CONSTRAINT-DECOUPLED ON-POLICY DISTILLATION FOR MULTI-CONSTRAINT INSTRUCTION FOLLOWING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-constraint instruction following requires a model to satisfy every constraint in an instruction simultaneously. We show that the dominant failure mode of current LLMs is partial satisfaction: the all-satisfy rate (ASR) collapses as constraints accumulate, while the per-constraint satisfaction rate stays above 89%. Neither scalar RLVR nor standard on-policy distillation repairs this, because both discard constraint-level structure: RLVR compresses all constraint evaluations into a single reward, and on-policy distillation conditions the teacher on the full instruction, so the student inherits the teacher's own compositional decay. We propose Constraint-Decoupled On-Policy Distillation (CD-OPD), which decouples the instruction into single-constraint prompts, scores each constraint's token-level effect against an unconstrained base, and combines them through a differentiable min-pooling operator. This operator acts as a soft veto—a token violating *any* constraint is suppressed—while a full-instruction anchor preserves fluency, and the resulting target is distilled into the student along its own on-policy trajectories. Across three teacher–student scales(0.6/1.7/4B) and four benchmarks, CD-OPD improves average ASR over the base student by 4.48%/2.54%/3.57%, reduces the FollowBench decay from 36.6% to 24.5% over one to five constraints, while training substantially faster than RLVR.

1 INTRODUCTION

Modern large language models (LLMs) are expected to follow complex user instructions that go far beyond simple task completion (OpenAI, 2026a). In real-world deployments, a single instruction typically couples a base task with multiple heterogeneous constraints spanning format, length, style, and semantics (e.g., *"Write a summary in bullet points, under 100 words, using formal tone"*). We refer to this setting as *multi-constraint instruction following*: the model must simultaneously accomplish the base task and satisfy all attached constraints. This conjunctive nature makes multi-constraint IF a stringent and practically critical capability for deploying LLMs in safety-sensitive, format-sensitive, and specification-heavy agent applications (OpenAI, 2026b; Anthropic, 2025).

Despite its importance, multi-constraint IF remains an open challenge. The two dominant alignment paradigms both discard the structure of individual constraints. Reinforcement learning with verifiable rewards (RLVR), which succeeds in domains such as mathematical reasoning (Zeng et al., 2026a; Liu et al., 2025), collapses every constraint evaluation into a single scalar reward, so the training signal no longer indicates which constraint a response violates. On-policy distillation (OPD) (Song & Zheng, 2026; Yang et al., 2026) instead offers dense token-level supervision, but conditions the teacher on the full instruction at once, asking it to resolve all constraints jointly. It is therefore unclear whether either paradigm can enforce the strict conjunction that multi-constraint IF demands.

We investigate this question in a three-part pilot study, and both paradigms fall short. **(i) The bottleneck is compositional.** Across four advanced LLMs, the all-satisfy rate (ASR) collapses with the number of constraints, falling from 90–98% at one constraint to 69–77% at the highest counts for three of the four models. Even the strongest model loses 6.5 points. The per-constraint satisfaction rate, by contrast, stays above 89%. Models satisfy most constraints but rarely all, so the gap between the two widens with N (§3.1). **(ii) Scalar rewards cannot repair it.** Only an exact code verifier converts partial satisfaction into full satisfaction; replacing it with an LLM judge on the *same* constraints nearly eliminates the gain, and the policy instead inflates its score through superficial

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

cues (§3.2). **(iii) A holistic teacher propagates it.** The full-instruction teacher decays just as steeply, so the distilled student reproduces the same slope: distillation shifts the curve without flattening it (§3.3). These results point to a single conclusion: escaping the bottleneck requires *decoupling* constraints rather than processing them jointly.

Motivated by these results, we propose **Constraint-Decoupled On-Policy Distillation (CD-OPD)**, which replaces holistic teacher supervision with a constraint-decomposed, token-level target. Its core idea is to *correct the distributional bias in teacher model's responses for multi-constraint instruction following*. Concretely, CD-OPD proceeds in four stages: (1) decompose the multi-constraint instruction into single-constraint prompts; (2) measure each constraint's effect at every token as a log-ratio against an unconstrained base, isolating that constraint's marginal contribution; (3) combine the per-constraint signals through a differentiable min-pooling operator that acts as a soft veto—a token violating *any* constraint is suppressed—while a full-instruction anchor preserves fluency; (4) distill the resulting target into the student along its own trajectories. This turns OPD from holistic imitation into explicit conjunctive constraint satisfaction.

On three teacher–student pairs (32B→4B, 8B→1.7B, 4B→0.6B) and four benchmarks (*IFEval* (Zhou et al., 2023), *IFBench* (Pyatkin et al., 2025), *FollowBench* (Jiang et al., 2024) and *CFBench* (Zhang et al., 2025)), CD-OPD raises average ASR over the base student by 4.48%/2.54%/3.57% and beats Standard OPD at every scale and benchmark (§5.2). It also flattens the decay instead of merely lifting the curve, achieving lower distillation losses: On FollowBench it loses 24.5% from $N{=}1$ to $N{=}5$, against 36.6% for the base student and 31.3% for Standard OPD, and retains 94.9% of the teacher's performance at $N{=}5$. Against scalar RLVR, the two paradigms are complementary: CD-OPD is competitive on the soft constraints [1] that resist deterministic verification, while a code verifier stays stronger on hard constraints. Crucially, CD-OPD matches RLVR while training far more cheaply: dense token-level supervision replaces sparse scalar rewards, and it reaches 58.2% ASR in 5.2 h, whereas RLVR peaks at 56.2% after 8.2 h. This makes CD-OPD a practical substitute for RLVR when instructions carry multiple, semantically nuanced constraints.

2 PRELIMINARIES

2.1 MULTI-CONSTRAINT INSTRUCTION FOLLOWING

We consider *multi-constraint instruction following*, where an instruction specifies not only a base task but also multiple additional constraints. The desired response should simultaneously accomplish the base task and satisfy all attached constraints. Formally, let an instruction be denoted as $x = (q_0, \mathcal{C}), \quad \mathcal{C} = \{c_1, c_2, \ldots, c_N\}$, where q_0 is the base task query and $\mathcal{C}$ is a set of constraints. Constraints may be *hard* constraints that can be verified by deterministic rules, or *soft* constraints that require semantic judgment.

Given a response y, let $\psi_k(y; c_k) \in [0, 1]$ denote the satisfaction level of constraint c_k, where $\psi_k = 1$ indicates full satisfaction. Rather than matching a single reference response y^*, the ideal target distribution for multi-constraint instruction following can be written as $\pi^*(y \mid x) \propto \pi_{\text{base}}(y \mid q_0) \prod_{k=1}^{n} \psi_k(y; c_k)$, where $\pi_{\text{base}}(y \mid q_0)$ captures basic task competence and fluency, the desired response must satisfy the *conjunction* of all constraints. An LLM policy π_θ generates a response autoregressively: $\pi_\theta(y \mid x) = \prod_{t=1}^{|y|} \pi_\theta(y_t \mid x, y_{<t})$. The goal is to align π_θ with the constrained target behavior $\pi^*(y \mid x)$.

2.2 RLVR AS A SCALAR-REWARD PARADIGM

Reinforcement learning with verifiable rewards (RLVR) optimizes a policy by sampling responses and updating it with reward models (code-based, or LLM-based). A representative algorithm is GRPO (Shao et al., 2024), which samples a group of responses per prompt, normalizes their rewards within the group to estimate relative advantages, and updates the policy toward higher-advantage responses without a separate value model; we provide its full objective in Appendix C. This scalar compression is a key limitation of RLVR for multi-constraint tasks. For multi-constraint instructions,

[1] Hard constraints are deterministic, verifiable rules (e.g., format, length); soft constraints are semantic and subjective requirements (e.g., tone, coherence).

all constraints in x are compressed into a single scalar reward: $r_i = R(x, y_i)$. this scalar must implicitly summarize whether *all* constraints are satisfied.

2.3 On-Policy Distillation

On-policy distillation (OPD) provides an alternative to scalar-reward RL by using dense token-level supervision from a teacher model. Unlike standard distillation, which trains the student on a fixed teacher-generated dataset, OPD lets the student generate its own trajectories and queries the teacher for token-level guidance on those on-policy prefixes.

Formally, given an instruction x, the student model π_θ first samples a response $y \sim \pi_\theta(\cdot \mid x)$. At each decoding step t, a teacher model π_T provides a target distribution over the vocabulary conditioned on the same prefix: $\pi_T(\cdot \mid x, y_{<t})$. A general OPD objective minimizes a token-level divergence between the teacher and student distributions:

$$\mathcal{J}_{\text{OPD}}(\theta) = \mathbb{E}_{x,\, y\sim\pi_\theta(\cdot|x)} \left[\sum_{t=1}^{|y|} \mathbb{D}\left(\pi_T(\cdot \mid x, y_{<t}) \,\|\, \pi_\theta(\cdot \mid x, y_{<t})\right) \right], \tag{1}$$

where $\mathbb{D}$ can be instantiated as KL divergence . The key limitation of OPD is that its supervision is bounded by the teacher.

