G-Zero: Self-Play for Open-Ended Generation from Zero Data

Chengsong Huang[1], **Haolin Liu**[2], **Tong Zheng**[3], **Runpeng Dai**[4], **Langlin Huang**[1], **Jinyuan Li**[1], **Zongxia Li**[3], **Zhepei Wei**[2], **Yu Meng**[2], **Jiaxin Huang**[1]

[1]Washington University in St. Louis [2]University of Virginia
[3]University of Maryland [4]University of North Carolina at Chapel Hill

{chengsong,jiaxinh}@wustl.edu

Abstract

Self-evolving LLMs excel in verifiable domains but struggle in open-ended tasks, where reliance on proxy LLM judges introduces capability bottlenecks and reward hacking. To overcome this, we introduce G-Zero, a verifier-free, co-evolutionary framework for autonomous self-improvement. Our core innovation is **Hint-δ**, an intrinsic reward that quantifies the predictive shift between a Generator model's unassisted response and its response conditioned on a self-generated hint. Using this signal, a Proposer model is trained via GRPO to continuously target the Generator's blind spots by synthesizing challenging queries and informative hints. The Generator is concurrently optimized via DPO to internalize these hint-guided improvements. Theoretically, we prove a best-iterate suboptimality guarantee for an idealized standard-DPO version of G-Zero, provided that the Proposer induces sufficient exploration coverage and the data filteration keeps pseudo-label score noise low. By deriving supervision entirely from internal distributional dynamics, G-Zero bypasses the capability ceilings of external judges, providing a scalable, robust pathway for continuous LLM self-evolution across unverifiable domains.

Code: https://github.com/Chengsong-Huang/G-Zero

1 Introduction

Self-evolving Large Language Models (LLMs) have emerged as a promising path beyond the limits of human-curated supervision. Rather than relying on static datasets, these models autonomously generate, refine, and learn from their own outputs, offering a scalable route to capabilities that exceed what human imitation alone can provide [27, 26, 31]. This potential has been most clearly demonstrated in reasoning-intensive tasks with strictly verifiable outcomes. In these settings, prior work [37, 10, 16] shows that models can discover complex problem-solving strategies through self-play, continuously improving toward expert-level performance.

However, this paradigm relies crucially on the existence of programmatic oracles. In domains like mathematics or code generation, deterministic signals, such as numerical correctness or functional execution, provide the ground truth required for Reinforcement Learning from Verifiable Rewards (RLVR) [23, 7]. Conversely, a broad class of real-world scenarios, including open-ended instruction following [24], multi-turn dialogue [34], and creative writing, lack such objective oracles. To navigate these settings, existing methods frequently rely on LLM-as-a-judge [6] mechanisms for surrogate reward signals. This workflow introduces two critical limitations. First, the evolving model's performance ceiling is fundamentally bottlenecked by the judge's capabilities. Second, the optimization process is highly vulnerable to reward hacking [28]; rather than genuinely improving response quality, the model learns to exploit the judge's stylistic vulnerabilities, such as bias, formatting preferences, or verbosity. This raises a crucial question: *How can self-evolution scale in unverifiable domains without internalizing these pathologies?*

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

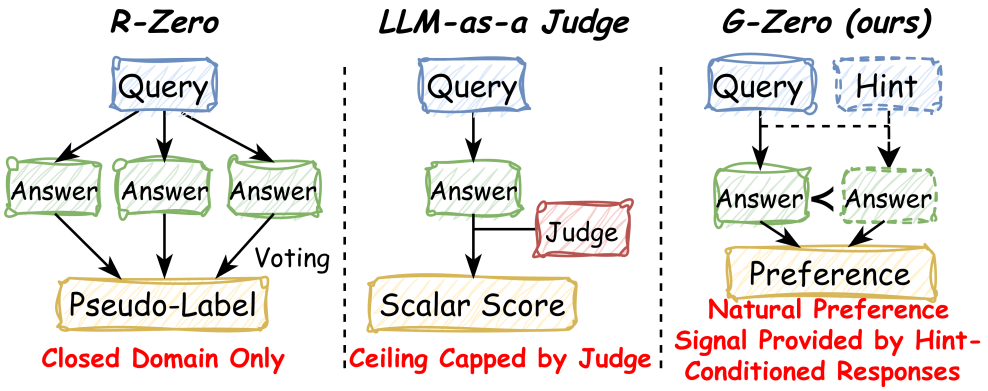

Figure 1: **Comparison of self-supervision signals.** R-Zero [10] uses majority voting, restricting it to verifiable closed-domain tasks. LLM-as-a-Judge assigns scalar scores, bounded by the judge's capability. In contrast, G-Zero creates an internal preference signal by preferring hint-conditioned responses over unassisted ones, eliminating the need for external verifiers or judges.

To move self-evolution beyond verifiable tasks and avoid the flaws of proxy LLM judges, we introduce G-Zero, a verifier-free, co-evolutionary framework that derives supervision entirely from internal dynamics. G-Zero operates through the interaction of two separate models: a Proposer and a Generator. The core innovation of G-Zero is our designed intrinsic signal called **Hint-δ**, which measures how much a hint shifts the Generator's predictive distribution over its own unassisted response without the hint. Hint-δ measures a cognitive gap by coupling two objectives into one scalar: it tends to be large when the underlying query is challenging for the Generator *and* the hint carries necessary information or reasoning that the Generator does not already possess. This is why a hint can stand in for a verifier: a verifier tells us whether an answer is good, whereas a useful hint changes what the model would say. The size of that change on the model's own unassisted response reveals the gap between what it produced and what it could produce given the right information, i.e., a blind spot. Since a large δ alone does not certify that a hint is informative (Appendix E.3), filtering then keeps only the cases where this gap is moderate and learnable, rather than extreme and likely noise or hacking. Using this intrinsic signal, the Proposer is trained via GRPO to synthesize challenging queries paired with informative hints, while the Generator is concurrently optimized via DPO to internalize these hint-guided improvements. Specifically, the Generator learns to favor the hint-guided response (the *chosen* output) over its initial, unassisted answer (the *rejected* output). The two models co-evolve through iterative rounds. This design directly addresses the two problems of judge-based self-evolution. Because Hint-δ is computed entirely from the Generator's own log-probabilities under matched contexts, the difficulty ceiling automatically improves with the Generator's capabilities.

We theoretically and empirically validate the G-Zero framework. Theoretically, we formalize the co-evolutionary loop and prove a best-iterate suboptimality guarantee for an idealized standard-DPO variant, under sufficient Proposer-induced coverage, and low δ-certified pseudo-label score noise. Empirically, G-Zero demonstrates robust improvements within several self-play iterations, and achieves substantial gains on both open-ended (*e.g.*, $+3.74$ points on AlpacaEval) and verifiable (*e.g.*, $+5.21$ points in AIME 25) tasks across diverse model families (Qwen and Llama). Further analysis shows that the model's substantial reasoning improvements do not stem from domain-specific memorization, but from internalizing logical depth in open-ended, non-verifiable tasks, which surprisingly transfers to rigorous domains like mathematical problem-solving.

In summary, our main contributions are:

- **A Verifier-Free Co-Evolutionary Framework**: We propose G-Zero, a self-play pipeline that drives continuous self-evolution through hint-induced response shifts in open-ended domains without external verifiers.

- **Theoretical Characterization of Intrinsic Self-Play**: We formalize the co-evolutionary loop and prove a best-iterate suboptimality guarantee for an idealized standard-DPO variant of G-Zero, with the bound controlled by Proposer-induced coverage, and δ-certified pseudo-label score noise.
- **Empirical Improvements on Both Open-Ended and Verifiable Domains**: We demonstrate that G-Zero brings substantial improvements on instruction-following, chatting, and reasoning tasks across different model families, and also successfully internalize logical depth from open-ended, non-verifiable tasks to rigorous domains like mathematical problem-solving.

2 Preliminaries: Optimization Objectives

Direct Preference Optimization (DPO). Direct Preference Optimization [21] aligns a language model policy π_θ with preferences without requiring a separate reward model. Given a dataset $\mathcal{D}$ of preference triples (x, y_w, y_l), where x represents the input prompt, y_w is the preferred (chosen) response, and y_l is the rejected response, DPO optimizes the policy against a frozen reference model π_{ref} by minimizing the following loss:

$$\mathcal{L}_{\text{DPO}}(\theta) = -\,\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]. \tag{1}$$

where σ is the logistic function and β is a hyperparameter that controls the deviation from the reference policy.

Group Relative Policy Optimization (GRPO). Group Relative Policy Optimization [23] is an efficient reinforcement learning algorithm that omits the need for an external value model. For a given context c sampled from a dataset $\mathcal{P}$, the policy π_θ samples a group of K outputs $\{o_1, \ldots, o_K\}$ (we use K rather than G to avoid clashing with the Generator subscript π_G). The policy is updated by maximizing the following clipped objective, where $\epsilon \in (0, 1)$ is the PPO clip range:

$$\begin{aligned}\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{c\sim\mathcal{P},\{o_i\}_{i=1}^{K}\sim\pi_{\text{old}}}\Bigg[\frac{1}{K}\sum_{i=1}^{K}\min\Bigg(&\frac{\pi_\theta(o_i \mid c)}{\pi_{\text{old}}(o_i \mid c)}A_i,\\ &\text{clip}\Big(\frac{\pi_\theta(o_i \mid c)}{\pi_{\text{old}}(o_i \mid c)}, 1-\epsilon, 1+\epsilon\Big)A_i\Bigg)\Bigg].\end{aligned} \tag{2}$$

Following prior work [20], we omit the KL divergence penalty in our formulation. The advantage A_i is computed by standardizing the scalar rewards $r(o_i)$ within the sampled group: $A_i = (r(o_i) - \mu_r)/\sigma_r$, where $\mu_r = \frac{1}{K}\sum_{j=1}^{K} r(o_j)$ and $\sigma_r = \sqrt{\frac{1}{K}\sum_{j=1}^{K}(r(o_j) - \mu_r)^2}$.

3 The G-Zero Framework

G-Zero is an iterative, co-evolutionary self-play framework designed for continuous LLM self-improvement. Instead of relying on external verifiers or inherently verifiable tasks, we construct preference pairs directly by contrasting the model's unassisted responses against those conditioned on intrinsic hints.

As illustrated in Figure 2, a single training round consists of two interacting phases: (1) **Proposer Training** (§3.2): The Proposer is trained using Hint-δ (defined in §3.1) to identify challenging queries and pair them with informative hints. (2) **Dataset Curation and Generator Training** (§3.3): Hint-δ is repurposed as a quality filter to curate well-suited response pairs. The Generator is then updated via DPO to favor the hint-guided responses over their unassisted baselines.

Through this iterative process, the Generator absorbs the structural and stylistic patterns elicited by the hints, and therefore learns to produce higher-quality independent responses. The improved model then serves as the base for the next round, enabling continuous self-evolution.

3.1 The Intrinsic Learning Signal: Hint-δ

G-Zero is fundamentally driven by a single intrinsic learning signal, **Hint-δ**. Let π_G denote the Generator LLM under training and π_P the Proposer model used to explore the open-ended task space.

