binary)
GRPO	Element-wise clip	Group mean + std	Selects unclipped (excess energy)	$A_i = 0$; zero gradient	Both failures active
DR.GRPO	Element-wise clip	Group mean only (no std, no length norm.)	Same as GRPO	$A_i = 0$; zero gradient	Fixes noise amplification; both failures persist
DAPO	Asym. clip ($\varepsilon_h > \varepsilon_l$)	Group mean + std; discard uniform groups	Same as GRPO (wider bound is Q1 only)	Discards uniform groups	Fixes starvation via discarding; NAHR persists; sample-inefficient
Truly PPO	Per-sample rollback	Group mean + std	Resets $r_i = 1$ for NAHR samples	$A_i = 0$; zero gradient	Fixes NAHR; gradient starvation persists
CARE	**Batch-mean selection**	**Batch-level; bypass when** $\mathrm{std}(R)$=0	**Batch-level excess eliminated**	**Raw rewards** (A_i=R_i)	**Addresses both; no discarding**

where $\mathrm{mean}(R)$ and $\mathrm{std}(R)$ are computed over the full batch of $N \cdot G$ samples. The surrogate selection operates at the batch level:

$$\overline{\mathrm{surr}_1} = \frac{1}{N}\sum_i r_i(\theta) \cdot A_i$$

$$\overline{\mathrm{surr}_2} = \frac{1}{N}\sum_i \mathrm{clip}\big(r_i(\theta), 1{-}\varepsilon, 1{+}\varepsilon\big) \cdot A_i$$

$$L_{\mathrm{CARE}} = \begin{cases} -\overline{\mathrm{surr}_2} & \text{if } |\overline{\mathrm{surr}_2}| < |\overline{\mathrm{surr}_1}| \\ -\overline{\mathrm{surr}_1} & \text{otherwise} \end{cases} \tag{A.9}$$

Mod 1 eliminates NAHR by committing to the less aggressive surrogate for the entire batch—no per-sample escape route exists. Mod 2 preserves gradient flow on uniform batches by using raw rewards directly. Since rewards from Eq. 1 are integers, the condition $\mathrm{std}(R) = 0$ is an exact equality requiring no threshold hyperparameter.

B Formal Proofs

Proposition 1 — Asymmetric Update

Statement. For NAHR samples ($A_i < 0$, $r_i > 1{+}\varepsilon$), the GRPO elementwise min selects the unclipped surrogate with excess magnitude $\Delta_i = (r_i - (1{+}\varepsilon)) \cdot |A_i| > 0$.

Proof. Let π_θ be the current policy, $\pi_{\theta_{\text{old}}}$ the reference policy. For action a with advantage A and IS ratio $r = \pi_\theta(a|x)/\pi_{\theta_{\text{old}}}(a|x)$:

$$\mathrm{surr1} = r \cdot A \qquad \text{(unclipped)}$$
$$\mathrm{surr2} = \mathrm{clip}(r, 1{-}\varepsilon, 1{+}\varepsilon) \cdot A \qquad \text{(clipped)}$$
$$L^{\text{elem}}_{\text{GRPO}} = \min(\mathrm{surr1}, \mathrm{surr2})$$

Case: $A < 0$ **and** $r > 1{+}\varepsilon$ **(NAHR):**

Since $r > 0$ and $A < 0$:

$$\mathrm{surr1} = rA < 0$$
$$\mathrm{surr2} = (1{+}\varepsilon)A < 0$$

Since $r > 1{+}\varepsilon$ and $A < 0$, multiplying both sides of the ratio inequality by A reverses the direction:

$$rA < (1{+}\varepsilon)A$$
$$\therefore \quad \mathrm{surr1} < \mathrm{surr2} < 0$$
$$\min(\mathrm{surr1}, \mathrm{surr2}) = \mathrm{surr1} = rA$$

The min selects the *unclipped* (more aggressive) surrogate. Excess magnitude relative to the intended clipped bound:

$$\Delta = |rA| - |(1{+}\varepsilon)A| = (r - 1 - \varepsilon)\,|A| > 0$$

Corollary (gradient under NAHR). When NAHR holds, the loss reduces to $L = rA$ and its gradient is:

$$\nabla_\theta L_{\text{GRPO}} = rA \cdot \nabla_\theta \log \pi_\theta(a|x)$$

Since $A < 0$, gradient ascent pushes π_θ away from a with scalar factor $r|A|$, unbounded above. In near-deterministic policies (low entropy), π_θ concentrates mass on few actions; for any such action where $\pi_{\theta_{\text{old}}}$ assigned comparatively low probability, $r = \pi_\theta(a|x)/\pi_{\theta_{\text{old}}}(a|x) \gg 1$, producing gradient spikes.