3 Pilot Study

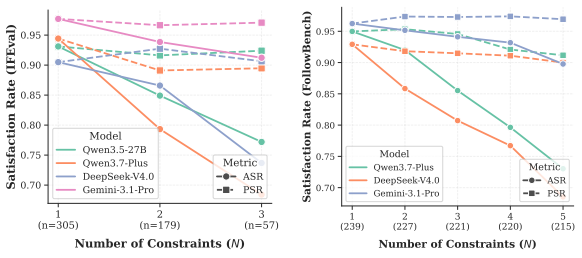
We conduct a three-part pilot study to motivate our method: (1) reveal current LLMs' intrinsic limitations on multi-constraint instructions, (2) show how mainstream RLVR paradigms amplify these issues via reward bias, and (3) demonstrate that Standard OPD inherits them because full-instruction teachers also struggle under multi-constraint IF.

3.1 Intrinsic Limitation: Universal Decay in Advanced LLMs

Setup. We evaluate four advanced LLMs: DeepSeek-V4.0 (DeepSeek-AI, 2026), Qwen3.7-Plus (Qwen Team, 2026b), Qwen3.5-27B (Qwen Team, 2026a), and Gemini-3.1-Pro (Google DeepMind, 2026). We evaluate on two multi-constraint benchmarks: IFEval (Zhou et al., 2023) (focusing hard constraint ♣) and FollowBench (Jiang et al., 2024) (focusing soft constraint ♠). For each sample x with N constraints, let $\{c_1, c_2, \ldots, c_N\}$ denote the constraint set, we define two metrics:

$$\text{ASR} = \frac{1}{|S|} \sum_{x\in S} \prod_{i=1}^{N} \mathbb{I}(c_i(x)), \quad \text{PSR} = \frac{1}{|S|} \sum_{x\in S} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(c_i(x)), \tag{2}$$

where S is the evaluation set. ASR measures the proportion of samples satisfying *all* constraints, while PSR measures the *average fraction* of satisfied constraints per sample. Both metrics are reported stratified by constraint count.



(a) Test on IFEval (b) Test on FollowBench

Figure 1: ASR and PSR vs. constraint count N. All models show a steep ASR decay, while PSR remains stable, highlighting partial satisfaction as the main failure mode.

Results. Figure 1 reveals a universal *compositional decay* across all models and benchmarks. When $N = 1$, both ASR and PSR are high (90–98%). As N increases, ASR drops sharply. For instance, on IFEval ($N = 1 \rightarrow 3$), Qwen3.7-Plus drops from 94.4% to 68.4%. In contrast, PSR remains remarkably stable (above 89%) and drops by at most 5%. on FollowBench (N=1→5), Qwen3.7-Plus falls from 95.0% to 73.0% (−22.0%) and DeepSeek-V4.0 from 92.9% to 68.5% (−24.4%). Gemini-3.1-Pro, despite being the strongest model, still loses 6.5% on IFEval and 6.5% on FollowBench over the same ranges. In stark contrast, PSR remains remarkably stable, declining by at most 5% and staying above 89% even at the highest N for all models. Consequently, the gap between PSR and ASR widens significantly as N grows.

Analysis. The gap between stable PSR and decaying ASR reveals the core bottleneck. LLMs rarely fail completely on single constraints. Instead, they satisfy most constraints but violate one or two. These partial failures multiply as N increases. This issue persists across all model sizes, proving that scaling alone cannot solve compositional generalization. Furthermore, because failures only involve a few constraints, holistic training signals cannot identify *which* specific constraint was violated. This leads to the credit-assignment failure discussed next.

3.2 Extrinsic Amplification: The Failure of Scalar RLVR

Setup. To test if standard RL alignment can fix this bottleneck, we fine-tune a base model (Qwen3-4B (Yang et al., 2025)) using RLVR (GRPO (Shao et al., 2024)). We use three reward setups: (1) a rule-based *Code Verifier* for hard constraints, (2) an *LLM-based Reward Model* for hard constraints, and (3) an *LLM-based Reward Model* for soft constraints. Training utilizes the *VerInstruct* dataset (Peng et al., 2025) (22K instances), which includes both hard constraints (validated via deterministic code) and soft constraints (judged by Qwen3-32B (Yang et al., 2025)).

Results and Analysis. Table 1 shows that RLVR's success depends entirely on reward accuracy. Under hard constraints, the precise Code Verifier improves ASR by +5.9% on IFEval and +8.7% on FollowBench, and narrows the PSR–ASR gap from 6.8% to 6.4% and from 17.1% to 14.7%, respectively. An LLM judge, by contrast, yields markedly smaller gains (+2.2% and +4.7%), while the PSR–ASR gap remains large (e.g., 15.7% on FollowBench). This contrast proves the bottleneck is the *scalar-reward paradigm*, not RL itself. As §3.1 shows, failures are partial: only a few constraints are violated per sample. A scalar reward therefore erases the credit-assignment signal needed to identify which constraint to correct. Under noisy LLM judgments, the policy instead exploits superficial cues that inflate the score (i.e., reward hacking) (Zeng et al., 2026c; Thaman, 2026). Thus, scalar RLVR fails to resolve the intrinsic compositional decay.

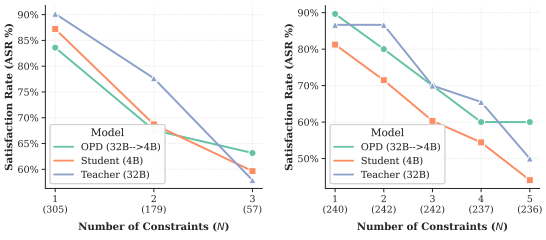
Table 1: Impact of RLVR with different rewards. LLM-based rewards yield negligible gains on both hard and soft constraints, the precise code verifier partly converts partial satisfaction into full satisfaction.

Model	Train	♣IFEval		♠FollowBench	
		ASR	PSR	ASR	PSR
Qwen3-4B	-	78.0	84.8	62.4	79.5
+ RLVR (Code)	Hard	83.9$_{\uparrow+5.9}$	90.3$_{\uparrow+5.5}$	71.1$_{\uparrow+8.7}$	85.8$_{\uparrow+6.3}$
+ RLVR (LLM)	Hard	80.2$_{\uparrow+2.2}$	87.2$_{\uparrow+2.5}$	67.1$_{\uparrow+4.7}$	82.8$_{\uparrow+3.4}$
+ RLVR (LLM)	Soft	81.0$_{\uparrow+3.0}$	88.2$_{\uparrow+3.4}$	68.2$_{\uparrow+5.8}$	83.9$_{\uparrow+4.5}$

Additional reward scores curves in training (Fig. 6) and test performance as constraint count increases (Fig. 7) are reported in Appendix D. These results further confirm excessively high reward scores for LLM-reward variants, a hallmark of reward hacking rather than genuine capability acquisition, consistent with findings in related work (Zeng et al., 2026b; Thaman, 2026).

3.3 Inherited Limitation: Standard OPD Cannot Escape the Decay

Setup. Since scalar RLVR fails to resolve the intrinsic decay (§3.2), we next examine Standard OPD as an alternative. We test two teacher–student pairs: Qwen3-32B → Qwen3-4B and Qwen3-8B → Qwen3-1.7B (Yang et al., 2025). Crucially, the teacher processes the *entire* instruction at once, mirroring standard inference. We then distill the teacher's behavior into the student and evaluate the ASR decay.



(a) Test on IFEval (b) Test on FollowBench

Figure 2: ASR vs. constraint count N. All three curves decay with comparable steepness: distillation shifts the level marginally but inherits the Teacher's compositional bottleneck.

Results and Analysis. Figure 2 shows the ASR decay for the Base student, Teacher, and OPD Student. The Teacher performs better at low N but suffers from the *same steep decay* at high N. Its ASR drops by 33–37 points.

The distilled student copies this exact decay curve, dropping by 20–30 points. OPD only provides minor improvements and cannot change the decay slope. At high N, all three models converge to the same low satisfaction rate (44–63%). Standard OPD merely *propagates* the teacher's limitations. Because the teacher itself fails partially at high N, the student learns these partial failures. Therefore, solving the multi-constraint bottleneck requires a new paradigm that *decouples* constraint processing, rather than handling all constraints at once. (See Section B for a theoretical comparison).