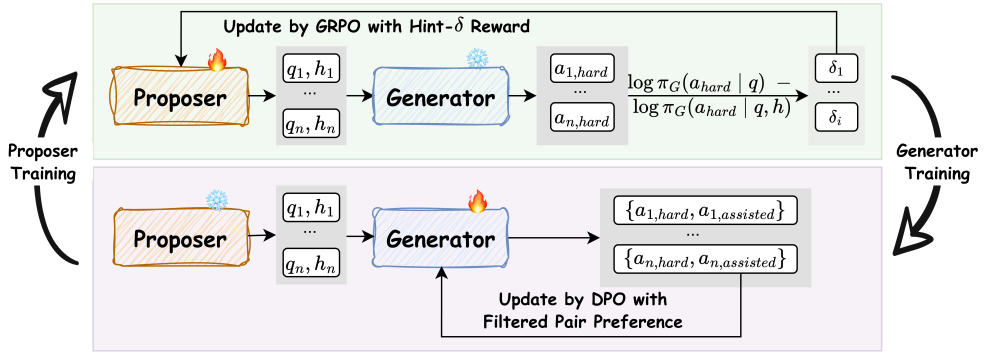

Figure 2: **The G-Zero co-evolutionary loop.** *Top (Proposer training):* The Proposer π_P generates query–hint pairs $\{(q_i, h_i)\}$. The frozen Generator π_G produces unassisted responses, and Hint-δ is computed from the log-probability shift each hint induces on the Generator's distribution. The δ values serve as the GRPO reward, driving π_P to explore the Generator's blind spots. *Bottom (Generator training):* With π_P frozen, the Generator answers each query with and without the hint; We then filter the resulting pairs into a preference dataset $\mathcal{D}_{R+1}$, where R is the round index, on which π_G is updated via DPO.

For a given query q and a proposed hint h, let $a_{\text{hard}} \sim \pi_G(\cdot \mid q)$ be the baseline response generated by the Generator without the hint, with token sequence $a_{\text{hard}} = (a_1, \ldots, a_T)$. The Hint-$\delta$ signal measures how much the hint shifts the Generator's predictive distribution over its own unassisted response, evaluated as a *per-token mean* log-likelihood difference:

$$\delta(q, h, a_{\text{hard}}) = \frac{1}{T} \sum_{t=1}^{T} \Big[\log \pi_G(a_t \mid q, a_{<t}) - \log \pi_G(a_t \mid q, h, a_{<t}) \Big]. \tag{3}$$

We deliberately use the per-token mean rather than the sequence-level sum so that δ is invariant to the length of a_{hard}: the Proposer cannot trivially inflate its reward by eliciting longer unassisted responses. Empirically, on our 1,824-sample R1 raw pool we measure a Spearman rank correlation of -0.41 between δ and the character length of a_{hard}, i.e., longer a_{hard} tend to receive *smaller* δ, which is consistent with the per-token normalization removing the naive length bias.

A key advantage of this formulation is that **Hint-δ** effectively captures both query difficulty and hint informativeness at once, yielding a large δ only when two conditions are jointly met: (i) the underlying query is genuinely challenging for the Generator, so that the unassisted response is flawed or uncertain, and (ii) the hint carries missing knowledge or reasoning steps needed by the Generator to largely reshape the response distribution. Either factor alone is insufficient: If the query is trivial, the hints tend to be redundant with π_G's prior knowledge, leaving the log-probability unchanged ($\delta \approx 0$). Symmetrically, for a difficult query, if the hint is uninformative, it fails to perturb the distribution. Consequently, maximizing δ drives the Proposer to jointly search over query difficulty and hint informativeness, automatically targeting the Generator's blind spots without any external difficulty signal. Crucially, the Proposer's reward is computed against the *current* Generator, so as π_G improves, the threshold for what counts as an "informative hint" rises with it. The two models therefore co-evolve across rounds.

3.2 Proposer Training

The objective of this phase is to train the Proposer π_P to actively propose challenging queries paired with informative hints that elicit a significant, constructive response shift in the Generator.

We design a specific system prompt that instructs π_P to jointly generate a query q and a corresponding hint h, enforcing a strict structural format utilizing `<question>` and `<hint>` XML tags. The Proposer is optimized via GRPO using Eq. (2), where the output o_i corresponds to the generated pair (q_i, h_i). We use the Hint-δ signal in Eq. (3) as our intrinsic reward.

However, optimizing purely for δ introduces vulnerabilities. A naive Proposer might learn to generate excessively verbose text to artificially shift the Generator's distribution. To prevent this reward

hacking, we introduce a **Length Penalty**

$$\mathcal{P}_{\text{length}} \;=\; \lambda \cdot \max\Big(0,\; \tfrac{|h|-200}{100}\Big), \tag{4}$$

where $|h|$ is the hint length in characters and $\lambda = 0.03$ is used in all reported runs, penalizing hints that exceed a reasonable budget of $\sim$200 characters. Furthermore, to prevent the Proposer from collapsing into generating repetitive pairs, we apply a **BLEU Duplication Penalty** ($\mathcal{P}_{\text{BLEU}}$). We agglomeratively cluster all generated questions in the step's batch using sentence-BLEU distance with average-linkage and a merge threshold of 0.5 (i.e., questions whose pairwise BLEU exceeds 0.5 are merged into a cluster). For each rollout we set $\mathcal{P}_{\text{BLEU}} = |C_i|/|B|$, the fraction of the step's batch B that lies in the rollout's own cluster C_i: a unique question receives a small $\sim 1/|B|$ penalty, while a question shared by many rollouts is heavily discounted.

The total reward combines the intrinsic δ signal with the penalties:

$$r(q, h) \;=\; \delta(q, h, a_{\text{hard}}) \;-\; \mathcal{P}_{\text{length}} \;-\; \mathcal{P}_{\text{BLEU}} \tag{5}$$

For formatted error pairs (e.g., missing mandatory XML blocks or empty fields), we apply a hard-coded penalty floor of -1 and skip the δ computation entirely to save computation, while still applying the duplication penalty to punish repeated formatting failures.

3.3 Generator Training and Dataset Curation

In this final phase, we train the Generator π_G on a curated preference dataset $\mathcal{D}_{R+1}$ using the DPO loss (Eq. (1)), with the hint-assisted response a_{assisted} as the chosen sample (y_w) and the unassisted response a_{hard} as the rejected sample (y_l). The reference model π_{ref} is initialized as a frozen snapshot of π_G taken at the start of the round, anchoring DPO updates to a stable behavioral baseline. To neutralize the well-known length bias of vanilla DPO, in which longer chosen responses contribute disproportionately to the gradient regardless of content, we adopt a length-normalized variant that replaces the sequence-summed log-ratio in Eq. (1) with its per-token mean:

$$\mathcal{L}^{\text{LN}}_{\text{DPO}}(\theta) = -\,\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}_{R+1}}\left[\log\sigma\big(\beta\big(\bar{r}_\theta(x,y_w) - \bar{r}_\theta(x,y_l)\big)\big)\right], \quad \bar{r}_\theta(x,y) = \frac{1}{|y|}\log\frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)}. \tag{6}$$

We adopt DPO rather than online RL with a learned reward model for two reasons. First, our preference pairs are constructed from the same model's output distribution under matched contexts, and DPO's closed-form, reference-anchored objective is a natural fit for this self-paired setting. Second, Hint-δ already provides an explicit chosen/rejected signal at the pair level; routing this signal through a separately trained reward model would introduce an additional information bottleneck and approximation error without any clear benefit.

The objective of this DPO training is hint internalization. By training on these pairs, the Generator is incentivized to favor the structural and stylistic patterns present in the hint-guided response, including more deliberate decomposition of the problem and more disciplined use of intermediate steps. As a result, the model tends to reproduce this higher-quality content independently, without requiring the explicit hint from the Proposer at inference time. This enables the Generator to perform substantially better on complex tasks during inference when no external assistance is available.

Training Set Curation. To maximize the efficacy of the DPO phase, we impose stringent filtering criteria on the preference pairs comprising $\mathcal{D}_{R+1}$. The Proposer's GRPO training, by maximizing Hint-δ, has already performed a first stage of selection: the (q, h) pairs it produces concentrate on hard queries equipped with informative hints. Our data curation performs a complementary selection on top of this pool to evaluate whether each pair is well-suited for DPO.

For each query-hint pair $(q, h) \sim \pi_P$, we sample the Generator's dual responses: the unassisted baseline $a_{\text{hard}} \sim \pi_G(\cdot \mid q)$ and the hint-conditional response $a_{\text{assisted}} \sim \pi_G(\cdot \mid q, h)$. We then recompute the δ score on these freshly sampled responses (Eq. (3)) and retain only pairs whose δ falls in the *lower half* of the empirical distribution within each round.

While the Proposer targets the Generator's blind spots by maximizing δ, we apply a contrasting filtering strategy for DPO data curation. In this stage, δ functions as a proxy for the distributional distance between a_{assisted} and a_{hard}. Within the generated pool of complex queries, explicitly retaining preference pairs with relatively lower δ is essential, driven by two fundamental reasons:

Lower-δ pairs serve as hard-to-distinguish training signals. In preference learning, training on pairs with a massive quality gap often yields diminishing returns, as the preference is trivially satisfied. A lower δ indicates that the log-probability shift between the chosen (a_{assisted}) and rejected (a_{hard}) responses is relatively minor. Consequently, these constitute hard-to-distinguish preference pairs. By focusing on these pairs where the reward gap is small, DPO is forced to learn fine-grained, structural improvements in reasoning rather than relying on superficial, easy-to-spot differences.

High-δ pairs violate DPO's implicit KL-divergence constraint. The DPO formulation inherently includes a KL-divergence penalty against the reference model π_{ref}. A very high δ implies that the hint-assisted response a_{assisted} is drastically far away from the Generator's original unassisted distribution. Pushing the policy towards such completely out-of-distribution responses severely violates this implicit KL constraint. This can lead to excessively large gradients, off-manifold drift, and severe training instability. By filtering out the top half of the δ distribution, we naturally regularize the optimization process and ensure a_{assisted} remains a plausible trajectory for the Generator to internalize.

A subtle point concerns the very bottom of the retained band. Pairs with near-zero δ have low *implicit-reward* margin under π_G, since the Generator assigns similar log-probabilities to a_{assisted} and a_{hard}. We deliberately keep these pairs rather than excluding them so that the lower-half filter is defined purely by the ranked δ statistic without an additional minimum-margin cutoff: the bulk of the constructive learning signal in $\mathcal{D}_{R+1}$ is carried by the middle of the lower-half band, while the tail near zero adds only a small amount of low-margin label noise on responses that are independently sampled at temperature 0.7 and therefore lexically distinct.

3.4 Theoretical Analysis

We analyze G-ZERO as an iterative δ-certified exploratory DPO procedure. We consider a simple linear case such that there exists a ground truth reward $R^\star(q,a) = \phi(q,a)^\top \theta^\star$ for any question q and response a, where ϕ is a feature and $\theta^\star \in \mathbb{R}^d$ is a hidden reward parameter. We assume the standard Bradley-Terry model [21] such that $\mathbb{P}(a^+ \succ a^- | q) = \sigma\left(R^\star(q,a^+) - R^\star(q,a^-)\right)$. The performance of the generator has the following guarantee.

Theorem 1. *(Informal) Suppose the game collects retained data from the Proposer such that, after δ-filtering, the data are sufficiently exploratory, and the Generator is updated iteratively by DPO on the cumulative retained data for T rounds. Then, with high probability, there exists an iterate $t_0 \le T$ such that, using a total number of retained samples $\widetilde{O}(d^2/\varepsilon^2)$, the Generator's policy π_{t_0} satisfies*

$$J(\pi^\star) - J(\pi_{t_0}) \le \widetilde{O}(\varepsilon + \sqrt{\eta_\delta})\,.$$

where $\widetilde{O}$ omits log *factors, $J(\pi) = \mathbb{E}_{q\sim Q, a\sim\pi(\cdot|q)}[R^\star(q,a)] - \mathbb{E}_{q\sim Q}\left[D_{\mathrm{KL}}(\pi(\cdot \mid q)\|\pi_{\mathrm{ref}}(\cdot \mid q))\right]$, Q is the target question distribution, $\pi^\star = \operatorname{argmax}_\pi J(\pi)$ and η_δ denotes the self-normalized cumulative score noise induced by incorrect pseudo-labels after δ filtration.*

The theorem separates the two intrinsic signals in G-ZERO. The Hint-δ and filter controls data quality: if retained pairs are calibrated so that a_{assisted} is truly better than a_{hard} with high probability, then the pseudo-label noise η_δ is small. The exploration reward (implemented by BLEU) for the challenger controls data coverage: it drives the challenger toward pairwise feature directions that the generator has not yet learned. Together, these two effects imply an iterative co-evolution guarantee: the Challenger supplies preference pairs that are both reliable enough to trust and novel enough to teach, while cumulative DPO distills them into the Generator. Detailed proof of Theorem 1 is provided in Appendix D.