Proposition 2 — Gradient Starvation

Statement. Under binary rewards $R_i \in \{+1, -1\}$ with per-prompt accuracy α_q and G independent completions, a prompt group is uninformative ($A_i = 0\ \forall\, i$) with probability $\alpha_q^G + (1 - \alpha_q)^G$.

Proof. A group is uninformative under any mean-centered advantage when all G completions receive the same reward. Under the independence assumption, this occurs when either all are correct (probability α_q^G) or all are incorrect (probability $(1-\alpha_q)^G$). These events are mutually exclusive, so:

$$P(\text{uniform group}) = \alpha_q^G + (1-\alpha_q)^G$$

For any mean-centered advantage $A_i = f(R_i, \bar{R}_q)$ where f depends on the group statistics: when all R_i are equal, $R_i = \bar{R}_q$ for all i, so $R_i - \bar{R}_q = 0$ and $A_i = 0$ regardless of normalization (with or without std division). The expected fraction of zero-gradient groups is:

$$\mathbb{E}_q\big[\alpha_q^G + (1-\alpha_q)^G\big]$$

For $\alpha_q > 1/2$, this is dominated by the α_q^G term, which is monotonically increasing in α_q. As model accuracy improves ($\alpha_q \to 1$), the fraction of uninformative groups grows monotonically toward 1.

Decision Variance Reduction

The following result formalises the variance-reduction property of CARE's batch-level conservatism, referenced in Section 5.1.

Proposition 3 (Decision Variance Reduction)**.** *Define the per-sample surrogate difference:*

$$\begin{aligned}\delta_i &= \mathrm{surr}_{1,i} - \mathrm{surr}_{2,i} \\ &= \big(r_i - \mathrm{clip}(r_i, 1{-}\varepsilon, 1{+}\varepsilon)\big) \cdot A_i\end{aligned} \tag{B.1}$$

with variance σ_δ^2 across samples in a batch. Let $M = N \cdot G$, $\bar{\delta} = \frac{1}{M}\sum_{i=1}^{M} \delta_i$, and $S = \overline{\mathrm{surr}_1} + \overline{\mathrm{surr}_2}$ (the sum of the two batch-mean surrogates).

When $S \neq 0$, CARE's selection criterion $|\overline{\mathrm{surr}_1}| > |\overline{\mathrm{surr}_2}|$ reduces to a sign test on $\bar{\delta}$, whose variance under sample independence is σ_δ^2/M—a factor of M smaller than the variance σ_δ^2 governing any individual per-sample surrogate decision under GRPO.

Proof. CARE's criterion compares magnitudes of the batch-mean surrogates. Since $|\overline{\mathrm{surr}_1}|$ and $|\overline{\mathrm{surr}_2}|$ are non-negative, the comparison is equivalent to comparing their squares:

$$\begin{aligned}|\overline{\mathrm{surr}_1}| > |\overline{\mathrm{surr}_2}| \quad &\iff \quad \overline{\mathrm{surr}_1}^{\,2} > \overline{\mathrm{surr}_2}^{\,2} \\ &\iff \quad \overline{\mathrm{surr}_1}^{\,2} - \overline{\mathrm{surr}_2}^{\,2} > 0\end{aligned} \tag{B.2}$$

Factoring the left-hand side:

$$\begin{aligned}&\overline{\mathrm{surr}_1}^{\,2} - \overline{\mathrm{surr}_2}^{\,2} \\ &\quad = \big(\overline{\mathrm{surr}_1} - \overline{\mathrm{surr}_2}\big)\big(\overline{\mathrm{surr}_1} + \overline{\mathrm{surr}_2}\big) \\ &\quad = \bar{\delta} \cdot S\end{aligned} \tag{B.3}$$

where $\bar{\delta} = \overline{\mathrm{surr}_1} - \overline{\mathrm{surr}_2} = \frac{1}{M}\sum_i \delta_i$ and $S = \overline{\mathrm{surr}_1} + \overline{\mathrm{surr}_2}$.