Key Takeaways

- **Intrinsic Compositional Decay:** Advanced LLMs maintain high single-constraint accuracy (PSR) but suffer a sharp drop in full-constraint accuracy (ASR) as constraints increases.
- **Failure of Scalar RLVR:** Standard RLVR with scalar rewards fails to fix this decay. Without precise verifiers, it suffers from credit-assignment issues and LLM reward hacking, leaving the partial-satisfaction gap intact.
- **Limitation of Standard OPD:** Standard OPD inherits the teacher's decay. Since the teacher struggles with multiple constraints, the student merely copies these partial failures.

4 Methodology: Constraint-Decoupled On-Policy Distillation

As demonstrated in Section 3, the fundamental barrier to standard OPD is the holistic processing of constraints within the teacher model, which inevitably propagates partial satisfaction. We address this by introducing **Constraint-Decoupled On-Policy Distillation (CD-OPD)**, a paradigm that replaces monolithic teacher supervision with a constraint-decoupled signal.

Formally, CD-OPD factorizes the multi-constraint objective into a theoretically grounded target distribution (Figure 3). This is achieved through four stages: constraint decomposition, contrastive signal extraction, strict conjunction via min-pooling, and alignment distillation. By evaluating each constraint in isolation and synthesizing their token-level probabilities via contrastive min-pooling, it guarantees that the distilled policy satisfies the strict conjunction of all constraints, thereby escaping the intrinsic decay observed in holistic approaches.

4.1 Constraint Decoupling and Conjunction

Constraint Decomposition. Let a complex instruction x consist of a base task query q_0 and a set of n constraints $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$. To enable isolated response, we decompose x into $N+1$ semantically self-contained prompts prior to training: $q_k = \text{concat}(q_0, c_k), \quad k \in \{1, \ldots, N\}$.

During on-policy training, the student model π_θ first generates a candidate trajectory $y = (y_1, \ldots, y_{|y|})$ based on full instruction x. Then, we introduce a shared teacher model π_T. The teacher π_T computes a reference probability distribution over y and q_0:

$$z_{0,t}(v) \triangleq \text{logit}_{\pi_T}(v \mid q_0, y_{<t}), \quad \forall v \in \mathcal{V}, \tag{3}$$

where $z_{0,t} \in \mathbb{R}^{|\mathcal{V}|}$ denotes the raw logit vector, as a *contrastive reference* at decoding step t. This reference captures the basic task-solving and linguistic prior without explicitly injected constraints. Besides, the teacher π_T also computes the full-instruction logits as an anchor: $z_{x,t}(v) \triangleq \text{logit}_{\pi_T}(v \mid x, y_{<t})$. This anchor preserve the joint semantics and naturalness of the complete instruction.

Contrastive Signal Extraction. To independently resolve each constraint, we instantiate $N+1$ constraint-specific responses using the same teacher backbone π_T but with distinct prompts q_k. For each token v at step t, the k-th constraint yields a conditioned logit

$$z_{k,t}(v) = \text{logit}_{\pi_T}(v \mid q_k, y_{<t}). \tag{4}$$

We isolate the directional influence of each constraint by computing the contrastive signal Δ_k:

$$\Delta_{k,t}(v) = z_{k,t}(v) - z_{0,t}(v). \tag{5}$$

This difference is equivalent to the log-ratio $\log \frac{\pi_T(v|q_k, y_{<t})}{\pi_T(v|q_0, y_{<t})}$. It thus measures the *evidence* that token v provides for satisfying constraint c_k beyond the unconstrained base policy: positive values indicate active promotion of c_k, while negative values signal violation.

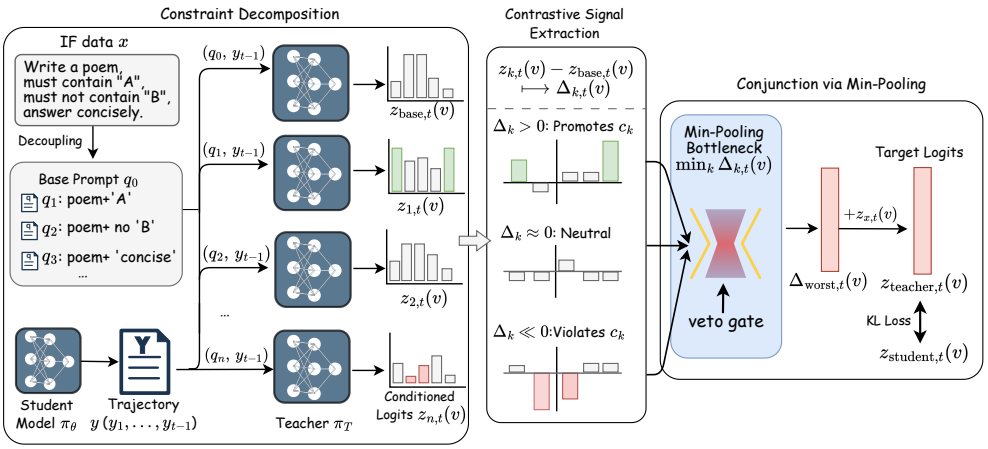


Figure 3: Overview of CD-OPD. It (1) decomposes multi-constraint instructions into isolated single-constraint responses, (2) extracting per-constraint contrastive signals at the token level. (3) A min-pooling bottleneck enforces strict logical conjunction over these signals while a full-instruction anchor preserves fluency, and (4) the student aligns to this synthesized target via on-policy distillation.

To ensure fair aggregation across constraints with varying intervention strengths, we normalize each $\Delta_{k,t}$ to $\widehat{\Delta}_{k,t}$, a zero-mean unit-variance over a restricted support set and clip extreme values. Full implementation details are provided in Appendix E.

Conjunction via Min-Pooling. Effective multi-constraint alignment requires enforcing a logical AND ($\wedge$) relationship across all constraints. Traditional averaging permits compensation, where strong adherence to one constraint can mask severe violations of another, leading to partial-satisfaction. To enforce an absolute veto mechanism, we introduce contrastive min-pooling.

$$\widetilde{\Delta}_t(v) = -\frac{1}{\beta} \log \sum_{k=1}^{N} \exp\left(-\beta\, \widehat{\Delta}_{k,t}(v)\right), \qquad z_{\text{target},t}(v) = z_{x,t}(v) + \alpha\, \widetilde{\Delta}_t(v), \tag{6}$$

where $\beta > 0$ controls the sharpness of the soft minimum and $\alpha > 0$ modulates the strength of constraint enforcement relative to the anchor. We adopt a *soft* minimum rather than a hard $\min$ operator because it provides smooth, non-sparse gradients during training. As $\beta \to \infty$, $\widetilde{\Delta}_t(v)$ approaches $\min_{k\in\{1,\ldots,n\}} \widehat{\Delta}_{k,t}(v)$.

It guaranties that if a token violates *any* single constraint, the aggregated signal immediately inherits the corresponding penalty. Conversely, tokens, that are neutral across all constraints, safely bypass the bottleneck without being suppressed. Crucially, this design decouples semantic preservation from constraint enforcement: the full-instruction anchor $z_{x,t}$ maintains the joint semantics and fluency, while the pooled contrastive signal $\widetilde{\Delta}_t$ applies a targeted corrective adjustment.

4.2 Alignment via Guided Distillation

The aggregated logits $z_{\text{target},t}$ from Eq. equation 6 are converted into a target distribution π_{T^*} via softmax over the anchor support $\mathcal{S}_t$. The student policy π_θ is then updated by minimizing the token-level forward KL divergence over on-policy trajectories:

$$\mathcal{J}_{\text{CD-OPD}}(\theta) = \mathbb{E}_{x,y\sim\pi_\theta}\left[\frac{1}{|y|}\sum_{t=1}^{|y|} \mathbb{D}_{\text{KL}}\left(q_t(v) \,\|\, p_t(v)\right)\right], \tag{7}$$

where $q_t(v) \triangleq \pi_{T^*}(v \mid x, y_{<t})$ and $p_t(v) \triangleq \pi_\theta(v \mid x, y_{<t})$, both evaluated on $\mathcal{S}_t$. Here q_t is renormalized on $\mathcal{S}_t$, while $p_t(v)$ takes the student's full-vocabulary probability at each $v \in \mathcal{S}_t$ without renormalization to preserve valid likelihood ratios.

Table 2: **Main Results on multi-constraint instruction following.** We report All-Satisfaction Rate (ASR, %) on hard-constraint benchmarks (IFEval, IFBench) and soft-constraint benchmarks (FollowBench, CFBench). CD-OPD achieves strong performance across all scales, closely approaching or surpassing the teacher on structured constraint satisfaction.