4 Experiments

4.1 Experimental Setup

Models. To evaluate the generalization capabilities of our proposed method, we evaluate Qwen3-8B-Base [33] and Llama-3.1-8B-Instruct [5]. By testing on both a foundational base model and an instruction-tuned model from distinct, widely adopted families, we demonstrate that our approach is robust to architectural variations and effective regardless of prior alignment stages.

Table 1: Main results. Absolute performance (%) of G-Zero after round 1 and round 2.

	Chat	IFEval				Math		Average
	AlpLC	IF-pS	IF-pL	IF-iS	IF-iL	AIME24	AIME25	
Qwen3-8B-Base								
base model	8.94	43.07	50.28	56.00	61.75	10.42	7.19	33.95
R-Zero	8.04	37.56	42.79	54.12	57.37	**14.92**	**13.34**	32.59
G-Zero (round 1)	**9.07**	42.33	**50.83**	56.24	62.71	12.19	11.36	34.96
G-Zero (round 2)	8.47	**43.81**	**50.83**	**57.92**	**63.43**	11.15	12.40	**35.43**
Llama-3.1-8B-Instruct								
base model	24.12	58.41	65.80	69.42	75.30	5.94	0.42	42.77
R-Zero	21.74	56.94	62.47	66.74	72.93	5.07	0.34	40.89
G-Zero (round 1)	23.88	58.59	**66.35**	69.78	75.54	6.35	**1.04**	43.08
G-Zero (round 2)	**27.86**	**59.52**	**66.35**	**70.38**	**75.78**	**6.77**	0.63	**43.90**

Benchmarks and Evaluation. To evaluate reasoning capabilities, we benchmark on AIME24 and AIME25, reporting the overall mean@32 score from 32 independent responses sampled at a temperature of 0.7. To evaluate instruction-following, we use IFEval [38] with greedy decoding, reporting the four standard metrics (prompt/instruction-level strict and loose accuracies). Lastly, to assess general conversational quality, we report the length-controlled win rate on AlpacaEval 2.0 [3] against GPT-4-Turbo, judged by the Qwen3-235B-A22B-Instruct-2507.

Experiment Configuration. We strictly standardize the hyperparameter settings across the iterative loop. All model training in our experiments is conducted via the Tinker API [1], exclusively utilizing Low-Rank Adaptation (LoRA) [9].

We supplement the δ-based filter with a set of lightweight heuristic checks on the chosen response a_{assisted} to remove pairs that are known to induce DPO artifacts, following standard practice in DPO data curation. Specifically, to prevent the model from learning length as a proxy for quality, we discard pairs exhibiting length inflation ($l_w/l_l > 2.5$, where l_w and l_l are the character lengths of the chosen a_{assisted} (y_w) and rejected a_{hard} (y_l) responses). We also enforce absolute length bounds, requiring $l_w \in [100, 10000]$ to avoid degenerate gradients from extremely short or long responses. Furthermore, to prevent repetition collapse, we discard responses with a zlib compression ratio < 0.15, which reliably flags repetitive or degenerate text (highly repetitive sequences compress to a small fraction of their original size). Finally, we filter out instances of prompt echoing, discarding pairs where a_{assisted} shares a prefix of ≥ 30 characters with q, as well as template leakage, removing any responses containing raw role markers (e.g., "Assistant:"). The remaining high-quality pairs form the final dataset $\mathcal{D}_{R+1}$. We show all hyperparameters in Appendix C, prompts and templates in Appendix A.

4.2 Main Results

Table 1 reports the absolute performance of our co-evolutionary pipeline across two distinct model families, highlighting three key trends. First, our method strictly enhances overall capabilities without external supervision. After two rounds, we observe average absolute gains from 33.95% to 35.43% on Qwen3-8B-Base and from 42.77% to 43.90% on Llama-3.1-8B-Instruct. Second, these improvements compound iteratively; for Llama-3.1, a modest Round 1 gain amplifies significantly by Round 2, validating that the improved model acts as a stronger foundation for subsequent cycles.

Interestingly, our framework naturally targets different capability bottlenecks depending on the model's prior alignment. For the foundational Qwen3, it drives massive improvements in hard reasoning and strict formatting, such as AIME25 jumping from 7.19% to 12.40% and IF-iS increasing from 56.00% to 57.92%. Conversely, for the instruction-tuned Llama-3.1, the most prominent gain emerges in conversational alignment, with AlpacaEval LC surging from 24.12% to 27.86%.

Finally, compared to R-Zero, our approach demonstrates robust generalizability across capability axes. R-Zero achieves strong targeted gains in mathematical reasoning (e.g., AIME24 on Qwen3

[1] https://thinkingmachines.ai/tinker/

Table 2: Composition of the DPO pool after δ-filter on Qwen3-8B-Base R1.

Category	% Pool	Mean δ
Math	9.6%	0.045
Code	9.0%	0.040
Explain	9.6%	0.058
Advice	30.2%	0.045
Writing	17.4%	0.060
Other	24.1%	0.056

Table 3: δ-filter design space on Qwen3-8B-Base (R1). Absolute performance (%). The filter intervals $[x, y]$ denote the percentile range of the retained δ values.

δ filter	Chat	IFEval	Math	Avg.
base model	8.94	52.78	8.81	33.95
$[0, 25]$	9.04	52.92	11.58	34.84
$[0, 50]$ (G-Zero)	9.07	53.03	11.78	**34.96**
$[0, 75]$	9.09	53.06	11.15	34.80
$[20, 80]$	9.07	51.82	**12.54**	34.40
$[50, 100]$	**9.68**	51.97	10.37	34.04
$[0, 100]$	9.10	**53.08**	10.58	34.65

increasing from 10.42% to 14.92%), but trades off other capabilities: its conversational performance on Qwen3 drops from 8.94% to 8.04%, IFEval-pS drops from 43.07% to 37.56%, and its overall 7-metric average on Llama-3.1 falls from 42.77% to 40.89%. In contrast, G-Zero R2 keeps all per-metric movements small in magnitude on both models: on Qwen3 the only regression is AlpLC (-0.47), with all other six metrics improving; on Llama-Instruct every metric is positive at R2, with AlpLC itself gaining $+3.74$.

Beyond Table 1, Appendix E.6 reports five additional benchmarks (MATH, AMC, Minerva, OlympiadBench, and Arena-Hard v2) together with pairwise comparisons by two LLM judges and blind human annotators, and Appendix E.7 reports a three-seed repetition of the Qwen3-8B-Base Round-2 run.

5 Analysis

Structural Transfer from Non-Verifiable Tasks Table 2 shows that the final DPO pool is overwhelmingly dominated by non-verifiable tasks, with categories such as advice, writing, and others collectively accounting for over 70% of the training data. In contrast, verifiable tasks like math and code represent less than 19% of the pool. Crucially, the highest reward signals (δ) originate from structured writing and detailed explanations rather than math. This confirms that G-Zero's substantial reasoning improvements do not come from domain-specific memorization, but from internalizing the logical depth and compositional complexity of the Challenger's trajectories and successfully transferring these structural paradigms to mathematical problem-solving.

The Necessity of the Lower-Half Filter. Table 3 evaluates alternative δ selection strategies to validate our $[0, 50]$ filter. A failure-mode analysis of retained and discarded pairs (Appendix E.2) supports a mechanistic dichotomy: low-δ pairs capture structural and logical refinements, while high-δ pairs are more susceptible to "answer leakage," where the Proposer's hint explicitly provides the solution. The pattern in Table 3 is consistent with this dichotomy, though the magnitudes are modest. Retaining the upper half ($[50, 100]$) trades verifiable instruction following (IFEval drops by 0.81 pp from 52.78 to 51.97) for chat-style helpfulness (AlpacaEval LC reaches the highest value, 9.68), suggesting the Generator partially absorbs hint content directly rather than internalizing reasoning steps. Removing the filter entirely ($[0, 100]$) yields a slight overall improvement in two of the three domains (IFEval 53.08 vs. 53.03 for $[0, 50]$; Chat 9.10 vs. 9.07) at the cost of a 1.20 pp drop in Math, while the $[20, 80]$ middle band gives a higher Math score (12.54) but lower IFEval. Across the bottom-anchored bands $[0, 25]$ to $[0, 75]$, Chat, IFEval, and Math move by at most 0.05, 0.14, and 0.63 pp, respectively, so the method does not depend on the cutoff being exactly the median, whereas Math declines monotonically as higher-δ data are admitted ($11.78 \rightarrow 11.15 \rightarrow 10.58$). The $[0, 50]$ filter therefore offers the most balanced profile we tested rather than a strictly dominant configuration; we adopt it as the default but note that the surrounding band $[0, 50] \pm 30$ pp produces broadly comparable averages. Appendix E further reports the label accuracy of retained pairs, controlled hint conditions, the source of the Proposer signal, query diversity, additional benchmarks and evaluators, and seed variance.

Capability Scaling Dynamics. Figure 3 tracks the performance trajectory (Δ) across incremental DPO pool sizes ($N \in \{100, 200, 400, 730\}$) versus the global from-scratch optimization (Round 2). The scaling behaviors reveal how the Generator internalizes the structural paradigms elicited by

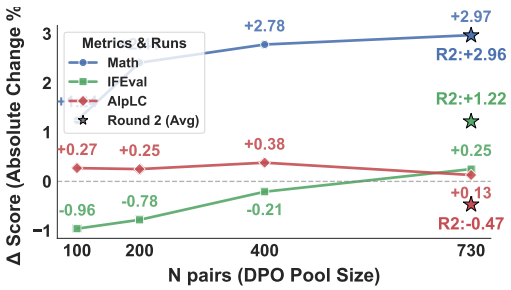

Figure 3: Performance change (Δ) relative to the base model across incremental DPO pool sizes ($N \in \{100, 200, 400, 730\}$) versus the from-scratch optimization in Round 2 (star at $N = 730$).

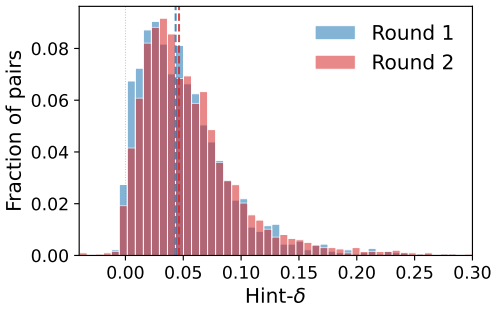

Figure 4: Empirical distributions of Hint-δ for Round 1 and Round 2. The dashed vertical lines denote the median δ for each round. The rightward shift in Round 2 demonstrates the co-evolutionary dynamic: as the Generator becomes more capable, the Proposer adapts by synthesizing increasingly impactful hints, thereby pushing the intrinsic learning signal to a higher baseline.

the Proposer. Mathematical reasoning (Math) exhibits rapid, monotonic gains and saturates early: $+1.24$ at $N = 100$ already covers more than 40% of the final $+2.97$ at $N = 730$, and the Round 2 from-scratch training matches the same final value ($+2.96$). This suggests the Generator absorbs the underlying logical structure with minimal data and quickly reaches its inherent capacity limit for structural reasoning. Verifiable instruction following (IFEval) shows the opposite shape, the incremental schedule starts at -0.96 ($N = 100$), monotonically recovers to a small positive ($+0.25$ at $N = 730$), and only the global Round 2 from-scratch optimization fully unlocks the capability at $+1.22$. General conversational helpfulness (AlpacaEval LC) is largely flat under incremental DPO — it stays in the $[+0.13, +0.38]$ band across all four checkpoints — while the Round 2 from-scratch model ends at -0.47 pp. The three capability axes therefore exhibit distinct, axis-specific saturation curves rather than a uniform trade-off, and Round 2 dominates incremental scaling on the IFEval and overall metric while the two coincide on Math.