When $S \neq 0$, the sign of $\bar{\delta} \cdot S$ is determined by the sign of $\bar{\delta}$ (if $S > 0$) or $-\bar{\delta}$ (if $S < 0$). In either case, the binary decision is governed by the magnitude and sign of $\bar{\delta}$, a sample mean of M terms.

Under the assumption that the δ_i are independent with common variance σ_δ^2:

$$\mathrm{Var}(\bar{\delta}) = \frac{\sigma_\delta^2}{M} = \frac{\sigma_\delta^2}{N \cdot G} \tag{B.4}$$

Under GRPO, the conservatism mechanism operates per sample: the elementwise $\min$ selects between $\mathrm{surr}_{1,i}$ and $\mathrm{surr}_{2,i}$ based on which is smaller—a decision driven by the individual δ_i with variance σ_δ^2. CARE replaces M independent noisy decisions with a single decision based on $\bar{\delta}$, reducing decision-noise variance by a factor of $N \cdot G$. □

Condition and scope. The factorization requires $S \neq 0$, i.e., the sum of batch-mean surrogates is non-zero. $S = 0$ arises only when the net expected policy improvement across the batch is exactly zero—a degenerate case where neither surrogate provides meaningful signal. In practice, $S \neq 0$ holds on every observed training step: for informative batches, $S > 0$ (net improvement); for uniform batches, Mod 2 activates before the surrogate selection is reached (Algorithm 1, line 4).

Practical correlation. The independence assumption provides an upper bound on variance reduction. In practice, samples within a batch share policy state and may share prompt-level features, introducing positive correlation. Under equicorrelation ρ among the δ_i:

$$\mathrm{Var}(\bar{\delta}) = \frac{\sigma_\delta^2}{M}\big[1 + (M{-}1)\rho\big] \tag{B.5}$$

The effective variance reduction factor is $M/[1 + (M{-}1)\rho]$. For $M = 32$ and moderate correlation $\rho \in [0.05, 0.15]$, this yields 5–11× reduction, consistent with the design-of-experiments principle that aggregation reduces noise even under correlation, provided $\rho < 1$.

C NAHR Dynamics Derivation

We derive the temporal dynamics of NAHR frequency and per-event magnitude to establish that Mod 1 is most critical in early training, and characterise when Mod 2 activates.

NAHR frequency. NAHR requires both conditions simultaneously: $A_i < 0$ (incorrect prediction) and $r_i > 1+\varepsilon$ (high IS ratio). Let acc_t denote model accuracy at step t. Under effectively binary rewards $R_i \in \{-1, +1\}$ with batch-level advantages:

$$f_{\mathrm{NA}}(t) = 1 - \mathrm{acc}_t \tag{C.1}$$

$$f_{\mathrm{NAHR}}(t) = f_{\mathrm{NA}}(t) \cdot P(r_i > 1+\varepsilon \mid A_i < 0) \tag{C.2}$$

where Eq. C.1 is the fraction with negative advantage (equal to the error rate under mean-centered binary rewards) and Eq. C.2 is the NAHR fraction, bounded above by $f_{\mathrm{NAHR}}(t) \leq f_{\mathrm{NA}}(t)$.

As training proceeds, both factors diminish: the model makes fewer mistakes (f_{NA} decreases) and the policy stabilises (KL divergence $D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$ decreases between reference updates, concentrating the distribution of r_i around 1). By Bayes' rule:

$$P(r_i > 1+\varepsilon \mid A_i < 0) \leq \frac{P(r_i > 1+\varepsilon)}{P(A_i < 0)} \tag{C.3}$$

Since the marginal tail probability $P(r_i > 1+\varepsilon)$ vanishes as $D_{\mathrm{KL}} \to 0$, the conditional also decreases. Under idealised monotone convergence, $f_{\mathrm{NAHR}}(t)$ is approximately monotonically decreasing.