Model	Hard Constraints		Soft Constraints		Avg.
	IFEval	IFBench	FollowBench	CFBench	
Teacher: Qwen3-32B → Student: Qwen3-4B					
Teacher (32B)	*82.62*	*31.97*	*71.81*	*35.20*	*55.40*
Base (4B)	78.01	27.25	62.41	24.05	47.93
Standard OPD	78.38$_{\uparrow 0.37}$	25.17$_{\downarrow 2.08}$	68.67$_{\uparrow 6.26}$	25.70$_{\uparrow 1.65}$	49.48$_{\uparrow 1.55}$
RLVR (Hard+LLM)	80.22$_{\uparrow 2.21}$	30.61$_{\uparrow 3.36}$	67.11$_{\uparrow 4.70}$	27.22$_{\uparrow 3.17}$	51.29$_{\uparrow 3.36}$
RLVR (Soft+LLM)	80.96$_{\uparrow 2.95}$	28.25$_{\uparrow 1.00}$	68.24$_{\uparrow 5.83}$	26.12$_{\uparrow 2.07}$	50.89$_{\uparrow 2.96}$
RLVR (Hard+Code)	**83.92**$_{\uparrow 5.91}$	**32.99**$_{\uparrow 5.74}$	**71.14**$_{\uparrow 8.73}$	**28.81**$_{\uparrow 4.76}$	**54.22**$_{\uparrow 6.29}$
CD-OPD (Ours)	81.95$_{\uparrow 3.94}$	29.93$_{\uparrow 2.68}$	70.84$_{\uparrow 8.43}$	26.90$_{\uparrow 2.85}$	52.41$_{\uparrow 4.48}$
Teacher: Qwen3-8B → Student: Qwen3-1.7B					
Teacher (8B)	*82.81*	*29.93*	*68.24*	*27.07*	*52.01*
Base (1.7B)	68.76	22.45	49.82	17.61	39.66
Standard OPD	68.21$_{\downarrow 0.55}$	22.79$_{\uparrow 0.34}$	50.26$_{\uparrow 0.44}$	18.88$_{\uparrow 1.27}$	40.04$_{\uparrow 0.38}$
RLVR (Hard+LLM)	69.13$_{\uparrow 0.37}$	23.67$_{\uparrow 1.22}$	50.69$_{\uparrow 0.87}$	19.32$_{\uparrow 1.71}$	40.70$_{\uparrow 1.04}$
RLVR (Soft+LLM)	67.59$_{\downarrow 1.17}$	21.72$_{\downarrow 0.73}$	49.67$_{\downarrow 0.15}$	19.03$_{\uparrow 1.42}$	39.50$_{\downarrow 0.16}$
RLVR (Hard+Code)	**70.24**$_{\uparrow 1.48}$	**26.15**$_{\uparrow 3.70}$	**53.66**$_{\uparrow 3.84}$	**21.71**$_{\uparrow 4.10}$	**42.94**$_{\uparrow 3.28}$
CD-OPD (Ours)	69.87$_{\uparrow 1.11}$	25.17$_{\uparrow 2.72}$	52.97$_{\uparrow 3.15}$	20.77$_{\uparrow 3.16}$	42.20$_{\uparrow 2.54}$
Teacher: Qwen3-4B → Student: Qwen3-0.6B					
Teacher (4B)	*78.01*	*27.25*	*62.41*	*24.05*	*47.93*
Base (0.6B)	54.16	20.41	25.00	8.52	27.02
Standard OPD	53.70$_{\downarrow 0.46}$	20.75$_{\uparrow 0.34}$	28.38$_{\uparrow 3.38}$	9.71$_{\uparrow 1.19}$	28.14$_{\uparrow 1.11}$
RLVR (Hard+LLM)	52.87$_{\downarrow 1.29}$	22.41$_{\uparrow 2.00}$	29.97$_{\uparrow 4.97}$	10.88$_{\uparrow 2.36}$	29.03$_{\uparrow 2.01}$
RLVR (Soft+LLM)	52.15$_{\downarrow 2.01}$	21.67$_{\uparrow 1.26}$	28.92$_{\uparrow 3.92}$	9.44$_{\uparrow 0.92}$	28.05$_{\uparrow 1.02}$
RLVR (Hard+Code)	**57.49**$_{\uparrow 3.33}$	**25.51**$_{\uparrow 5.10}$	**34.05**$_{\uparrow 9.05}$	**11.71**$_{\uparrow 3.19}$	**32.19**$_{\uparrow 5.17}$
CD-OPD (Ours)	56.19$_{\uparrow 2.03}$	23.09$_{\uparrow 2.68}$	32.41$_{\uparrow 7.41}$	10.67$_{\uparrow 2.15}$	30.59$_{\uparrow 3.57}$

Through this four-stage pipeline, CD-OPD directly addresses the intrinsic decay and partial satisfaction bottlenecks identified in Section 3: by decoupling constraint response, enforcing strict conjunction, and aligning via structured supervision, the student learns to satisfy all constraints jointly at the token level.

5 Experiments

5.1 Setup

Models and Baselines. We evaluate CD-OPD on three teacher-student pairs spanning diverse scales: Qwen3-32B → 4B, Qwen3-8B → 1.7B, and Qwen3-4B → 0.6B. For each pair, we compare against five baselines trained on the same VerInstruct dataset (Peng et al., 2025): (1) *Student* and *Teacher*, serving as the unaligned baseline and performance upper bound respectively; (2) *Standard OPD*, which distills full-instruction teacher logits and serves as the primary ablation for constraint decoupling; (3–5) three *GRPO-based RLVR* variants that differ in both training constraint type and verifier: *Hard+LLM* (hard constraints with LLM verifier), *Soft+LLM* (soft constraints with LLM verifier), and *Hard+Code* (hard constraints with code-based verifier). The training protocol is identical to the pilot study (§3), and all methods share the same hardware resources and hyperparameters. Additional implementation details are provided in Appendix §F.1.

Evaluation Benchmarks. We report All-Satisfaction Rate (ASR, %) on four multi-constraint instruction-following benchmarks covering both hard and soft constraints: *IFEval* (Zhou et al., 2023) and *IFBench* (Pyatkin et al., 2025) for strict, verifiable hard constraints; *FollowBench* (Jiang et al., 2024) and *CFBench* (Zhang et al., 2025) for nuanced soft constraints requiring semantic judgment. These benchmark's average constraint counts per instance of 1.54 (IFBench), 1.14 (IFEval), 3.00

(FollowBench), and 4.24 (CFBench) respectively. Detailed benchmark descriptions and statistics are provided in Appendix §F.2.

5.2 Main Results

Table 2 summarizes performance across all scales. CD-OPD consistently improves over the base student by +4.48%/+2.54%/+3.57% in average ASR for the 4B/1.7B/0.6B models respectively. We highlight three key findings.

Decoupling Outperforms Holistic Distillation. CD-OPD surpasses Standard OPD on every benchmark and scale, with average gains of +2.93%/+2.16%/+2.45% across the four benchmarks for 4B/1.7B/0.6B respectively. The improvement is most pronounced on FollowBench (+8.43%/+3.15%/+7.41%), confirming that decomposing multi-constraint instructions and enforcing strict conjunction via min-pooling effectively alleviates the partial-satisfaction ceiling inherent in holistic teacher supervision.

Structured Supervision Complements Scalar RL. Compared to the strongest RL baseline (RLVR Hard+Code), CD-OPD achieves competitive or superior performance on soft constraints: it outperforms on FollowBench at the 32B→4B scale (70.84 vs. 71.14, comparable) and substantially at smaller scales where RLVR struggles. On hard constraints, RLVR (Hard+Code) generally leads, suggesting that code-verifiable rules benefit more from scalar reward optimization, while CD-OPD's token-level structured supervision provides complementary strengths on semantically nuanced soft constraints that resist deterministic verification.

Consistent Gains Across Scales. CD-OPD delivers stable absolute improvements regardless of student capacity, with no degradation at smaller scales. Notably, at 32B→4B, CD-OPD closes the gap to the teacher to within 3.0 pp on average ASR (52.41 vs. 55.40), demonstrating that structured distillation preserves compositional generalization capacity even as task complexity increases. At smaller scales where the teacher itself is weaker, CD-OPD still extracts meaningful signal beyond Standard OPD, confirming its robustness to teacher quality.

ASR Decay Curves. Figure 4 plots FollowBench ASR against constraint count N for Qwen3-32B → 4B. While all methods degrade with increasing N, CD-OPD exhibits a substantially flatter slope: it loses only 24.5 pp from $N = 1$ (85.7%) to $N = 5$ (61.2%), compared to 36.6 pp for the Student base (75.7% → 39.1%) and 31.3 pp for Standard OPD (75.7% → 45.4%). At $N = 5$, CD-OPD outperforms the strongest RLVR baseline (Hard Code, 56.2%) by +5.0 pp and retains 94.9% of the Teacher's performance (64.5%), versus 70.4% for Standard OPD. This confirms that constraint decoupling reduces the compositional decay rate itself rather than merely lifting the curve, preserving generalization capacity at high complexity.