The Shifting Distribution in Different Rounds. Figure 4 reveals a distinct rightward shift in the Hint-δ distribution from Round 1 to Round 2, accompanied by an increased median. Counter-intuitively, this baseline *increases* even though the Generator has become a stronger reasoner. This evidences a co-evolutionary arms race: because the upgraded Generator is no longer perturbed by trivial assistance, the Proposer must adapt by synthesizing profoundly complex hints to maximize its GRPO reward. By continuously uncovering newly elevated blind spots, the Proposer raises the difficulty ceiling and prevents the Generator from stagnating across iterations.

6 Related Work

6.1 Self-Evolving Language Models

Self-evolution enhances large language model (LLM) reasoning without human annotations. Early frameworks [11, 29] leveraged fine-tuning on high-confidence self-generated trajectories. This progressed into iterative self-play [2] and multi-role co-evolution pipelines [15, 1], which mitigate feedback saturation through cross-verification [4, 22]. Recently, the field has shifted toward dynamic self-challenging [40, 14, 17] and unsupervised post-training [30], marking a transition from supervised imitation [36] to intrinsically verifiable, zero-data frameworks [10, 37, 8, 13]. However, robust filtering remains essential, as unconstrained recursive training on synthetic outputs risks model collapse [25].

6.2 Verifier-Free RL

To address the reliance of Reinforcement Learning from Verifier Feedback (RLVR) on explicit rules, recent works explore verifier-free paradigms for open-ended domains by extracting intrinsic rewards directly from the generation process. Foundational methods [39, 18] bypass external verifiers by optimizing the conditional probability of reference answers. To stabilize training and prevent reasoning degradation (e.g., CoT shortening), likelihood-based designs are further refined into smooth, dense rewards that reduce gradient variance [12, 35]. Beyond final-answer probabilities, recent advancements construct step-wise optimization signals by leveraging internal hidden states as implicit verifiers [32] or modeling reasoning as a continuous probabilistic flow [19], establishing a foundation for self-evolution in completely unverifiable environments.

7 Conclusion

In this work, we introduced G-Zero, a verifier-free framework that enables LLMs to self-improve in open-ended and unverifiable domains. By replacing external judges with a single internal signal(**Hint-**δ)—our approach naturally measures both query difficulty and hint informativeness while avoiding the bottlenecks of traditional alignment. Through a continuous loop where a Proposer targets the model's blind spots and a Generator internalizes these hints, G-Zero drives autonomous self-evolution. Ultimately, this proves that models can elevate their capabilities using only intrinsic feedback, paving the way for self-aligning systems completely independent of human ground truth.

Acknowledgement

We gratefully acknowledge the Thinking Machines Lab Tinker Research Grant for supporting the experimental efforts of this work. We thank the anonymous reviewers and the meta-reviewer for their valuable feedback and constructive suggestions, which helped improve this paper.This material is based upon work supported in part by the National Science Foundation under Award No. 2541822 and AI for Health Grants. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

References

[1] Jiaqi Chen, Bang Zhang, Ruotian Ma, Peisong Wang, Xiaodan Liang, Zhaopeng Tu, Xiaolong Li, and Kwan-Yee K. Wong. Spc: Evolving self-play critic via adversarial games for llm reasoning, 2025. URL https://arxiv.org/abs/2504.19162.

[2] Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models, 2024. URL https://arxiv.org/abs/2401.01335.

[3] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

[4] Wenkai Fang, Shunyu Liu, Yang Zhou, Kongcheng Zhang, Tongya Zheng, Kaixuan Chen, Mingli Song, and Dacheng Tao. Serl: Self-play reinforcement learning for large language models with limited data, 2025. URL https://arxiv.org/abs/2505.20347.

[5] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[6] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on LLM-as-a-judge. *The Innovation*, 2024.

[7] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[8] Yicheng He, Chengsong Huang, Zongxia Li, Jiaxin Huang, and Yonghui Yang. Visplay: Self-evolving vision-language models from images. *arXiv preprint arXiv:2511.15661*, 2025.

[9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

[10] Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-Zero: Self-evolving reasoning LLM from zero data. *arXiv preprint arXiv:2508.05004*, 2025.

[11] Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve, 2022. URL https://arxiv.org/abs/2210.11610.

[12] Ariel Kwiatkowski, Natasha Butt, Ismail Labiad, Julia Kempe, and Yann Ollivier. Likelihood-based reward designs for general llm reasoning. *arXiv preprint arXiv:2602.03979*, 2026.

[13] Zongxia Li, Hongyang Du, Chengsong Huang, Xiyang Wu, Lantao Yu, Yicheng He, Jing Xie, Xiaomin Wu, Zhichao Liu, Jiarui Zhang, and Fuxiao Liu. Mm-zero: Self-evolving multi-model vision language models from zero data, 2026. URL https://arxiv.org/abs/2603.09206.

[14] Xiao Liang, Zhong-Zhi Li, Yeyun Gong, Yang Wang, Hengyuan Zhang, Yelong Shen, Ying Nian Wu, and Weizhu Chen. Sws: Self-aware weakness-driven problem synthesis in reinforcement learning for llm reasoning, 2025. URL https://arxiv.org/abs/2506.08989.

[15] Zi Lin, Sheng Shen, Ilia Kulikov, Jingbo Shang, Jason Weston, and Yixin Nie. Learning to solve and verify: A self-play framework for code and test generation. *arXiv preprint arXiv:2502.14948*, 2025.

[16] Bo Liu, Chuanyang Jin, Seungone Kim, Weizhe Yuan, Wenting Zhao, Ilia Kulikov, Xian Li, Sainbayar Sukhbaatar, Jack Lanchantin, and Jason Weston. Spice: Self-play in corpus environments improves reasoning. *arXiv preprint arXiv:2510.24684*, 2025.

[17] Fuxiao Liu, Xiaoyang Wang, Wenlin Yao, Jianshu Chen, Kaiqiang Song, Sangwoo Cho, Yaser Yacoob, and Dong Yu. Mmc: Advancing multimodal chart understanding with large-scale instruction tuning. 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 1287–1310, 2024.

[18] Wei Liu, Siya Qi, Xinyu Wang, Chen Qian, Yali Du, and Yulan He. Nover: Incentive training for language models via verifier-free reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7450–7469, 2025.

[19] Yan Liu, Feng Zhang, Zhanyu Ma, Jun Xu, Jiuchong Gao, Jinghua Hao, Renqing He, Han Liu, and Yangdong Deng. Efficient paths and dense rewards: Probabilistic flow reasoning for large language models. *arXiv preprint arXiv:2601.09260*, 2026.

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

[21] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

[22] Sheikh Shafayat, Fahim Tajwar, Ruslan Salakhutdinov, Jeff Schneider, and Andrea Zanette. Can large reasoning models self-train?, 2025. URL https://arxiv.org/abs/2505.21444.

[23] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[24] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

[25] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022): 755–759, 2024.

[26] Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. Large language models for data annotation and synthesis: A survey. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

[27] Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, et al. A survey on self-evolution of large language models. *ArXiv preprint*, abs/2404.14387, 2024.

[28] Xiaohua Wang, Muzhao Tian, Yuqi Zeng, Zisu Huang, Jiakang Yuan, Bowen Chen, Jingwen Xu, Mingbo Zhou, Wenhao Liu, Muling Wu, et al. Reward hacking in the era of large models: Mechanisms, emergent misalignment, challenges. *arXiv preprint arXiv:2604.13602*, 2026.

[29] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions, 2022.

[30] Lai Wei, Yuting Li, Chen Wang, Yue Wang, Linghe Kong, Weiran Huang, and Lichao Sun. First sft, second rl, third upt: Continual improving multi-modal llm reasoning via unsupervised post-training, 2025. URL https://arxiv.org/abs/2505.22453.

[31] Zhishang Xiang, Chengyi Yang, Zerui Chen, Zhimin Wei, Yunbo Tang, Zongpei Teng, Zexi Peng, Zongxia Li, Chengsong Huang, Yicheng He, et al. A systematic survey of self-evolving agents: From model-centric to environment-driven co-evolution, 2025.

[32] Changyi Xiao, Caijun Xu, and Yixin Cao. Reinforcement learning with conditional expectation reward. *arXiv preprint arXiv:2603.10624*, 2026.

[33] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[34] Zihao Yi, Jiarui Ouyang, Zhe Xu, Yuwen Liu, Tianhao Liao, Haohao Luo, and Ying Shen. A survey on recent advances in LLM-based multi-turn dialogue systems. *ACM Computing Surveys*, 58(6):1–38, 2025.

[35] Tianyu Yu, Bo Ji, Shouli Wang, Shu Yao, Zefan Wang, Ganqu Cui, Lifan Yuan, Ning Ding, Yuan Yao, Zhiyuan Liu, et al. Rlpr: Extrapolating rlvr to general domains without verifiers. *arXiv preprint arXiv:2506.18254*, 2025.

[36] Wenhao Yu, Zhenwen Liang, Chengsong Huang, Kishan Panaganti, Tianqing Fang, Haitao Mi, and Dong Yu. Guided self-evolving LLMs with minimal human supervision. *arXiv preprint arXiv:2512.02472*, 2025.

[37] Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data, 2025. URL https://arxiv.org/abs/2505.03335.

[38] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[39] Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang, Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin, and Chao Du. Reinforcing general reasoning without verifiers. *arXiv preprint arXiv:2505.21493*, 2025.

[40] Yifei Zhou, Sergey Levine, Jason Weston, Xian Li, and Sainbayar Sukhbaatar. Self-challenging language model agents, 2025. URL https://arxiv.org/abs/2506.01716.

A Prompts and Templates

Configuration: Proposer (Phase 1 GRPO and Phase 2 generation)

- **Model**: base policy (Qwen/Qwen3-8B-Base by default)
- **Temperature**: 1.0
- **Max tokens**: 8 192
- **System message**: *(none)*

User Message:

Produce one challenging request that a real user might ask a capable assistant, plus a short hint that helps the assistant give a noticeably better response.

The request should come from a general-domain distribution. Sample across task types, not from a single area. Examples of task types you can draw from:
- writing (email, story, essay, pitch, review, poem)
- explanation (make a concept clear to a specific audience)
- advice or planning (career, travel, project, learning)
- analysis (argument, text, dataset description, product)
- coding (small function, debugging, design question)
- role-play, dialogue, or creative tasks
- open-ended questions about ethics, science, everyday life
- reasoning, math, or logic problems (fine to include — roughly 1 in 6 requests, no more)

Weight the non-math categories above heavily. A little math is good for diversity, but it should not dominate — favor tasks where the response quality depends on tone, structure, audience-fit, clarity, or creativity, not just arithmetic correctness.

Requirements:
- The request must be self-contained and non-trivial to answer well.
- The hint must guide the approach (e.g. tone, structure, what to include, what to avoid) but must not give away the full answer.
- Wrap the request in <question> and </question> tags.
- Wrap the hint in <hint> and </hint> tags.
- Output nothing else before, between, or after the tagged blocks.

Example 1 (writing):
<question>Write a resignation email to my manager that keeps the door open for future collaboration. I've been at the company for 4 years and I'm leaving to join a competitor. Tone should be professional and warm without being effusive.</question>
<hint>Lead with gratitude for specific experiences rather than generic thanks, keep the departure reason brief and non-defensive, and close with a concrete offer to help during the transition.</hint>

Example 2 (explanation):
<question>Explain what a Kalman filter does to a software engineer who is comfortable with linear algebra but has never touched signal processing. Avoid control-theory jargon where possible.</question>
<hint>Frame it as 'recursive Bayesian least-squares on a noisy dynamical system' and build up from a 1-D example (estimating position from noisy GPS) before generalizing.</hint>

Now produce one new request and hint of your own:

Configuration: Generator (Phase 2 dual-response sampling)

- **Model**: round-R Generator checkpoint (base for round 1)
- **Temperature**: 0.7
- **Max tokens**: 16 384
- **System message**: *(none)*

User Message (a_{hard}, no-hint):

```
{question}
```

User Message (a_{assisted}, with hint):

```
{question}

Hint: {hint}
```

Configuration: AIME24 / AIME25 evaluation (mean@32)

- **Model**: trained Generator checkpoint
- **Temperature**: 0.7 (**not** 0; mean@k requires variance)
- **Samples per problem**: 32
- **Max tokens**: 16 384

System Message:

Please reason step by step, and put your final answer within \boxed{...}.