Per-event magnitude growth. For incorrect predictions with reward $R_i = -1$, the advantage magnitude under batch-level mean centering is:

$$|A_i| = |R_i - \mathrm{mean}(R_t)| = 1 + \mathrm{mean}(R_t) \tag{C.4}$$

since $\mathrm{mean}(R_t) = 2\,\mathrm{acc}_t - 1$ for binary rewards. The signed advantage is:

$$\begin{aligned} A_i = R_i - \mathrm{mean}(R_t) &= -1 - (2\,\mathrm{acc}_t - 1) \\ &= -2\,\mathrm{acc}_t < 0 \end{aligned} \tag{C.5}$$

giving $|A_i| = 2\,\mathrm{acc}_t$.

Magnitude under group-level normalization. Under GRPO's group-level normalization with G completions, non-uniform groups in late early-training are dominated by the $k = G-1$ case (all correct except one outlier). For the incorrect sample:

$$|A_i| = \sqrt{\frac{k}{G-k}}\Bigg|_{k=G-1} = \sqrt{G-1} \tag{C.6}$$

giving $|A_i| \approx 2.65$ for $G = 8$ (GRPO) and $|A_i| \approx 3.87$ for $G = 16$ (DAPO)—both exceeding the batch-level bound of $2\,\mathrm{acc}_t \leq 2$. The magnitude-growth phenomenon is thus present under both normalization schemes and is more severe under group-level normalization with large G.

As acc_t increases toward 1, individual NAHR events carry excess energy $\Delta_i = (r_i - (1+\varepsilon)) \cdot |A_i|$ with $|A_i| \to 2$—each surviving event becomes more destructive even as events become rarer. This amplification is most damaging on training sets containing persistent hard examples—samples the model repeatedly misclassifies across rollouts, causing NAHR events to recur on the same inputs rather than appearing transiently on different ones. The +48.4 F1 gap on the Internal QA Hard evaluation (Table E.1) is consistent with this account: baselines destabilise during training on the Internal QA set, while CARE's Mod 1 absorbs the compounding excess.

Mod 2 activation dynamics. Mod 2 activates when the entire batch is uniform: all $N \cdot G$ samples receive the same reward, yielding $\mathrm{std}(R) = 0$. Under binary rewards, this requires either all samples correct or all incorrect:

$$P(\text{batch uniform}) = \prod_{q=1}^{N} \alpha_q^G + \prod_{q=1}^{N} (1-\alpha_q)^G \tag{C.7}$$

Under homogeneous accuracy $\alpha_q = \alpha$:

$$P(\text{batch uniform}) = \alpha^{NG} + (1-\alpha)^{NG} \tag{C.8}$$

For $G = 2$, $N = 16$, and $\alpha = 0.95$: $P = 0.95^{32} + 0.05^{32} \approx 0.19$. At $\alpha = 0.99$: $P = 0.99^{32} + 0.01^{32} \approx 0.72$.

Mod 2 therefore activates on an increasing fraction of batches as training progresses, with the transition occurring gradually rather than at a sharp threshold. Empirically, Mod 2 first activates at approximately step 150 and becomes dominant (active on >50% of batches) by step 250 (Figure 3).

Temporal complementarity. The two modifications provide complementary coverage. Let T_{late} denote the step beyond which NAHR events become negligible ($f_{\mathrm{NAHR}} < 0.01$), and T_{early} the

step at which Mod 2 first activates (first batch with $\mathrm{std}(R) = 0$). Mod 1 is most active during steps 0–T_{late} (typically ≈ 300 steps), and Mod 2 is active from T_{early} onwards (typically ≈ 150 steps). The overlap region $[T_{\text{early}}, T_{\text{late}}]$ is where both are simultaneously active, leaving no phase of training uncovered. Figure 3 in the main text confirms this empirically.

Cumulative excess suppression by Mod 1. In standard PPO, the clip mechanism is designed to produce zero gradient for samples whose IS ratio leaves the trust region: when $r_i > 1{+}\varepsilon$, $\nabla_\theta \mathrm{surr2}_i = \nabla_\theta[(1{+}\varepsilon)A_i] = 0$ since $(1{+}\varepsilon)$ is constant with respect to θ. However, the $\min$ operator selects surr1 in the NAHR case (Proposition 1), defeating this protection and allowing gradient of magnitude proportional to $r_i|A_i|$.