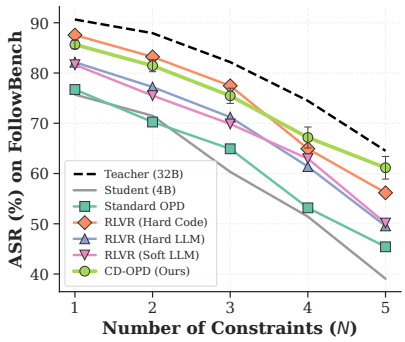


Figure 4: ASR decay curves (Qwen3-32B → 4B, FollowBench). CD-OPD exhibits a flatter slope than all baselines, mitigating the compositional decay.

5.3 Ablation Study

We evaluate three core design choices of CD-OPD on the Qwen3-32B → 4B pair (Table 3):

(1) Mean vs. Min Aggregation. Replacing soft-min with arithmetic averaging degrades ASR by ↓5.2/↓5.1% on IFEval/FollowBench. Averaging permits high adherence to one constraint to offset violations of another, thereby reproducing the scalar-reward failure mode. The min-pooling veto is thus essential for enforcing strict logical conjunction across both hard and soft constraints.

(2) Contrastive Reference (q_0 vs. x). Substituting the base-prompt reference $z_{0,t}$ with full-instruction logits $z_{x,t}$ causes a substantial drop of ↓15.2/↓17.5%. Since z_x already encodes all constraints jointly, the difference $\Delta_k = z_k - z_x$ conflates the absence of other constraints with the presence of c_k,

introducing cross-constraint interference. In contrast, the unconstrained baseline q_0 cleanly isolates each constraint's marginal effect.

(3) Without Contrastive Signal. Removing the contrastive correction $\Delta_k = z_k - z_0$ and directly distilling full-instruction teacher logits recovers Standard OPD, yielding the largest degradation among single-component ablations ($\downarrow$3.6/$\downarrow$2.2%). This confirms that our corrected target distribution effectively alleviates the partial-satisfaction ceiling inherent in holistic supervision.

Together, these results demonstrate that non-compensatory aggregation, a constraint-free reference, and contrastive decomposition are each indispensable to CD-OPD's gains.

Table 3: Ablation on CD-OPD components (Qwen3-32B $\rightarrow$ 4B). Reported as ASR (%) on IFEval (hard) and FollowBench (soft).

Variant	IFEval	FollowBench
CD-OPD (Full)	81.95	70.84
w/ Mean Agg.	76.71$_{\downarrow 5.2}$	65.77$_{\downarrow 5.1}$
w/ Full-Inst Ref.	66.80$_{\downarrow 15.2}$	53.34$_{\downarrow 17.5}$
w/o Contrastive	78.38$_{\downarrow 3.6}$	68.67$_{\downarrow 2.2}$

5.4 Computational Efficiency Analysis

A natural concern for CD-OPD is the computational overhead introduced by constraint decoupling: evaluating $N + 1$ prompts per step requires multiple teacher forward passes, potentially limiting scalability. To assess this trade-off, we measure the *training wall-clock time* required to reach target ASR levels under identical hardware conditions.

Figure 5 presents the efficiency-performance Pareto frontier across all alignment methods. CD-OPD clearly dominates the frontier: at only 5.2 h of training, it already achieves 58.2% ASR, a performance level that RLVR (Hard Code) does not reach until 8.2h (56.2%) and that RLVR (Soft LLM) never attains within the observed 21.6h budget. Conversely, at a comparable ASR target of $\sim$50%, CD-OPD requires merely 5.2h, whereas RLVR (Hard Code) needs 5.4h and RLVR (Soft LLM) needs 10.9h, representing a 1.0$\times$ and 2.1$\times$ speedup respectively. IF-OPD, while fastest per step, saturates at $\sim$45% ASR after 3.1h and shows no meaningful improvement with additional training, confirming that its apparent efficiency is illusory for high-complexity multi-constraint tasks.

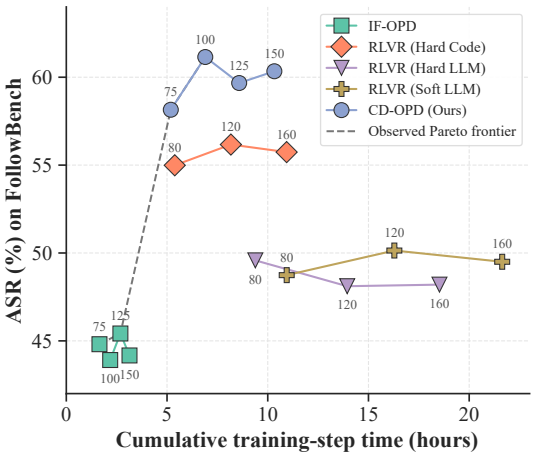


Figure 5: Efficiency-Performance Trade-off. CD-OPD dominates the observed Pareto frontier.

This advantage stems from two factors: (1) structured token-level supervision provides denser gradient signals than sparse scalar rewards, accelerating convergence; and (2) the $N + 1$ teacher inference calls are fully parallelized across the accelerator cluster, so the per-step latency increase is sub-linear relative to N. These results confirm that CD-OPD's decoupling overhead is effectively amortized by superior sample efficiency and parallelized inference, rendering it the most time-effective path to robust multi-constraint alignment.

6 Conclusion

We studied multi-constraint instruction following. Our pilot study identifies partial satisfaction as the dominant failure mode, which neither scalar RLVR nor standard OPD resolves due to lost constraint-level structure. We propose CD-OPD, which decouples instructions into single-constraint prompts, scores each constraint's token-level effect against an unconstrained base, and combines them via min-pooling (a soft veto) with a fluency anchor. Across three scales and four benchmarks, CD-OPD consistently outperforms baselines, flattens ASR decay, and converges faster than RLVR.

AI USE STATEMENT

In this work, we used ChatGPT to assist with English-language editing, code debugging, and data processing. The authors reviewed all AI-assisted text paragraph by paragraph for technical accuracy and consistency with the reported methods and results. We executed the modified code and checked its outputs. We also checked sample counts and manually inspected sampled records after data processing. The authors take full responsibility for the final manuscript and associated artifacts, including all content produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide implementation details of CD-OPD in Appendix E. Training hyperparameters, hardware settings, and generation configurations are documented in Appendix F.1 and Tables 4 and 5. Benchmark descriptions are provided in Appendix F.2. Hyperparameter sensitivity analyses are presented in Appendix G.1 and Figure 8, while component ablations are reported in Section 5.3. Our implementation has been submitted as supplementary material via OpenReview. We will make the code publicly available.

LIMITATION

CD-OPD has three limitations. First, it does not match code-verified RLVR on hard constraints, where deterministic rules supply an exact and cheap reward; on such constraints the two paradigms are complementary rather than substitutable. Second, decoupling requires $N+1$ teacher forward passes per step, so compute and memory grow with the number of constraints, although our efficiency analysis indicates that this overhead is amortized by faster convergence and parallel inference. Third, decomposition assumes that constraints can be restated as self-contained prompts; constraints that are implicit, mutually dependent, or inexpressible in isolation would be difficult to isolate, and our evaluation stratifies constraint counts up to five.

REFERENCES

Anthropic. Claude code. https://github.com/anthropics/claude-code, 2025. Accessed: 2026-09-25.

Aaron Blakeman, Aaron Thomas, Aastha Jhunjhunwala, Abhibha Gupta, Abhinav Khattar, Adam Rajfer, Adi Renduchintala, Adil Asif, Aditya Vavre, Adriana Flores Miranda, et al. Nemotron 3 ultra: Open, efficient mixture-of-experts hybrid mamba-transformer model for agentic reasoning. *arXiv preprint arXiv:2606.15007*, 2026.

Muzhi Dai, Chenxu Yang, and Qingyi Si. S-grpo: Early exit via reinforcement learning in reasoning models. *arXiv preprint arXiv:2505.07686*, 2025.

DeepSeek-AI. Deepseek-v4.0. https://www.deepseek.com/, 2026. Accessed: 2026-09-21.

Guanting Dong, Keming Lu, Chengpeng Li, Tingyu Xia, Bowen Yu, Chang Zhou, and Jingren Zhou. Self-play with execution feedback: Improving instruction-following capabilities of large language models. *arXiv preprint arXiv:2406.13542*, 2024.

Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. Concise reasoning via reinforcement learning. *arXiv preprint arXiv:2504.05185*, 2025.