User Message:

```
{question}
```

Configuration: IFEval evaluation

- **Model**: trained Generator checkpoint
- **Temperature**: 0.0 (greedy)
- **Max tokens**: 16 384
- **Scorer**: rule-based verifier from `instruction_following_eval`

System Message:

You are a helpful assistant. Follow the user's instructions precisely.

User Message:

```
{prompt}
```

Configuration: AlpacaEval LC judge

- **Judge model**: `Qwen/Qwen3-235B-A22B-Instruct-2507`
- **Temperature**: 0.0
- **Max tokens**: 32
- **Position**: m/M randomized per pair to mitigate residual position bias

System Message:

You are a highly efficient assistant, who evaluates and selects the best large language model (LLM) based on the quality of their responses to a given

instruction. This process will be used to create a leaderboard reflecting the most accurate and human-preferred answers.

User Message Template:

I require a leaderboard for various large language models. I'll provide you with prompts given to these models and their corresponding outputs. Your task is to assess these responses, and select the model that produces the best output from a human perspective.

Instruction

```
{ "instruction": <prompt> }
```

Model Outputs

```
{ {"model_identifier": "m", "output": <out_m>},
{"model_identifier": "M", "output": <out_M>} }
```

Task

Evaluate the models based on the quality and relevance of their outputs, and select the model that generated the best output. Answer by providing the model identifier of the best model. ... Output only one of: `m` or `M`.

Best Model Identifier

B Pseudo-Code

Algorithm 1 One round of the G-Zero Co-Evolutionary Loop

Require: Proposer π_P, Generator π_G.
Phase 1: Proposer Training
Sample group rollouts $\{(q_i, h_i)\}_{i=1}^{K} \sim \pi_P$ using GRPO.
for each rollout i **do**
  Sample $a_{\text{hard}}^{(i)} \sim \pi_G(\cdot \mid q_i)$ from the (frozen) Generator.
  Compute the per-token mean Hint-δ signal δ_i via Eq. 3.
end for
Compute reward r_i from δ_i and the structural penalties (Eq. 5).
Update π_P by maximizing the group-relative advantage A_i.
Phase 2: Generator Training and Dataset Curation
for each $(q_j, h_j) \sim \pi_P$ **do**
  Sample unassisted $a_{\text{hard}}^{(j)} \sim \pi_G(\cdot \mid q_j)$ and assisted $a_{\text{assisted}}^{(j)} \sim \pi_G(\cdot \mid q_j, h_j)$.
  Compute the per-token mean Hint-δ signal δ_j via Eq. 3.
end for
Filter: Retain pairs in the lower 50% of the δ distribution (the lower-shift regime suitable for DPO).
Refine: Apply quality heuristic checks to construct dataset $\mathcal{D}_{R+1} = \{(x = q, y_w = a_{\text{assisted}}, y_l = a_{\text{hard}})\}$.
Initialize reference model π_{ref} as a frozen snapshot of π_G.
Update π_G via DPO loss $\mathcal{L}_{\text{DPO}}$ on $\mathcal{D}_{R+1}$ with reference model π_{ref}.
return Updated π_P and π_G for the next round.

C Hyperparameters and Configuration

Table 4 lists the full G-Zero configuration used throughout. All numbers are defaults unless otherwise noted in the experiment.

Table 4: Default G-Zero hyperparameters.

LoRA rank	32
Proposer batch size	128
Proposer group size (GRPO)	16
Proposer steps (Phase 1)	6
Proposer learning rate	4×10^{-5}
Proposer max_tokens	8,192
Hint length target L_{tgt}	200 chars
Hint length penalty λ_{len}	0.03
BLEU cluster merge threshold	0.5 (average linkage)
BLEU penalty scale	cluster fraction $\lvert C_i\rvert/\lvert B\rvert$ (no extra constant)
Normalize δ	False
Questions per round N	2,000
Generator sampling temperature	0.7
Generator max_tokens	16,384
δ cutoff $[p_{\text{low}}, p_{\text{high}}]$	$[0, 50]$ (*bot50*)
`chosen_max_chars`	10,000
`chosen_min_chars`	100
`chosen_rejected_ratio_max`	2.5
`chosen_repetition_zlib` threshold	0.15
DPO β	2.0
DPO learning rate	1×10^{-5}
DPO max steps	50
DPO batch size	8
DPO log-ratio normalization	length-normalized
Eval max_tokens	16,384

D Proof of Theorem 1

We analyze an idealized standard-DPO version of G-Zero. The key distinction from ordinary offline preference learning is that the preference data are not sampled from the target prompt distribution Q. Instead, at each round, an adaptive helper generates question–hint pairs (q, h), the current generator produces an unassisted response and a hint-assisted response, and a δ-filter retains a subset of these pairs for DPO. Our analysis mainly highlights the benefit of exploration for the proposer (implemented by the BLEU reward) and the quality of preference data (implemented by the Hint-δ and filtration).

Relation to the practical algorithm. The analysis concerns an idealized version, and the practical algorithm departs from it in four ways that differ in kind. (i) *LoRA* restricts the policy class without changing the objective, so the guarantee holds against the best LoRA-representable policy, plus an approximation term for the gap; no step requires full-rank updates, and the pull toward the reference model that LoRA induces works with, not against, the KL-proximity the analysis assumes. (ii) *Length-normalized DPO* replaces a sequence-level implicit reward with a per-token-average log-ratio, a reparameterization of the reward class; the Bradley–Terry step and the structure of the bound are unchanged, though the constants do not transfer and we do not claim that they do. The normalization exists to suppress length exploitation, one of the noise sources the bound is about. (iii) *Heuristic filtering* is not an unmodeled complication but the mechanism realizing the premise: the theorem is conditional on low pseudo-label noise, and the filter delivers it. The two premises pull against each other, since a tighter filter lowers noise but narrows coverage, a trade-off traced by Table 3. (iv) *Adaptive data generation* is the genuine gap: the theorem is a per-round statement that takes the round's data distribution as given and does not model the closed loop across rounds, in which δ compresses so that noise rises and dispersion falls so that coverage narrows (Appendix F). Both premises are measured rather than assumed: the label noise is roughly 0.25–0.35 (Appendix E.1), and coverage spans 429 clusters at Round 1 with none above 3.3% (Appendix E.5).

Helper-induced tuples. Let $\mathcal{Q}$, $\mathcal{H}$, and $\mathcal{A}$ denote the prompt, hint, and response spaces. Let Q be the target prompt distribution on which we evaluate the generator. At round t, the current generator is π_t. The helper policy κ_t, which is measurable with respect to the history before round t, samples a question–hint pair $(q, h) \sim \kappa_t$. The generator then samples an unassisted response $a^- \sim \pi_t(\cdot \mid q)$ and

a hint-assisted response $a^+ \sim \pi_t(\cdot \mid q, h)$, where $\pi_t(\cdot \mid q, h)$ denotes the same generator conditioned on the augmented input containing both the question and the hint.

Let $z = (q, h, a^+, a^-)$ denote the resulting tuple. The raw helper-induced tuple law at round t is denoted by P_t^0. The δ-filter is represented by a measurable indicator $F_t(z) \in \{0, 1\}$. In the actual algorithm, F_t is determined by the Hint-δ score and additional data-quality checks; for the proof, only the retained distribution matters. Assume $P_t^0(F_t = 1) > 0$, and define the retained tuple distribution as $P_t = P_t^0(\cdot \mid F_t = 1)$. Its prompt marginal is denoted by ρ_t. We collect m retained tuples $z_{t,1}, \ldots, z_{t,m} \sim P_t$ at round t, conditionally independently given the past.

Pairwise features and covariance. Let $\phi(q, a) \in \mathbb{R}^d$ be a feature map, and define the pairwise feature $w(z) = \phi(q, a^+) - \phi(q, a^-)$ for $z = (q, h, a^+, a^-)$. In G-Zero, the pseudo-label always declares a^+ preferred to a^-, so the pseudo-label is $\widetilde{Y} = +1$. We nevertheless keep the notation $\widetilde{Y} \in \{+1, -1\}$ because the proof is cleaner. The observed DPO feature is $\widetilde{x} = \widetilde{Y} w$. The clean Bradley–Terry label is $Y^\star \in \{+1, -1\}$, and the clean feature is $x^\star = Y^\star w$.

Let $\Sigma_1 = \lambda I_d$. After round t, define the empirical batch covariance $\widehat{M}_t = m^{-1} \sum_{i=1}^m w_{t,i} w_{t,i}^\top$ and update $\Sigma_{t+1} = \Sigma_t + \widehat{M}_t$. Since $\|w_{t,i}\|_2 \le 1$, we have $\mathrm{tr}(\widehat{M}_t) \le 1$.

Assumption 1 (Linear reward and bounded features)**.** *There exists $\theta^\star \in \mathbb{R}^d$ such that $R^\star(q, a) = \phi(q, a)^\top \theta^\star$. The parameter set is $\Theta = \{\theta \in \mathbb{R}^d : \|\theta\|_2 \le B\}$, and $\theta^\star \in \Theta$. For all q, a, $\|\phi(q, a)\|_2 \le 1/2$. Hence every pairwise feature satisfies $\|w(z)\|_2 \le 1$.*

Assumption 2 (Bradley–Terry clean preference model)**.** *For every retained tuple $z = (q, h, a^+, a^-)$, the clean label $Y^\star$ satisfies $\Pr(Y^\star = +1 \mid z) = \sigma(\theta^{\star\top} w(z))$, where $\sigma(u) = 1/(1 + \exp(-u))$. The hint h affects the sampling of a^+, but the reward comparison itself is between the two responses to the same question q.*

Assumption 3 (Standard-DPO policy class)**.** *For a fixed reference policy π_{ref}, the generator policy class is the Gibbs class $\pi_\theta(a \mid q) \propto \pi_{\mathrm{ref}}(a \mid q) \exp(\phi(q, a)^\top \theta)$, with $\theta \in \Theta$. The target policy is $\pi^\star \in \arg\max_{\theta \in \Theta} J_Q(\pi_\theta)$, where $J_Q(\pi) = \mathbb{E}_{q \sim Q, a \sim \pi(\cdot|q)}[R^\star(q, a)] - \mathbb{E}_{q \sim Q}[\mathrm{KL}(\pi(\cdot \mid q) \| \pi_{\mathrm{ref}}(\cdot \mid q))]$.*

Assumption 4 (δ-certified DPO score noise)**.** *For horizon T, the retained data produced by the δ-filter are $(\eta_\delta, \zeta_\delta)$-certified if, with probability at least $1 - \zeta_\delta$, for every $t \le T + 1$,*

$$\left\| \frac{1}{m} \sum_{s<t} \sum_{i=1}^m \mathbf{1}\{\widetilde{Y}_{s,i} \neq Y_{s,i}^\star\} \, Y_{s,i}^\star w_{s,i} \right\|_{\Sigma_t^{-1}} \le \sqrt{\eta_\delta}.$$

This is the exact pseudo-label noise quantity needed by DPO: wrong retained pairs are allowed, but they cannot concentrate in a low-coverage feature direction.

For any policy π, write $\bar{\phi}(q, \pi) = \mathbb{E}_{a \sim \pi(\cdot|q)}[\phi(q, a)]$. Define the target-direction gap at round t as $v_t(q) = \bar{\phi}(q, \pi^\star) - \bar{\phi}(q, \pi_t)$. Define the target uncertainty $\Psi_{Q,t}^2 = \mathbb{E}_{q \sim Q} \|v_t(q)\|_{\Sigma_t^{-1}}^2$, and the actual retained-batch exposure $\widehat{\Psi}_t^2 = m^{-1} \sum_{i=1}^m \|w_{t,i}\|_{\Sigma_t^{-1}}^2 = \mathrm{tr}(\Sigma_t^{-1} \widehat{M}_t)$.