The expected per-step *unintended gradient energy* from NAHR events is:

$$G_{\text{NAHR},t} = f_t \cdot \mathbb{E}\big[r \cdot |A| \;\big|\; r > 1{+}\varepsilon,\, A < 0\big]_t \tag{C.9}$$

where f_t is the NAHR fraction at step t. This is the *full* gradient contribution from NAHR samples, all of which should have been zero under PPO's intended clipping semantics.

Under CARE, Mod 1 operates at the batch level. Define the mean absolute surrogate magnitude for $k \in \{1, 2\}$:

$$\overline{|\mathrm{surr}|}_k = \frac{1}{N \cdot G} \sum_{i=1}^{N \cdot G} |\mathrm{surr}_{k,i}| \tag{C.10}$$

When $|\overline{\mathrm{surr}_1}| > |\overline{\mathrm{surr}_2}|$, the batch objective switches to surr2 entirely. The gradient consequences are:

- **NAHR samples** ($r_i > 1{+}\varepsilon$, $A_i < 0$): $\nabla_\theta \mathrm{surr2}_i = 0$. The entire unintended gradient contribution is eliminated, not merely reduced.
- **Non-NAHR samples** ($r_i \in [1{-}\varepsilon, 1{+}\varepsilon]$): $\mathrm{surr1}_i = \mathrm{surr2}_i = r_i A_i$, so the gradient is unchanged regardless of which surrogate is selected.

Thus Mod 1 selectively zeroes out NAHR gradient contributions while leaving all well-behaved samples unaffected.

The per-step gradient energy eliminated by Mod 1 is:

$$\Delta_{\text{CARE},t} = G_{\text{NAHR},t} \cdot \mathbf{1}\big[|\overline{\mathrm{surr}_1}| > |\overline{\mathrm{surr}_2}|\big]_t \tag{C.11}$$

The cumulative $\sum_t \Delta_{\text{CARE},t}$ quantifies the total unintended gradient energy eliminated by Mod 1 over training.

Remark 1 (Relationship to loss-value excess). The per-sample loss-value excess $\Delta_i^{\text{loss}} = (r_i - (1{+}\varepsilon))|A_i|$ (Proposition 1) measures how much further the objective overshoots the clipped bound in value. The gradient-level excess is the *full* $r_i|A_i|$, since the intended gradient is zero, not $(1{+}\varepsilon)|A_i|$. Both quantities are reported: Δ_i^{loss} characterises the severity of the PPO failure mode, while $G_{\text{NAHR},t}$ characterises the actual gradient energy that Mod 1 must suppress.

D Full Algorithm

The primary change relative to standard GRPO is the batch-level surrogate selection (lines 14–19, Mod 1). Lines 4–9 add conditional advantage computation (Mod 2). The condition $\mathrm{std}(R) > 0$ is an exact integer check on discrete rewards (Section 3.3), requiring no threshold hyperparameter.

Composability. To compose with DR.GRPO, replace the normalized advantage (line 7) with $A_i \leftarrow R_i - \mathrm{mean}(R)$ when $\mathrm{std}(R) > 0$. To compose with DAPO, replace the symmetric clip in line 14 with $\mathrm{clip}(r_i, 1{-}\varepsilon_{\text{l}}, 1{+}\varepsilon_{\text{h}})$.

Practitioners' guide. Use CARE when: (1) your task has low output variability and discrete rewards with fewer than 10 distinct values, such as binary or multi-class classification; or (2) you observe policy entropy collapsing during training due to constrained output entropy, indicating the model is converging to a near-deterministic distribution before reaching a good solution.

E Implementation Details

Model and LoRA. Mistral 7B Instruct v0.3 (Jiang et al., 2023); LoRA rank = 8, α = 16, dropout = 0.05; target modules: `q_proj`, `v_proj`.

Library. All baseline algorithms (GRPO, DR.GRPO, DAPO) are implemented using the TRL library (von Werra et al., 2020). CARE and its compositions (CARE+DR.GRPO, CARE+DAPO) are implemented via customizations to the same

Table E.1: Macro-F1 (%) for models trained on the Internal QA dataset. Results on the in-distribution Internal QA test sets (Easy, Hard), out-of-distribution RAGTruth subtasks (QA, Summarization, Data-to-Text), and out-of-distribution HaluEval subtasks (Dialogue, QA, Summarization).