Zixuan Fu, Bingxiang He, Yuxin Zuo, Haohuan Huang, Jinqian Zhang, Ruhang Xiao, Cheng Qian, Qinyu Luo, Huan-ang Gao, Yudong Wang, et al. Rethinking on-policy distillation of large language models ii: One training example. *arXiv preprint arXiv:2609.04172*, 2026.

Google DeepMind. Gemini 3.1 pro model card. https://deepmind.google/models/model-cards/gemini-3-1-pro/, February 2026. Accessed: 2026-09-21.

Zhengkang Guo, Wenhao Liu, Mingchen Xie, Jingwen Xu, Zisu Huang, Muzhao Tian, Jianhan Xu, Yuanzhe Shen, Qi Qian, Muling Wu, Xiaohua Wang, Changze Lv, He-Da Wang, Hu Yao, Xiaoqing Zheng, and Xuanjing Huang. Recast: Expanding the boundaries of llms' complex instruction following with multi-constraint data, 2025. URL https://arxiv.org/abs/2505.19030.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. Followbench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4667–4688, 2024.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

OpenAI. Gpt-6 astra technical report. Technical report, OpenAI, September 2026a. URL https://deploymentsafety.openai.com/gpt-6-astra/gpt-6-astra.pdf.

OpenAI. Codex CLI: Coding agent for local development. https://github.com/openai/codex, 2026b. Accessed: 2026-09-25.

Hao Peng, Yunjia Qi, Xiaozhi Wang, Bin Xu, Lei Hou, and Juanzi Li. Verif: Verification engineering for reinforcement learning in instruction following. *arXiv preprint arXiv:2506.09942*, 2025.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents. https://qwen.ai/blog?id=qwen3.5, February 2026a. Accessed: 2026-09-21.

Qwen Team. Qwen3.7-plus: Multimodal agent intelligence. https://qwen.ai/blog?id=qwen3.7-plus, 2026b. Accessed: 2026-09-21.

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.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. *arXiv preprint arXiv:2604.00626*, 2026.

Jiao Sun, Yufei Tian, Wangchunshu Zhou, Nan Xu, Qian Hu, Rahul Gupta, John Wieting, Nanyun Peng, and Xuezhe Ma. Evaluating large language models on controlled generation tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3155–3168, 2023.

Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. Let me speak freely? a study on the impact of format restrictions on performance of large language models. *arXiv preprint arXiv:2408.02442*, 2024.

Kimi Team and Yifan Bai. Kimi k2: Open agentic intelligence, 2025. URL https://arxiv.org/abs/2507.20534.

Kunvar Thaman. Reward hacking benchmark: measuring exploits in llm agents with tool use. *arXiv preprint arXiv:2605.02964*, 2026.

Kuan Wang, Alexander Bukharin, Haoming Jiang, Qingyu Yin, Zhengyang Wang, Tuo Zhao, Jingbo Shang, Chao Zhang, Bing Yin, Xian Li, et al. Rnr: Teaching large language models to follow roles and rules. *arXiv preprint arXiv:2409.13733*, 2024.

Zhaoyang Wang, Jinqi Jiang, Huichi Zhou, Wenhao Zheng, Xuchao Zhang, Chetan Bansal, and Huaxiu Yao. Verifiable format control for large language model generations. *arXiv preprint arXiv:2502.04498*, 2025.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, et al. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*, 2026.

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.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*, 2026.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026a.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

Yirong Zeng, Yufei Liu, Xiao Ding, Yutai Hou, Yuxian Wang, Wu Ning, Haonan Song, Dandan Tu, Qixun Zhang, Yuxiang He, et al. Tinyjudge: Unverifiable constraint alignment via lightweight specialist ensembles. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26198–26212, 2026b.

Yirong Zeng, Yufei Liu, Xiao Ding, Yutai Hou, Yuxian Wang, Haonan Song, Wu Ning, Dandan Tu, Qixun Zhang, Bibo Cai, et al. Precision over diversity: High-precision reward generalizes to robust instruction following. *arXiv preprint arXiv:2601.04954*, 2026c.

Tao Zhang, Chenglin Zhu, Yanjun Shen, Wenjing Luo, Yan Zhang, Hao Liang, Fan Yang, Mingan Lin, Yujing Qiao, Weipeng Chen, et al. Cfbench: A comprehensive constraints-following benchmark for llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32926–32944, 2025.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *CoRR*, 2023.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

A Related Work

Instruction Following and RLVR. While single-constraint instruction following has been largely addressed through SFT, satisfying multiple heterogeneous constraints simultaneously remains a fundamental challenge (Tam et al., 2024; Sun et al., 2023; Zhang et al., 2025). Prior efforts have focused on data synthesis (AutoIF (Dong et al., 2024), RNR (Wang et al., 2024), VFF (Wang et al., 2025)) and benchmark-driven evaluation (Jiang et al., 2024; Zhang et al., 2025; Zhou et al., 2023), yet these approaches do not address the underlying compositional bottleneck at the algorithmic level. RLVR offers an alternative alignment pathway and has proven effective in reasoning domains with explicit ground truth (Yue et al., 2025; Zhu et al., 2025; Dai et al., 2025; Team & Bai, 2025; Zeng et al., 2025; Fatemi et al., 2025; Liu et al., 2025). However, extending RLVR to multi-constraint IF via LLM-as-a-judge (Pyatkin et al., 2025; Guo et al., 2025; Peng et al., 2025; Lambert et al., 2024) introduces a scalar compression bottleneck: collapsing multiple constraint evaluations into a single reward erases which constraint was violated, enabling reward hacking and reinforcement of partial satisfaction (§3.2). Existing remedies improve the judge model (Zeng et al., 2026b; Thaman, 2026) but leave this structural limitation intact. In contrast, our work departs from scalar rewards entirely, replacing sequence-level advantage broadcasting with token-level distributional supervision that preserves constraint-level structure throughout training.

On-Policy Distillation. OPD trains the student on its own rollouts with teacher supervision, aligning the training distribution with the student's inference-time behavior (Song & Zheng, 2026). Recent studies (Li et al., 2026; Fu et al., 2026) provide practical guidelines for applying OPD at scale, Yang et al. (2026) cast OPD as dense KL-constrained RL, showing that the per-token log-ratio acts as an implicit reward signal. OPD has been widely adopted in frontier post-training pipelines (Yang et al., 2025; Zeng et al., 2026a; Xiao et al., 2026; DeepSeek-AI, 2026; Blakeman et al., 2026). However, its application to instruction following, particularly multi-constraint settings, remains unexplored. Crucially, existing OPD formulations condition the teacher on the *full* instruction holistically, causing the teacher's compositional bottleneck to propagate directly to the student (§3.3). Our CD-OPD addresses this by decoupling constraints into independent prompts and aggregating their token-level distributions via contrastive min-pooling, explicitly encoding logical-AND semantics into the distillation objective.

B Why Scalar Rewards Struggle in Multi-Constraint IF?

In this part, we provide a theoretical comparison of RLVR and OPD for multi-constraint instruction following. We compare them from the perspective of credit assignment. In RLVR, the learning signal for a sampled response y_i is approximately scaled by a sequence-level advantage $\hat{A}_i$:

$$\nabla_\theta \mathcal{J}_{\mathrm{RLVR}} \;\approx\; \mathbb{E}\left[\sum_i \hat{A}_i \sum_t \nabla_\theta \log \pi_\theta(y_{i,t} \mid x, y_{i,<t})\right]. \tag{8}$$

Thus, the same scalar advantage is broadcast to all tokens in the response. If a response receives a high reward despite violating some constraints, RLVR will reinforce both compliant and violating tokens, since the scalar reward does not identify which constraint has failed. This problem is amplified when the reward model is biased.

OPD instead replaces this scalar broadcast with the token-level distributional supervision in Eq. equation 1. Such supervision is localized to individual decoding decisions: tokens that tend to violate a constraint can be assigned lower target probability, while constraint-neutral tokens need not be suppressed. Therefore, OPD provides finer credit assignment and is structurally less vulnerable to partial-satisfaction reinforcement than scalar-reward RLVR.

C GRPO Objective

For each instruction x, GRPO samples a group of G responses $\{y_i\}_{i=1}^{G}$ from the old policy $\pi_{\theta_{\mathrm{old}}}$. Each response receives a scalar reward $r_i = R(x, y_i)$. The group-normalized advantage is

$$\hat{A}_i = \frac{r_i - \mu}{\sigma}, \qquad \mu = \frac{1}{G}\sum_{i=1}^{G} r_i, \qquad \sigma = \sqrt{\frac{1}{G}\sum_{i=1}^{G}(r_i - \mu)^2 + \epsilon}.$$

Let

$$\rho_{i,t} = \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\mathrm{old}}}(y_{i,t} \mid x, y_{i,<t})}.$$

A simplified GRPO objective is

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{x,\{y_i\}\sim\pi_{\theta_{\mathrm{old}}}}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\Big(\min\left(\rho_{i,t}\hat{A}_i, \mathrm{clip}(\rho_{i,t}, 1-\varepsilon, 1+\varepsilon)\hat{A}_i\right) - \beta\,\mathbb{D}_{\mathrm{KL}}[\pi_\theta(\cdot \mid x, y_{i,<t}) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid x, y_{i,<t})]\Big)\right].$$

Here ε is the clipping coefficient and β controls the strength of the KL regularization.