Assumption 5 (Helper-induced coverage with target-distribution mismatch)**.** *There exist constants $C_Q \ge 1$ and $\alpha_{\mathrm{S}} \in (0, 1]$ such that for every round t, the helper-induced retained prompt marginal ρ_t dominates the target distribution Q, with $\|dQ/d\rho_t\|_\infty \le C_Q$, and the realized retained batch satisfies*

$$\widehat{\Psi}_t^2 \ge \alpha_{\mathrm{S}}^2 \mathbb{E}_{q \sim \rho_t} \|v_t(q)\|_{\Sigma_t^{-1}}^2.$$

The constant C_Q measures the mismatch between the helper-generated question distribution and the target question distribution. The constant α_{S} measures how well the generated question–hint pairs expose the feature directions along which the current generator differs from the target policy. This assumption enables the helper's generated data are exploratory enough.

Assumption 6 (Exact cumulative standard-DPO update)**.** *For $t \ge 2$, the generator parameter $\widehat{\theta}_t$ is obtained by cumulative regularized standard DPO on all retained pairs from rounds $1, \ldots, t - 1$:*

$$\widehat{\theta}_t \in \arg\min_{\theta \in \Theta} \left\{ \frac{1}{m} \sum_{s<t} \sum_{i=1}^m -\log \sigma(\theta^\top \widetilde{Y}_{s,i} w_{s,i}) + \frac{\kappa_{\mathrm{BT}} \lambda}{2} \|\theta\|_2^2 \right\},$$

where $\kappa_{\mathrm{BT}} = \min_{|u|\le B} \sigma(u)\sigma(-u)$. *We set* $\pi_t = \pi_{\widehat{\theta}_t}$. *For* $t = 1$, $\widehat{\theta}_1$ *is the minimizer of the ridge term over* Θ, *i.e.,* $\widehat{\theta}_1 = 0$.

Let $D_T = d\log(1 + T/(\lambda d))$, and define $s_{m,T,\zeta} = \sqrt{2(D_T + \log(T/\zeta))/m}$. Let $r = \frac{2}{\kappa_{\mathrm{BT}}}\left(s_{m,T,\zeta} + \sqrt{\eta_\delta}\right) + 2\sqrt{\lambda}B$.

Theorem 1. Under Assumptions 1–6, with probability at least $1 - \zeta - \zeta_\delta$, there exists $t_0 \le T$ such that

$$J_Q(\pi^\star) - J_Q(\pi_{t_0}) \le \frac{2r\sqrt{C_Q}}{\alpha_{\mathrm{S}}}\sqrt{\frac{d\log(1 + T/(\lambda d))}{T}}.$$

Consequently, choosing $T = \widetilde{\Theta}(C_Q d/\alpha_{\mathrm{S}}^2)$, $m = \widetilde{\Theta}(d/(\kappa_{\mathrm{BT}}^2\varepsilon^2))$, and $\sqrt{\lambda}B = \widetilde{O}(\varepsilon)$ gives total retained sample complexity

$$mT = \widetilde{O}\left(\frac{C_Q d^2}{\alpha_{\mathrm{S}}^2\kappa_{\mathrm{BT}}^2\varepsilon^2}\right),$$

and guarantees $J_Q(\pi^\star) - J_Q(\pi_{t_0}) \le \widetilde{O}(\varepsilon + \kappa_{\mathrm{BT}}^{-1}\sqrt{\eta_\delta})$.

We prove the theorem in five steps. First, we show how the helper-induced distribution mismatch produces the factor C_Q. Second, we prove a self-normalized concentration inequality for the clean Bradley–Terry logistic score. Third, we use it to prove the DPO confidence bound. Fourth, we convert parameter confidence into generator suboptimality on the target distribution Q. Finally, we use an elliptical-potential argument to obtain a best-iterate bound.

Lemma 2 (Coverage transfer from helper distribution to target distribution)**.** *Under Assumption 5, for every round* t,

$$\widehat{\Psi}_t \ge \frac{\alpha_{\mathrm{S}}}{\sqrt{C_Q}}\Psi_{Q,t}.$$

Proof. Let $f_t(q) = \|v_t(q)\|^2_{\Sigma_t^{-1}}$. Since $Q \ll \rho_t$ and $\|dQ/d\rho_t\|_\infty \le C_Q$, we have $\mathbb{E}_{q\sim Q} f_t(q) \le C_Q \mathbb{E}_{q\sim\rho_t} f_t(q)$. Hence $\mathbb{E}_{q\sim\rho_t} f_t(q) \ge C_Q^{-1}\Psi_{Q,t}^2$. Assumption 5 gives $\widehat{\Psi}_t^2 \ge \alpha_{\mathrm{S}}^2 \mathbb{E}_{q\sim\rho_t} f_t(q)$, so $\widehat{\Psi}_t^2 \ge \alpha_{\mathrm{S}}^2 C_Q^{-1}\Psi_{Q,t}^2$. Taking square roots proves the claim. □

Lemma 3 (Self-normalized concentration for the clean logistic score)**.** *Fix any adaptive sequence of pairwise features* $w_1, \dots, w_n$, *with* $\|w_k\|_2 \le 1$, *where each* w_k *is measurable before its clean Bradley–Terry label* $Y_k^\star$ *is drawn. Suppose* $\Pr(Y_k^\star = +1 \mid w_k) = \sigma(\theta^{\star\top} w_k)$. *Let* $V_n = \lambda I_d + \sum_{k=1}^n x_k x_k^\top$, *where* $x_k = w_k/\sqrt{m}$. *Then, with probability at least* $1 - \delta$,

$$\left\| \frac{1}{m}\sum_{k=1}^n \left(\sigma(\theta^{\star\top} w_k) - \mathbf{1}\{Y_k^\star = +1\}\right) w_k \right\|_{V_n^{-1}} \le \sqrt{\frac{2}{m}\log\frac{\det(V_n)^{1/2}}{\det(\lambda I_d)^{1/2}\delta}}.$$

Proof. Let $p_k = \sigma(\theta^{\star\top} w_k)$ and $\epsilon_k = p_k - \mathbf{1}\{Y_k^\star = +1\}$. Conditional on the past and on w_k, ϵ_k has mean zero and lies in $[-1, 1]$. Hence Hoeffding's lemma gives $\mathbb{E}[\exp(\gamma\epsilon_k) \mid \mathcal{F}_{k-1}, w_k] \le \exp(\gamma^2/2)$ for every $\gamma \in \mathbb{R}$.

Define $S_n = \sum_{k=1}^n \epsilon_k x_k$. For any fixed $a \in \mathbb{R}^d$, the process $\exp(a^\top S_n - \frac{1}{2}a^\top(\sum_{k=1}^n x_k x_k^\top)a)$ is a supermartingale. Integrating this supermartingale over $a \sim N(0, \lambda^{-1} I_d)$ and completing the square yields $\mathbb{E}[\sqrt{\det(\lambda I_d)/\det(V_n)}\exp(\|S_n\|^2_{V_n^{-1}}/2)] \le 1$. By Markov's inequality, with probability at least $1 - \delta$, $\|S_n\|^2_{V_n^{-1}} \le 2\log(\det(V_n)^{1/2}/(\det(\lambda I_d)^{1/2}\delta))$. Since $m^{-1}\sum_{k=1}^n \epsilon_k w_k = m^{-1/2} S_n$, the result follows. □

Lemma 4 (Cumulative standard-DPO confidence)**.** *Under Assumptions 1, 2, 4, and 6, with probability at least* $1 - \zeta - \zeta_\delta$, *for every* $t \le T + 1$,

$$\|\widehat{\theta}_t - \theta^\star\|_{\Sigma_t} \le r.$$

Proof. We work on the event from Assumption 4, which has probability at least $1-\zeta_\delta$. Fix $t \le T+1$, and define the cumulative DPO objective $\widehat{L}_t(\theta) = m^{-1}\sum_{s<t}\sum_{i=1}^{m} -\log\sigma(\theta^\top \widetilde{Y}_{s,i} w_{s,i}) + \kappa_{\mathrm{BT}}\lambda\|\theta\|_2^2/2$. Let $e_t = \widehat{\theta}_t - \theta^\star$.

For every $\theta \in \Theta$, every observed feature satisfies $|\theta^\top \widetilde{Y}_{s,i} w_{s,i}| \le B$. The scalar second derivative of $-\log\sigma(u)$ is $\sigma(u)\sigma(-u)$, so it is at least κ_{BT} on $[-B,B]$. Therefore, $\nabla^2\widehat{L}_t(\theta) \succeq \kappa_{\mathrm{BT}}\Sigma_t$ for all $\theta \in \Theta$.

By strong convexity, $\widehat{L}_t(\theta^\star) \ge \widehat{L}_t(\widehat{\theta}_t) + \langle \nabla\widehat{L}_t(\widehat{\theta}_t), \theta^\star - \widehat{\theta}_t\rangle + \kappa_{\mathrm{BT}}\|e_t\|_{\Sigma_t}^2/2$. Since $\widehat{\theta}_t$ minimizes $\widehat{L}_t$ over the convex set Θ, the constrained first-order condition gives $\langle \nabla\widehat{L}_t(\widehat{\theta}_t), \theta^\star - \widehat{\theta}_t\rangle \ge 0$. By convexity at $\theta^\star$, $\widehat{L}_t(\widehat{\theta}_t) \ge \widehat{L}_t(\theta^\star) + \langle \nabla\widehat{L}_t(\theta^\star), e_t\rangle$. Combining these inequalities yields $\kappa_{\mathrm{BT}}\|e_t\|_{\Sigma_t}^2/2 \le -\langle \nabla\widehat{L}_t(\theta^\star), e_t\rangle$. Thus, by Cauchy–Schwarz, $\|e_t\|_{\Sigma_t} \le 2\kappa_{\mathrm{BT}}^{-1}\|\nabla\widehat{L}_t(\theta^\star)\|_{\Sigma_t^{-1}}$.

It remains to bound the score at $\theta^\star$. Decompose it as $G_t + C_t + \kappa_{\mathrm{BT}}\lambda\theta^\star$, where $G_t = m^{-1}\sum_{s<t}\sum_i \nabla\ell(\theta^\star; Y_{s,i}^\star w_{s,i})$, $C_t = m^{-1}\sum_{s<t}\sum_i[\nabla\ell(\theta^\star; \widetilde{Y}_{s,i} w_{s,i}) - \nabla\ell(\theta^\star; Y_{s,i}^\star w_{s,i})]$, and $\ell(\theta; x) = -\log\sigma(\theta^\top x)$.

For the clean term, $\nabla\ell(\theta^\star; Y^\star w) = (\sigma(\theta^{\star\top} w) - \mathbf{1}\{Y^\star = +1\})w$. Applying Lemma 3 at all round endpoints and taking a union bound with failure probability ζ/T, we obtain simultaneously for all $t \le T+1$ that $\|G_t\|_{\Sigma_t^{-1}} \le s_{m,T,\zeta}$. Here we used $\mathrm{tr}(\Sigma_t) \le \lambda d + T$ and AM–GM to get $\det(\Sigma_t)/\det(\lambda I_d) \le (1 + T/(\lambda d))^d$.

For the corruption term, if $\widetilde{Y}_{s,i} = Y_{s,i}^\star$, the summand is zero. If $\widetilde{Y}_{s,i} \ne Y_{s,i}^\star$, then $\widetilde{Y}_{s,i} w_{s,i} = -Y_{s,i}^\star w_{s,i}$, and a direct calculation gives $\nabla\ell(\theta^\star; -Y_{s,i}^\star w_{s,i}) - \nabla\ell(\theta^\star; Y_{s,i}^\star w_{s,i}) = Y_{s,i}^\star w_{s,i}$. Therefore Assumption 4 implies $\|C_t\|_{\Sigma_t^{-1}} \le \sqrt{\eta_\delta}$. Finally, since $\Sigma_t \succeq \lambda I_d$, we have $\|\kappa_{\mathrm{BT}}\lambda\theta^\star\|_{\Sigma_t^{-1}} \le \kappa_{\mathrm{BT}}\sqrt{\lambda}B$.