	In-distribution		*Out-of-distribution*					
	Internal QA		*RAGTruth*			*HaluEval*		
Method	**Easy**	**Hard**	**QA**	**Summ.**	**Data2Text**	**Dialogue**	**QA**	**Summ.**
GRPO	49.77	44.97	48.81	55.50	41.44	62.47	76.49	56.26
DR.GRPO	62.33	66.27	51.89	57.47	53.63	45.18	81.02	54.20
DAPO	50.44	57.47	43.93	40.66	46.12	42.29	75.38	37.61
Ours-CARE	**95.66**	**93.40**	65.39	**74.34**	**66.67**	50.45	83.11	69.83
Ours-CARE+DR.GRPO	95.00	92.42	**66.21**	63.61	48.68	**68.07**	80.15	67.70
Ours-CARE+DAPO	**95.66**	88.47	64.62	63.73	53.00	60.72	**83.56**	**71.54**

Table E.2: Effect of group size G on macro-F1 (%) for RAGTruth QA.

G	CARE (%)
2	80.41
4	81.44
8	81.25

library, modifying the surrogate selection and advantage computation modules while reusing the existing rollout, reward, and optimization infrastructure.

Optimiser. AdamW, lr = 4×10^{-5}, weight decay = 0.01, cosine scheduled decay with 5% warmup.

Hyperparameters. Table E.3 summarises the per-algorithm hyperparameter configuration used in all experiments. Hardware: 8×H100 GPUs.

Wall-Clock Times. CARE completes training in 2.0 hrs, while the compositions CARE+DR.GRPO and CARE+DAPO take 2.1 hrs and 2.2 hrs, respectively. It takes 4.1 hrs for DR.GRPO, 4.3 hrs for GRPO, and 7.3 hrs for DAPO.

Datasets. Table E.4 presents the dataset statistics. Training uses the RAGTruth QA split (Wu et al., 2023) and an internal QA corpus. Evaluation covers RAGTruth (QA, Summarization, Data2Text), HaluEval (QA, Summarization, Dialogue) (llm-semantic router, 2025), and internal QA sets at two difficulty levels.

Span-to-binary label conversion. RAGTruth provides span-level hallucination annotations: each response is associated with a list of spans marked as hallucinatory. We convert these to binary document-level labels by checking whether the span annotation list is non-empty. If at least one span is marked as hallucinatory, the sample is labelled as hallucinated (`True`); if the list is empty, the sample is labelled as not hallucinated (`False`).

Dataset balancing. The RAGTruth QA training split has class imbalance, with fewer hallucinated samples than non-hallucinated samples. Initial experiments showed that baselines failed to learn meaningful decision boundaries on the unbalanced data, while CARE was robust. To ensure a fair comparison across all methods, we undersample the non-hallucinated class to match the hallucinated class count (1,447 per class, 2,894 total), using Python's `random.sample` with seed 42. All reported results use this balanced split.

Modification flags. Both mods `USE_CONSERVATIVE_CLIP=1` (Mod 1); `USE_CONDITIONAL_NORM=1` (Mod 2) enabled for all CARE experiments. Ablation (Table 2) disables each independently.

F Prompt Templates

All tasks are formulated as binary hallucination detection: the model outputs exactly one token (`True` or `False`). Below we present the prompt structure shared across all four evaluation tasks, followed by the task-specific question field that distinguishes each setting.

Shared instruction block. The following instruction is prepended to every input across all tasks:

Table E.3: Hyperparameter configurations for all algorithms.

Parameter	CARE	DR.GRPO	GRPO	DAPO	CARE+DAPO	CARE+DR.GRPO
Num. generations (G)	2	8	8	16	2	2
PPO inner epochs (K)	4	1	1	1	4	4
Num. outer epochs (N)	10	20	20	20	10	10
PPO clip (ε_{low})	0.2	0.2	0.2	0.2	0.2	0.2
PPO clip ($\varepsilon_{\text{high}}$)	0.2	0.2	0.2	0.28	0.28	0.28
Sampling temperature (T)	1.0	1.0	1.0	1.0	1.0	1.0
Sampling top-p	1.0	1.0	1.0	1.0	1.0	1.0
Per-GPU batch size	16	16	16	16	16	16
Normalization scheme	Batch	Group	Group	Group	Batch	Batch

Table E.4: Dataset statistics across training and evaluation splits.