D Experiments in Pilot Study

Visualization of Training. To further understand the above observation, we visualize the model training curves (Qwen3-32B as the reward for soft constraint) in Figure 6 and Figure 7. We surprisingly find that soft-only models achieve significantly higher reward scores during training yet

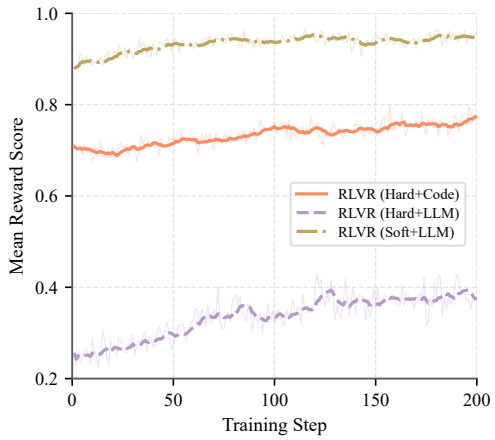


Figure 6: Training rewards of three Qwen3-4B RLVR variants over the first 200 steps.

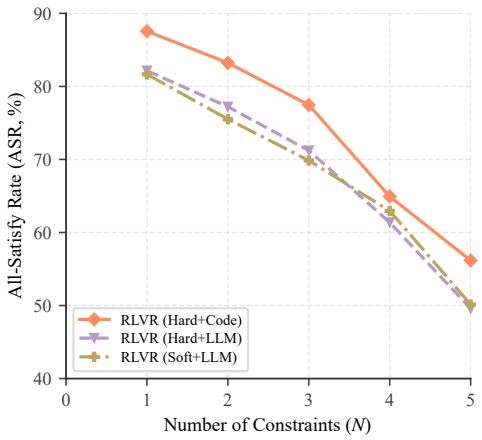


Figure 7: FollowBench ASR of the same RLVR variants as constraint count increases.

ultimately deliver lower test performance. This is mainly because soft-only models explore biases in LLM-as-a-judge to game the reward scores, rather than genuinely mastering constraint adherence. This finding indicates that LLM-based rewards lead to inflated reward scores, which do not translate into improved test performance. Therefore, we argue that directly using LLMs as rewards is currently unreliable.

E Implementation Details of Contrastive Signal Extraction

This appendix provides the full specification of the contrastive signal extraction procedure summarized in Section 4.

Log-Probability vs. Logit Equivalence. Although Eq. equation 4.1 is defined on raw logits, in practice we compute $\Delta_{k,t}$ using top-K log-probabilities returned by the teacher API. Let $\ell_{k,t}(v) = \log \pi_T(v \mid q_k, y_{<t})$ denote the log-probability. Since $\ell_{k,t}(v) = z_{k,t}(v) - \log Z_{k,t}$ where $\log Z_{k,t}$ is the position-dependent log-partition function, we have:

$$\ell_{k,t}(v) - \ell_{0,t}(v) = z_{k,t}(v) - z_{0,t}(v) + (\log Z_{0,t} - \log Z_{k,t}).$$

The term $(\log Z_{0,t} - \log Z_{k,t})$ is constant across all tokens v at position t. Because all downstream operations (min-pooling, normalization) are either shift-invariant or performed within a fixed support set, this constant does not affect the resulting target distribution. We therefore use log-probabilities directly without explicit correction.

Top-K Support Set and Missing-Token Fallback. All contrastive computations are restricted to the anchor support set:

$$\mathcal{S}_t \triangleq \mathrm{topK}\left(z_{x,t}\right), \qquad |\mathcal{S}_t| = K.$$

The anchor $z_{x,t}$ represents the teacher's holistic assessment of the full instruction; its top-K tokens define the semantically relevant subspace at step t, filtering out long-tail noise that would otherwise corrupt min-pooling.

When a support token $v \in \mathcal{S}_t$ is absent from the top-K list returned for variant k (or for the base q_0), we substitute its log-probability with the rank-K (i.e., smallest returned) log-probability of that variant. This serves as a conservative upper bound on the true log-probability, ensuring that missing tokens do not receive spuriously high contrastive scores. In practice, this occurs rarely ($< 2\%$ of evaluations) because the anchor and variant distributions share substantial overlap.

RMS Normalization and Clipping. Different constraints exert vastly different magnitudes of influence on the generation process (e.g., structural format constraints vs. stylistic tone constraints).

Without standardization, min-pooling would be dominated by the strongest constraint, effectively silencing weaker but equally important requirements.

We address this via root-mean-square (RMS) normalization over the support set:

$$\sigma_{k,t} = \sqrt{\frac{1}{|\mathcal{S}_t|} \sum_{v \in \mathcal{S}_t} \Delta_{k,t}(v)^2 + \epsilon}, \qquad \widehat{\Delta}_{k,t}(v) = \mathrm{clip}\left(\frac{\Delta_{k,t}(v)}{\sigma_{k,t}},\ -C,\ C \right), \tag{9}$$

where $\epsilon = 10^{-4}$ stabilizes the normalization denominator and $C = 5.0$ bounds the normalized contrastive signal. RMS normalization is performed separately for each constraint and response position over the anchor support set, without subtracting the mean. It rescales the signal while preserving its sign, and clipping limits the magnitude of individual token contributions.

Hyperparameter Summary. We use a top-K support size of $K = 64$, an RMS stabilization constant of $\epsilon = 10^{-4}$, and a clipping bound of $C = 5.0$. The task-only query q_0 serves as the contrastive reference. The selected constraint weight, soft-min sharpness, target temperature, and learning rate are reported in Appendix F.1, with their sensitivity analysis shown in Figure 8.

F More Experiments Details

F.1 Training Implementation Details

Implementation. We implement CD-OPD using verl, with vLLM for student rollout generation and teacher inference. The teacher remains frozen throughout training. The construction of contrastive signals and the associated numerical settings are described in Appendix E. Here, we report the optimization, hardware, and inference configurations.

Hardware and distributed training. The training setup comprises four nodes, each equipped with eight Ascend 910B2 accelerators, for a total of 32 accelerators. Each accelerator has a nominal device-memory capacity of 60 GiB. Sixteen accelerators are shared between student training and rollout generation, while the remaining sixteen are allocated to teacher inference.

Student training uses Fully Sharded Data Parallel with BF16 mixed precision. We enable gradient checkpointing, padding removal, and parameter and optimizer offloading. Student rollout inference uses tensor-parallel size one. Teacher inference uses four replicas, each with tensor-parallel size four. The vLLM memory-utilization settings are 0.60 for student rollouts and 0.50 for teacher inference.

Optimization and shared hyperparameters. Table 4 summarizes the training configuration. Unless explicitly varied in the hyperparameter analysis or component ablations, the selected hyperparameters are shared across all CD-OPD experiments, without separate tuning for individual teacher–student pairs or evaluation benchmarks. The hyperparameter analysis is presented in Figure 8.

We optimize the student using AdamW with a constant learning rate and no warmup. Training is typically stopped after approximately 200 rollout steps. Each rollout step collects 256 student-generated trajectories and performs one optimization pass using mini-batches of 128. Dynamic batching uses a per-device token budget of 16,384.

Decoding mode and sampling parameters. All Qwen3 models used in our experiments, including the 0.6B, 1.7B, 4B, 8B, and 32B variants, operate in non-thinking mode throughout training and evaluation. We disable thinking by setting `enable_thinking=False` when applying the chat template. For both student rollout generation and benchmark inference, we adopt the temperature, top-p, and top-k settings reported for non-thinking mode in the Qwen3 technical report (Yang et al., 2025): temperature 0.7, top-$p = 0.8$, and top-$k = 20$. These sampling parameters are kept fixed across model sizes and evaluation benchmarks.

Training-time rollouts. During training, the current student generates one response per instruction using the sampling configuration above, allowing up to 8,192 generated tokens. The prompt-length budget is obtained from the dataset manifest, and the inference context budgets are configured to

Table 4: Shared training configuration for CD-OPD. Parameters are kept fixed unless explicitly varied in the hyperparameter analysis or component ablations. Settings for contrastive signal extraction are provided separately in Appendix E.