Combining the three score bounds gives $\|\nabla\widehat{L}_t(\theta^\star)\|_{\Sigma_t^{-1}} \le s_{m,T,\zeta} + \sqrt{\eta_\delta} + \kappa_{\mathrm{BT}}\sqrt{\lambda}B$. Substituting into $\|e_t\|_{\Sigma_t} \le 2\kappa_{\mathrm{BT}}^{-1}\|\nabla\widehat{L}_t(\theta^\star)\|_{\Sigma_t^{-1}}$ proves the desired bound. □

Lemma 5 (Target suboptimality from parameter confidence)**.** *On the event of Lemma 4, for every $t \le T$, $J_Q(\pi^\star) - J_Q(\pi_t) \le r\Psi_{Q,t}$.*

Proof. For any θ, let $J_{Q,\theta}(\pi)$ denote the same KL-regularized value as J_Q, but with reward $\phi(q,a)^\top\theta$ instead of $\phi(q,a)^\top\theta^\star$. By the Gibbs variational identity, $\pi_{\widehat{\theta}_t}$ maximizes $J_{Q,\widehat{\theta}_t}$, so $J_{Q,\widehat{\theta}_t}(\pi_t) \ge J_{Q,\widehat{\theta}_t}(\pi^\star)$. Therefore $J_Q(\pi^\star) - J_Q(\pi_t) \le \mathbb{E}_{q\sim Q}[(\theta^\star - \widehat{\theta}_t)^\top v_t(q)]$. Applying Cauchy–Schwarz in the Σ_t-norm gives $J_Q(\pi^\star) - J_Q(\pi_t) \le \|\widehat{\theta}_t - \theta^\star\|_{\Sigma_t}(\mathbb{E}_{q\sim Q}\|v_t(q)\|_{\Sigma_t^{-1}}^2)^{1/2} \le r\Psi_{Q,t}$. □

Lemma 6 (Elliptical potential for helper-generated batches)**.** *For any adaptive sequence of retained batches satisfying $\|w_{t,i}\|_2 \le 1$,*

$$\sum_{t=1}^{T} \log(1 + \widehat{\Psi}_t^2) \le d\log\left(1 + \frac{T}{\lambda d}\right).$$

Proof. Let $A_t = \Sigma_t^{-1/2}\widehat{M}_t\Sigma_t^{-1/2}$. Since $A_t \succeq 0$, $\det(I + A_t) \ge 1 + \mathrm{tr}(A_t) = 1 + \widehat{\Psi}_t^2$. Hence $\log\det(\Sigma_{T+1}) - \log\det(\Sigma_1) = \sum_{t=1}^{T}\log\det(I + A_t) \ge \sum_{t=1}^{T}\log(1+\widehat{\Psi}_t^2)$. On the other hand, $\mathrm{tr}(\Sigma_{T+1}) \le \lambda d + T$, so AM–GM gives $\det(\Sigma_{T+1}) \le (\lambda + T/d)^d$, while $\det(\Sigma_1) = \lambda^d$. Combining the lower and upper determinant bounds proves the claim. □

Proof of Theorem 1. By Lemma 2, $\widehat{\Psi}_t^2 \ge \alpha_{\mathrm{S}}^2 C_Q^{-1}\Psi_{Q,t}^2$ for every t. Lemma 6 therefore implies

$$\sum_{t=1}^{T} \log\left(1 + \frac{\alpha_{\mathrm{S}}^2}{C_Q}\Psi_{Q,t}^2\right) \le D_T.$$

Thus there exists $t_0 \le T$ such that $\log(1 + \alpha_S^2 C_Q^{-1} \Psi_{Q,t_0}^2) \le D_T/T$. Equivalently, $\Psi_{Q,t_0} \le \sqrt{C_Q}\alpha_S^{-1}\sqrt{\exp(D_T/T) - 1}$. Applying Lemma 5 at t_0 gives the first claim.

If $D_T/T \le 1$, then $\exp(D_T/T) - 1 \le 2D_T/T$, and we relax the constant $\sqrt{2}$ to 2, obtaining the simplified bound. Finally, choosing $T = \widetilde{\Theta}(C_Q d/\alpha_S^2)$, $m = \widetilde{\Theta}(d/(\kappa_{BT}^2 \varepsilon^2))$, and $\sqrt{\lambda}B = \widetilde{O}(\varepsilon)$ gives $r = \widetilde{O}(\varepsilon + \kappa_{BT}^{-1}\sqrt{\eta_\delta})$, and the stated total retained sample complexity follows. □

E Additional Analyses

E.1 Label Accuracy of Retained DPO Pairs

We audited the entire set of retained DPO pairs with two LLM judges external to our pipeline (Qwen3-235B and DeepSeek-V3.1), together with 50 blind human annotations per model. Table 5 reports how often the chosen (hint-assisted) response is preferred. The chosen response is preferred in 65.3% to 75% of pairs: the labels are noisy but majority-correct, which is the property DPO relies on, since a signal that is right most of the time moves the policy in the right direction even when individual pairs are wrong. The implied noise rate of roughly 0.25 to 0.35 is the quantity in terms of which our guarantee (Theorem 1) bounds suboptimality.

On math-type queries, where correctness is checked rather than judged, a confident but wrong answer cannot fool the label (Table 6). The balance of pairs have equal correctness. Among the pairs whose correctness differs, the chosen response is the correct one in 88.9% (Qwen3-8B-Base) and 78.5% (Llama-3.1-8B-Instruct) of cases.

Table 5: Share of retained DPO pairs (%) in which the chosen response is preferred.

	Qwen3-235B	DeepSeek-V3.1	Human (n=50)
Qwen3-8B-Base	72.5	70.6	75.0
Llama-3.1-8B-Instruct	65.3	71.7	74.0

Table 6: Correctness of math-type pairs (%).

	Chosen correct, rejected not	Rejected correct, chosen not
Qwen3-8B-Base	56	7
Llama-3.1-8B-Instruct	62	17

E.2 Failure Modes of High-δ Hints

We classified 250 discarded upper-half and 250 retained lower-half pairs with Qwen3-235B as the classifier, along four failure modes (Table 7). Every mode is significantly rarer among retained pairs: hints that maximize δ are disproportionately the hacked ones, so discarding the top half removes them faster than ordinary ones. The modes differ in consequence. Misleading hints can invert a DPO pair and are the main contributor to the label noise in Appendix E.1; leakage usually yields a correct response but teaches copying instead of reasoning; templated hints mostly change style, not labels. Filtering removes the modes that matter most, although 35.6% of retained pairs still carry some flaw and 10.4% still leak.

Table 7: Failure modes of hints (%) among discarded upper-half and retained lower-half pairs.

Failure mode	Upper half, discarded	Lower half, retained	p
Answer leakage	19.6	10.4	0.006
Misleading / wrong	26.8	13.6	< 0.001
Generic template	27.6	16.0	0.002
Any of the above	47.6	35.6	0.008

E.3 Controlled Hint Conditions

To test whether a high δ indicates informative guidance rather than mere distribution shift, we evaluated every retained pair under several hint conditions, keeping the queries fixed and varying only the hint (Table 8). A noisy string reaches a mean δ of 0.043 and an adversarial hint (stylistically strong but unhelpful) 0.049, against 0.052 for the trained Proposer, so a high δ on its own does not certify that a hint is informative. What separates the conditions is how often the assisted response is

actually preferred: adversarial and noisy hints sit at chance (48.2% and 45.1%), while the trained Proposer is the only condition high on both axes, its 71.4% matching the audit in Appendix E.1. This clarifies the role of δ: it is the Proposer's exploration signal, locating queries where the Generator's output is most movable, while certifying quality is the job of the lower-half filter and the DPO construction.

Table 8: Mean δ and preference for the assisted response under controlled hint conditions.

Hint condition	Mean δ	Assisted response preferred (%)
Trained Proposer (G-Zero)	0.052	71.4
Adversarial (strong style, unhelpful)	0.049	48.2
Noisy strings	0.043	45.1
Random unrelated	0.036	54.8
Untrained Proposer	0.021	54.3
Generic fixed hint	0.016	51.5

E.4 Source of the Proposer Signal

Table 9 holds the DPO stage fixed and changes only the source of the query–hint data (Qwen3-8B-Base, Round 1). An untrained Proposer with Phase-1 GRPO skipped lifts Math somewhat but not the average, falling 1.12 points short of the full pipeline, so the gains are not explained by the presence of hints or by extra synthetic preference data alone. Standardizing δ within each batch, the closest variant we have to a Proposer trained without a δ signal since z-scoring destroys the absolute scale that identifies genuinely movable queries, recovers the base level but almost none of the gain. Replacing the Proposer's queries with human-written UltraFeedback prompts recovers most of the average but loses roughly 40% of the Math gain, so a substantial part of the reasoning improvement comes from the queries being aimed at this Generator rather than merely being open-ended. Among the heuristic design choices, batch-normalizing δ and a length penalty with $\lambda = 0.03$ both fall below the base model on the seven-metric average (33.69 and 33.68 against 33.95), and OPD in place of DPO gives 34.36 against 34.96.

Table 9: Source of the query–hint data, with the DPO stage held fixed (Qwen3-8B-Base, Round 1).

Variant	Chat	IFEval	Math	Avg.
base model	8.94	52.78	8.81	33.95
Untrained Proposer, Phase-1 GRPO skipped	8.12	52.32	9.73	33.84
Proposer reward standardized (δ z-scored per batch)	8.96	52.83	9.01	34.04
Queries replaced by UltraFeedback prompts	9.21	53.34	10.63	34.83
Full G-Zero (Round 1)	9.07	53.03	11.78	34.96

E.5 Query Diversity

Because the BLEU penalty controls surface-level repetition, we also measured diversity in embedding space. We embedded the Proposer-generated queries and compared them against three human-written instruction sets at matched sample size (Table 10). At this clustering threshold the human corpora sit near the ceiling of 500 clusters, so the cluster column reads as a near-duplication rate: 14.2% of G-Zero queries are absorbed into a merge, against 0.8% (AlpacaEval), 2.6% (IFEval), and 3.6% (UltraFeedback). The largest cluster covers 3.3% of queries in R1, 3.8% in R2, and 3.0% combined, so the distribution does not fall back onto the task types listed in the seed prompt; in a separate mode-collapse run we observed, the same statistic rises to 55.9%. R2 explores new regions rather than repeating R1: the median cosine similarity from each R1 query to its nearest R2 neighbor is 0.608, and only 3.8% exceed 0.9. Dispersion is of the same order as the human sets but lower (379 effective clusters against 473–495); the mean cosine is lower, i.e., more dispersed, than IFEval's, although IFEval is higher on the cluster measures, and we report both metric families. Between rounds, R2 is slightly less dispersed than R1 (351.3 vs. 379.3 effective clusters; largest cluster 3.8% vs. 3.3%).

Table 10: Query diversity in embedding space at matched sample size.

Corpus	Mean cosine	Clusters	Largest cluster (%)	$\exp(H)$
G-Zero R1	0.114	429	3.3	379.3
G-Zero R2	0.131	409	3.8	351.3
G-Zero R1+R2	0.124	421	3.0	371.7
UltraFeedback	0.059	482	0.9	473.1
AlpacaEval	0.047	496	0.4	494.8
IFEval	0.176	487	0.4	481.8

E.6 Additional Benchmarks and Evaluators

Table 11 evaluates Qwen3-8B-Base after two self-play rounds on five additional benchmarks. All five improve, with an average gain of $+2.94$, including Arena-Hard v2, which is generally regarded as less saturated than AIME24/25 and IFEval. The gains concentrate on verifiable benchmarks, where style cannot help, while AlpacaEval LC, the most style-susceptible benchmark, is the one that declines (Table 1); this runs opposite to what stylistic inflation would predict.