Dataset	Sub-task	Split	Halluc.	Total
RAGTruth	QA	Train	1,447	4,597
RAGTruth Bal.	QA	Train	1,447	1,447
Internal QA	QA	Train	650	650
RAGTruth	QA	Test	160	875
RAGTruth	Summ.	Test	204	900
RAGTruth	Data2Text	Test	579	900
HaluEval	QA	Test	3,589	9,052
HaluEval	Summ.	Test	4,807	9,989
HaluEval	Dialogue	Test	9,812	19,670
Internal QA	QA (easy)	Test	150	150
Internal QA	QA (hard)	Test	150	150

System Instruction (shared across all tasks)

```
## INSTRUCTIONS:

Your task is to determine whether the answer
contains either or both of the following two
types of hallucinations:

1. conflict: instances where the answer
presents direct contradiction or opposition
to the passages;

2. baseless info: instances where the answer
includes information which is not
substantiated by or inferred from the
passages.

If either or both of these types of
hallucinations are present, then output True,
otherwise output False. You can only output
one of these two options: True or False.
```

Input format. Each prompt follows a structured <question>, <passages>, <answer> template. The <question> field varies by task as shown below.

Question Answering (QA)

```
<question>
{user question, e.g., "how do automotive
technicians get paid"}
</question>

<passages>
{retrieved passages relevant to the question}
</passages>

<answer>
{model-generated answer to evaluate}
</answer>
```

Summarization

```
<question>
Summarize the key points of the following
passage.
</question>

<passages>
{source passage to be summarized}
</passages>

<answer>
{model-generated summary to evaluate}
</answer>
```

Data-to-Text

```
<question>
Convert the following structured data into a
natural language description.
</question>

<passages>
{structured data, e.g., JSON with business
attributes}
</passages>

<answer>
{model-generated natural language description
to evaluate}
</answer>
```

Algorithm 1 CARE for RLVR Classification

Require: policy π_θ, ref. model π_{ref}, dataset $\mathcal{D}$
Config: ε=0.2, K=4, G=2, N=16, lr η
for each training iteration **do**
  Sample N prompts $\{(x_q, y_q)\}$ from $\mathcal{D}$
  Generate G completions per prompt: $a_{q,g} \sim \pi_{\theta_{\text{old}}}(\cdot \mid x_q)$
  Compute $R_i \in \{+1, -1, -2\}$ for all $N{\cdot}G$ samples (Eq. 1)
  [Mod 2] Compute advantages:
  if $\text{std}(R) > 0$ **then** ▷ non-uniform batch
    $A_i \leftarrow (R_i - \text{mean}(R))/(\text{std}(R) + 10^{-8})$
  else ▷ uniform: bypass
    $A_i \leftarrow R_i$
  end if
  for $k = 1$ to K inner epochs **do**
    Compute log-probs for generated tokens
    $r_i \leftarrow \pi_\theta(a_i \mid x_i)/\pi_{\theta_{\text{old}}}(a_i \mid x_i)$ ▷ log-space
    $\overline{\text{surr}_1} \leftarrow \frac{1}{NG}\sum_i r_i \cdot A_i$
    $\overline{\text{surr}_2} \leftarrow \frac{1}{NG}\sum_i \text{clip}(r_i, 1{\pm}\varepsilon) \cdot A_i$
    [Mod 1] Batch-level selection:
    if $|\overline{\text{surr}_2}| < |\overline{\text{surr}_1}|$ **then**
      $L \leftarrow -\overline{\text{surr}_2}$ ▷ conservative
    else
      $L \leftarrow -\overline{\text{surr}_1}$
    end if
    Update θ via $\nabla_\theta L$
  end for
  $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
end for

Knowledge-Grounded Dialogue

```
<question>
Respond to the following dialogue using only
information from the given passages.
[Human]: {utterance} [Assistant]: {response}
[Human]: {follow-up}
</question>

<passages>
{knowledge graph triples or grounding
passages}
</passages>

<answer>
{model-generated dialogue response to
evaluate}
</answer>
```

The reward signal is computed based on correctness of the single-token output against ground-truth labels (Section 3.1, Equation 1).