Parameter	Value
Constraint weight α	1.0
Soft-min sharpness β	5.0
Target temperature τ	1.0
Learning rate η	5×10^{-7}
Optimizer	AdamW
Adam moment coefficients	$(0.9, 0.999)$
Weight decay	0.01
Gradient clipping norm	1.0
Learning-rate schedule	Constant
Warmup ratio	0
Typical training duration	Approximately 200 rollout steps
Rollout batch size	256
Optimization mini-batch size	128
Passes per rollout batch	1
Per-device token budget	16,384
Training precision	BF16 mixed precision

Table 5: Generation settings for training-time student rollouts and benchmark inference. Both stages use the same non-thinking sampling parameters. Benchmark settings apply to IFEval, IFBench, FollowBench, and CFBench. Response limits count newly generated tokens.

Setting	Training rollouts	Benchmark inference
Sampling temperature	0.7	0.7
Sampling top-p	0.8	0.8
Sampling top-k	20	20
Responses per prompt per run	1	1
Maximum response length	8,192	10,240
Thinking mode	Disabled	Disabled
Tensor-parallel size	1	8

accommodate the corresponding prompts and response allowance. The teacher scores the student-generated trajectories under the required prompt views.

The rollout sampling temperature of 0.7 controls student generation and is distinct from the target temperature $\tau = 1.0$, which controls the synthesized distillation distribution.

Benchmark inference. For benchmark evaluation, models are served through vLLM in BF16 on a single node with eight Ascend accelerators, using tensor-parallel size eight and pipeline-parallel size one. The maximum context length is 32,768 tokens.

For IFEval, IFBench, FollowBench, and CFBench, we use the same non-thinking sampling configuration and allow up to 10,240 generated tokens. Each evaluation run generates one response per instruction. Table 5 summarizes the settings for training-time rollouts and benchmark inference.

F.2 Benchmark Details

IFEval (Zhou et al., 2023) is a widely adopted benchmark designed to evaluate the ability of LLMs to follow objective and verifiable instructions. It consists of around 500 prompts containing 25 types of verifiable constraints (e.g., word count limits, formatting requirements).

IFBench (Pyatkin et al., 2025) addresses the issue of model overfitting to common instruction datasets. It focuses on evaluating the generalization capabilities of models by introducing 58 novel, unseen, and challenging verifiable constraints. It employs strict code-based verification modules to measure performance on out-of-domain instructions.

FollowBench (Jiang et al., 2024) evaluates the robustness of LLMs through a multi-level difficulty mechanism. It contains 820 instructions across five fine-grained categories (Content, Situation, Style, Format, Example). The benchmark is constructed by incrementally adding constraints to seed instructions, creating a difficulty gradient (Level 1 to Level N).

CFBench (Zhang et al., 2025) is a benchmark comprising 1,000 high-quality samples derived from the real world. It features a hierarchical taxonomy with 10 major constraint categories (e.g., style, logical rules, numerical) and over 25 subcategories. We report the average results across all samples. To ensure consistency with prior work, we use the official evaluation code, which retains only integers.

G Additional Results & Analysis in Main

G.1 Hyperparameter Analysis

We examine the sensitivity of CD-OPD to four hyperparameters: the constraint-enforcement weight α, soft-min sharpness β, target-distribution temperature τ, and learning rate η. Figure 8 presents two parameter sweeps on IFEval and IFBench.

Constraint Weight and Pooling Sharpness. We jointly vary $\alpha \in \{0.5, 1.0, 1.5, 2.0\}$ and $\beta \in \{0.5, 1.0, 5.0, 10.0\}$, fixing $\eta = 5 \times 10^{-7}$ and $\tau = 1.0$. As shown in Figure 8(a,b), $\alpha = 1.0$ and $\beta = 5.0$ yield the highest scores on both benchmarks: 81.95% on IFEval and 29.93% on IFBench. At $\alpha = 1.0$, increasing β from 1.0 to 5.0 improves the scores by 2.50 and 1.34 percentage points, respectively. Further increasing β to 10.0 reduces them by 0.91 and 1.51 points. Similarly, at $\beta = 5.0$, increasing α from 1.0 to 2.0 reduces the scores by 2.91 and 2.19 points. Thus, neither stronger constraint correction nor sharper pooling consistently improves performance within the evaluated range. The effect of β also depends on α: at $\alpha = 0.5$, increasing β from 5.0 to 10.0 improves both scores, highlighting the importance of considering these parameters jointly.

Learning Rate and Target Temperature. We vary $\eta \in \{10^{-6}, 5 \times 10^{-7}\}$ and $\tau \in \{0.8, 1.0, 1.2\}$, fixing $\alpha = 1.0$ and $\beta = 5.0$. Here, τ controls the temperature of the synthesized teacher target. Figure 8(c,d) shows that $\tau = 1.0$ achieves the highest scores on both benchmarks at each learning rate. The lower learning rate, 5×10^{-7}, also outperforms 10^{-6} at every evaluated temperature. At $\tau = 1.0$, reducing the learning rate improves IFEval and IFBench scores by 1.61 and 1.37 percentage points, respectively.

Both parameter sweeps attain their highest scores at the same configuration: $\alpha = 1.0$, $\beta = 5.0$, $\tau = 1.0$, and $\eta = 5 \times 10^{-7}$. This configuration provides an empirical reference for parameter selection within the tested ranges; the two sweeps do not constitute an exhaustive search over all four hyperparameters.

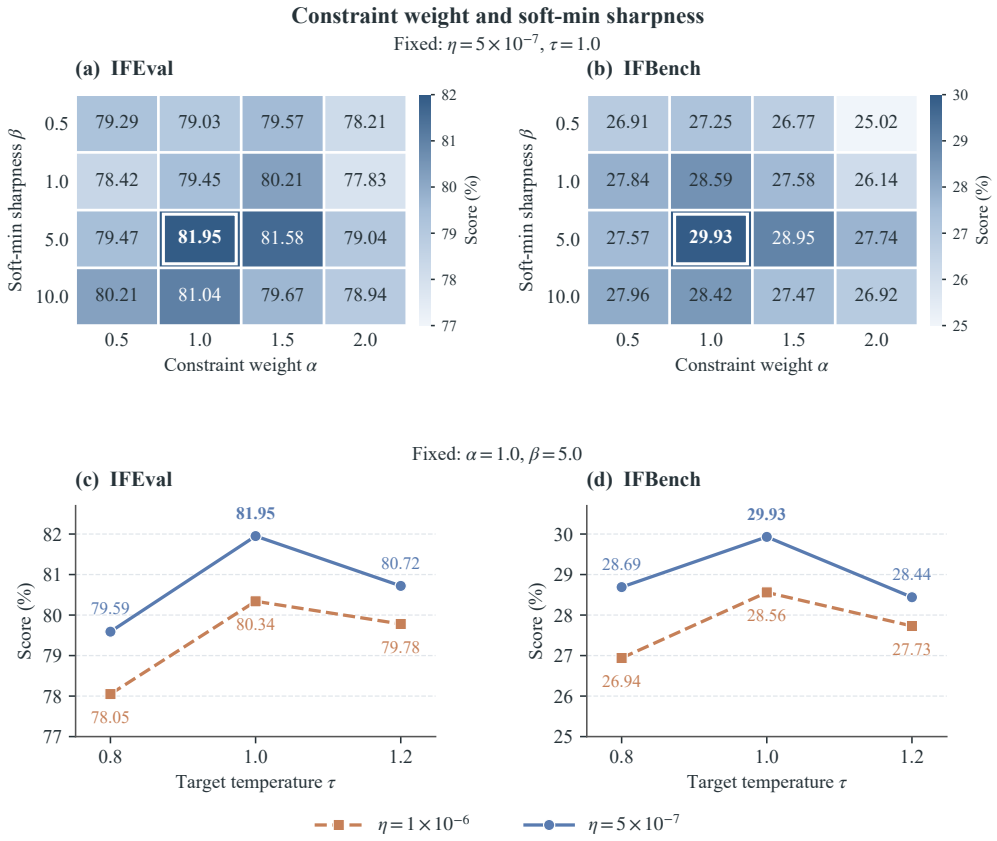


Figure 8: **Hyperparameter sensitivity of CD-OPD on IFEval and IFBench. (a,b)** Joint sweep of the constraint-enforcement weight α and soft-min sharpness β, with learning rate $\eta = 5 \times 10^{-7}$ and target temperature $\tau = 1.0$. Outlined cells mark the highest score within each grid; heatmap rows represent discrete β settings. **(c,d)** Target-temperature sweep at two learning rates, with $\alpha = 1.0$ and $\beta = 5.0$. Lines connect evaluated configurations. All scores are percentages, with higher values indicating better performance. Color scales and vertical score ranges are specific to each benchmark. Point estimates are shown without uncertainty intervals.