We also compared G-Zero R2 against the base model on the same set of open-ended prompts using two independent LLM judges and blinded human evaluation with randomized response order (Table 12). All three evaluations favor G-Zero, with the human result lying between the two LLM judges. However, G-Zero responses are approximately 25% longer than those of the base model, and pairwise preference evaluations are known to favor longer responses, so part of this win rate may reflect length rather than capability; we therefore view the gains on the verifiable benchmarks as stronger evidence.

Table 11: Additional benchmarks (%), Qwen3-8B-Base after two rounds.

	MATH	AMC	Minerva	Olympiad	Arena-Hard v2
base model	57.80	54.73	24.82	29.70	8.92
G-Zero R2	61.60	56.68	27.03	32.14	13.22

Table 12: Win rate (%) of G-Zero R2 against the base model.

Qwen3-235B	DeepSeek-V3.1	Human
78	69	75

E.7 Seed Variance

We repeated the Qwen3-8B-Base Round-2 run with three random seeds (Table 13). The seed-to-seed variation is small: at most 1.87 and below 1.1 on four of the five metrics.

Table 13: Three-seed results (%) for Qwen3-8B-Base at Round 2.

	AlpLC	IF-iS	IF-iL	AIME24	AIME25
base model	8.94	56.00	61.75	10.42	7.19
G-Zero R2 (3-seed mean)	8.51	58.19	63.25	10.85	12.18
Std. over 3 seeds	0.14	1.08	0.00	0.37	1.87

F Limitation

The total Tinker compute consumed across all reported runs (two models, two rounds each, the cutoff and Phase 1 ablations, the data-scaling sweep, and the temperature-0.7 AIME re-evaluation) is on the order of US$2,000, so every cell in Table 1 is a single end-to-end run. We repeated the Qwen3-8B-Base Round-2 run with three seeds (Appendix E.7); the seed-to-seed standard deviation is at most 1.87 pp and below 1.1 pp on four of the five metrics reported there, but multi-seed results for the remaining cells, and hence tighter per-cell error bars (relevant for AIME24/25, which have only $n = 30$ unique problems and a $1\sigma \approx 8$ pp at $p \approx 0.3$), are left as future work. Separately, we cap our reported results at $R = 2$: an exploratory R3 run on Llama-3.1-8B-Instruct collapses, with the Phase 2 quality filter rejecting all $1{,}994$ candidate pairs because the R2-trained Generator has

converged to responses too short to satisfy the `chosen_min_chars` bound. We attribute this in part to a reward-hacking pathway: once the Generator has internalized the most accessible structural improvements, the Proposer continues maximizing Hint-δ via increasingly idiosyncratic hint patterns whose effect on $\log \pi_G(a_{\text{hard}})$ is large but no longer corresponds to genuinely informative guidance. The per-token mean structure of length-normalized DPO (Eq. 6) further amplifies the resulting length collapse, since shortening the chosen response mechanically raises the average per-token log-ratio. Multi-round-stable variants are an open direction we leave to future work.

A complementary account of this collapse follows from how δ is defined. Because δ is measured against the current Generator, the set of queries that a hint can still move shrinks as the Generator internalizes hint-guided behavior, and the δ distribution compresses from round to round. The Proposer is pushed onto the residual movable regions, and query dispersion declines (from 379.3 to 351.3 effective clusters between R1 and R2; Appendix E.5). More damaging, the lower half of a compressed distribution fills with pairs whose assisted and unassisted responses barely differ, so their preference signal approaches noise exactly as the filter admits more of them. Since our guarantee bounds suboptimality by per-round pseudo-label noise, composing rounds on a degrading signal is where it stops being informative. This account also predicts that the already instruction-tuned Llama-3.1-8B-Instruct, which starts closer to its ceiling, compresses sooner. We do not have a validated fix; three changes follow from this account as a plan rather than a result: anchoring the DPO reference to the original base model in every round, so that the KL term bounds cumulative drift rather than only the per-round step; stopping once the median δ of retained pairs falls below a threshold; and monitoring the largest-cluster share of Proposer queries, which is 3–4$\%$ in our runs but reached 55.9% in a separate mode-collapse run we observed.

Reward hacking of the Proposer is reduced but not eliminated. Among the retained lower-half pairs, 35.6% still carry a flawed hint and 10.4% still leak the answer (Appendix E.2); an explicit anti-leakage term in the Proposer reward is a natural next step. Relatedly, the retained DPO labels are noisy but majority-correct, with a noise rate of roughly 0.25–0.35 (Appendix E.1). Finally, δ could be combined with verifiable rewards where a verifier exists, either by keeping only DPO pairs on verifiable queries in which the assisted response is correct and the unassisted one is not, or by using a verifiable Proposer reward and falling back to δ elsewhere; we have not tested whether this improves on RLVR in verifiable domains.

Broader Impacts

This research introduces G-Zero, a verifier-free framework for autonomous LLM self-evolution. The broader implications of this work are summarized below:

Positive Societal Impacts: (1) **Democratizing AI Alignment:** By replacing expensive human labeling and proprietary judge-based APIs with intrinsic predictive signals, our framework significantly lowers the financial and computational barriers to high-level model alignment. This enables the open-source community and academic institutions to develop advanced reasoning models without relying on centralized, closed-source infrastructure. (2) **Advancing Scalable Oversight:** As AI systems approach and eventually surpass human expertise in complex tasks, providing external supervision becomes increasingly difficult. G-Zero demonstrates a viable path toward "scalable oversight," where models can autonomously identify and rectify their own logical blind spots without requiring human-curated ground truth.

Potential Risks and Mitigations: (1) **Value Drift:** Since the self-evolution process is driven by internal distributional shifts rather than direct human feedback, the model may experience "value drift," where it prioritizes complex structural depth at the expense of general helpfulness or safety alignment. To mitigate this, we recommend that models undergo a final safety validation phase or be constrained by lightweight human-in-the-loop checkpoints. (2) **Dual-Use Risks:** The ability to autonomously refine capabilities in open-ended domains could potentially be exploited by malicious actors to iteratively improve harmful outputs, such as sophisticated social engineering or cyberattack scripts. We emphasize that this framework should be applied to base models that already possess robust safety alignment, and researchers should implement output filtering to ensure the self-improvement remains within ethical boundaries.

G Case Study

Question: Create a persuasive pitch email to a non-profit organization advocating for increased funding for public school art programs. Highlight the dual benefits of fostering creativity and improving academic performance. Audience: staunch supporter of education but business-focused, with limited time for reading. Aim for a conversational tone that's concise but impactful.

Hint ($\delta = +0.056$)**:** Start with a compelling anecdote or statistic that grabs attention, frame the request as a strategic investment rather than a cost, and emphasize specific, measurable outcomes that align with the organization's priorities.

a_{hard} ***(no-hint response)*****:**
Subject: Elevate Our Community's Future: A Call for Art in Education
Hi [Prospective Donor's Name],
I hope this message finds you well. I am reaching out on behalf of [Your Organization], a group dedicated to enriching our local educational landscape through innovative approaches.
We believe that by investing in the arts within our public school system, we are not only nurturing the creative spirits of our youth but also enhancing their overall academic achievements. Recent studies have shown that students involved in the arts score higher on standardized tests, exhibit greater critical thinking skills, and show improved attendance rates. These statistics are compelling evidence that supports the integration of arts education into the core curriculum.
Our proposal is simple yet profound: allocate a portion of our budget to bolster existing art programs or establish new ones where they are currently lacking. This investment will act as a catalyst for holistic development, ensuring that every child has access to an enriching educational experience that goes beyond traditional academics.
We understand that your time is valuable, which is why we aim to make this conversation both impactful and efficient. If you're interested in learning more about how your support can transform lives and communities, please let me know. Together, we can create a brighter future for our children—one brushstroke at a time.
Thank you for considering this vital initiative. I look forward to discussing this further.
Best regards,
[Your Name]
[Your Position]
[Your Organization]
[Contact Information]

a_{assisted} ***(hint-conditioned response)*****:**
Subject: Invest in Art. Transform Lives.
Dear [Recipient's Name],
Imagine a classroom where the walls are covered with student artwork, not just in the hallways, but in every corner—a place where creativity and imagination run wild. This isn't just a dream; it's a reality for students when their schools prioritize art programs. Did you know that students who participate in the arts are 4 times more likely to be recognized for academic achievement? (Source: Americans for the Arts)
Now, think about it this way: investing in arts education isn't just about supporting the arts. It's about equipping students with critical thinking, problem-solving, and communication skills that are invaluable in any career. In fact, studies show that students with access to arts education perform better on standardized tests and are more likely to graduate high school.
Here's what we can achieve with your support:
1. **Enhanced Academic Performance**: Students in art-rich environments show improved math and reading scores. 2. **Increased Engagement**: Art classes keep students engaged, reducing behavioral issues and truancy. 3. **Career Readiness**: Exposure to various art forms prepares students for a wide range of careers, from graphic design to engineering.
By supporting our public school art programs, you're not just investing in the arts—you're investing in the future. This is a strategic move that aligns with your commitment to fostering talented, innovative, and well-rounded individuals.
Let's work together to make this vision a reality. Your contribution can have a profound impact on the lives of countless students. We look forward to discussing this further and exploring how we can collaborate to bring about meaningful change.
Warm regards,
[Your Name]
[Your Position]
[Your Organization]
[Contact Information]

Feel free to customize this pitch to better fit your organization's voice and the specific needs of your audience.

Figure 5: An illustrative (q, h, a_{hard}, a_{assisted}) pair from G-Zero R1 on Qwen3-8B-Base. The hint specifies three structural improvements (anecdote/statistic, investment-not-cost framing, measurable outcomes); a_{assisted} applies all three, while a_{hard} defaults to a generic template.

Question: Design a memorable slogan for a new line of eco-friendly diapers that appeals to environmentally conscious parents between the ages of 25 and 40, highlighting both sustainability and comfort.

Hint ($\delta = +0.031$): Focus on creating a slogan that balances the environmental benefits of the product with its user-friendly convenience, using language that evokes positive emotions and a connection to nature without being overly technical or abstract.

a_{hard} ***(no-hint response)*:**
Certainly! Here are a few slogan ideas for an eco-friendly diaper line targeting environmentally conscious parents aged 25-40:
"Soft Comfort, Pure Earth."
"Nature's Touch, Gentle on Bums."
"Sustainable Softness for Little Ones."
"Eco-Friendly, Eco-Friendly, and Still Gentle."
"Green Comfort, Happy Baby, Happy Planet."
"Sustainable Solutions for Sensible Parents."
"Earth-Friendly, Baby-Friendly, Always Gentle."
"Pure Comfort, Pure Planet."
"Nature's Embrace, Your Baby's Best Choice."
"Eco-Conscious, Comfort-Aware, Baby's Delight."

a_{assisted} ***(hint-conditioned response)*:**
Here's a potential slogan:
"Soft on Skin, Kind to Earth - EcoGentle Diapers"
This slogan aims to balance the environmental benefits of the product ("Kind to Earth") with its user-friendly convenience ("Soft on Skin"). The phrase "EcoGentle" creates a positive emotional association with nature while also conveying the gentle, comfortable nature of the diapers. It's simple, memorable, and conveys a clear message that resonates with environmentally conscious parents.

Figure 6: Second example. The hint asks for a slogan balancing sustainability with comfort using emotionally resonant language. a_{hard} throws ten generic options without commentary; a_{assisted} commits to a single slogan with a brand name (*EcoGentle*) and explains why it satisfies the brief.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflects the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed it in Section F

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We discuss in App. D.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We show details in Sec C

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: All dataset are opensource dataset, and we will release the codes once accepted.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We shown them in Sec. C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The experiments is quite expensive so we just show results in different models/datasets to show the effectiveness of our method.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We explain it in the Sec. F and Experiment details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We follow the code of ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss them in Sec. F.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We will check the misinfo before we public our model

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All models and benchmarks used in the paper are public.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No human in the loop.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human in the loop.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: No important components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.