LEARNING FROM GRADED FEEDBACK: TIER-AWARE SELF-PLAY LISTWISE OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline preference optimization relies on fixed response comparisons that become increasingly mismatched to an evolving policy. Self-play mitigates this mismatch by regenerating responses from the current policy. Extending self-play to listwise learning is challenging because multi-constraint evaluations often assign multiple responses the same discrete score, producing ordered tiers with ties. Strict-order listwise objectives require tie-breaking, which introduces unsupported preferences, while binary pass/fail feedback provides no relative signal when all rollouts are imperfect. We propose **Tier-Aware Self-Play Listwise Optimization (T-SPLO)**, a framework for learning from graded verifier feedback. At each round, prompts are prioritized according to their current tier gap, and the policy generates multiple rollouts that an objective verifier groups into tiers according to constraint satisfaction. T-SPLO optimizes the likelihood of the resulting tiered order by marginalizing over all within-tier permutations using subset dynamic programming. The resulting objective is invariant to permutations among equally scored responses and learns from different degrees of partial success. Across two backbones and three benchmarks, T-SPLO achieves the highest final average instruction-following score among the compared baselines, with end-to-end ablations favoring tie preservation and average scores improving across training rounds. Our code is available at `https://anonymous.4open.science/r/T-SPLO-956D/`.

1 INTRODUCTION

Learning to follow instructions requires language models to satisfy multiple constraints within a single response (Zhou et al., 2023; Guo et al., 2026). In verifiable tasks, feedback can identify how many constraints each response satisfies, providing a graded measure of success even when no candidate is fully correct. Preference optimization offers a way to learn from response comparisons (Rafailov et al., 2023; Meng et al., 2024), while self-play methods regenerate candidates as the policy evolves (Chen et al., 2024; Wu et al., 2025). Together, these approaches provide a setting for learning from the current policy's successes and errors. The central question is how to turn graded feedback on these candidates into effective training supervision.

Common supervision formats capture different parts of this feedback. Binary pass/fail labels distinguish complete success from failure, but cannot separate different degrees of partial success. Selecting a single preferred–rejected pair uses a local comparison without representing the quality structure of the full candidate set. Listwise objectives can incorporate multiple candidates jointly (Song et al., 2024; Liu et al., 2025), but a strict ranking requires an order even between responses that receive the same verifier score. Such an order introduces preferences that the verifier does not support. Effective use of graded feedback therefore requires preserving both differences in quality and ties between equally scored responses.

Figure 1 illustrates this requirement with four responses satisfying 2, 2, 1, and 0 of three constraints. Binary labels mark all four as failures, although their degrees of success differ. A strict ranking distinguishes the two responses satisfying two constraints, despite their equal scores. The verifier instead supports three ordered tiers: the two responses satisfying two constraints rank above the response satisfying one, which ranks above the response satisfying none. This structure retains useful comparisons among imperfect responses without imposing an arbitrary order within a tier. It motivates learning directly from the likelihood of the observed tiered order.

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.

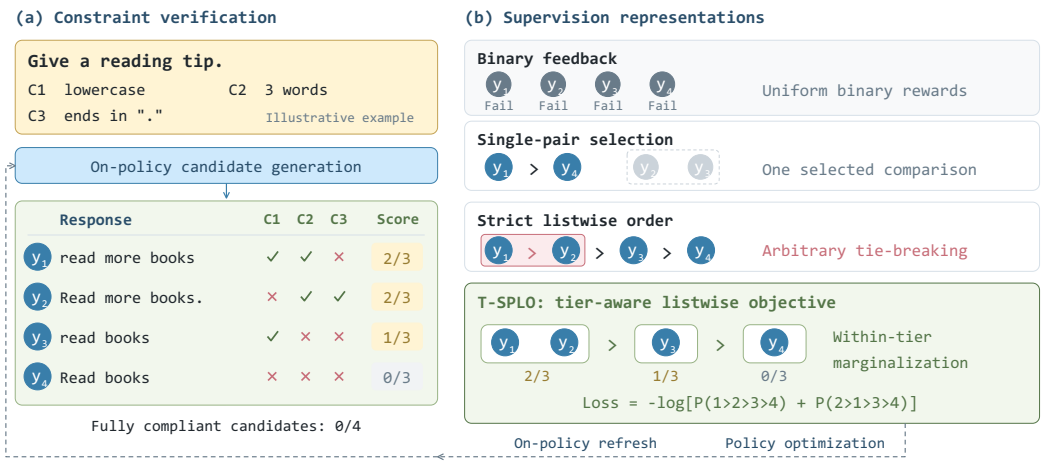


Figure 1: Graded feedback and supervision representations. (a) Four responses satisfy 2, 2, 1, and 0 of three constraints; fractions show satisfied/total counts. (b) T-SPLO sums Plackett–Luce probabilities over tier-consistent orders.

We introduce **Tier-Aware Self-Play Listwise Optimization (T-SPLO)**, which uses graded verifier feedback for response ranking and prompt selection. At each round, a tier-gap sampler prioritizes prompts with more constraints left unsatisfied by the current policy, while retaining uniform exploration. The policy generates multiple rollouts, and the verifier groups them into ordered quality tiers. A tie-aware Plackett–Luce objective then sums the probabilities of all strict orders consistent with these tiers, using subset dynamic programming for exact computation. This objective learns preferences across tiers and is invariant to permutations within a tier. After updating the policy, T-SPLO recomputes prompt gaps and generates fresh candidates, adapting both prompt selection and response supervision to the evolving model.

Experiments on IFEval, IFBench, and Multi-IF show that T-SPLO achieves the highest final average score among the compared methods on both Qwen2.5-3B and Llama-3.2-3B, exceeding the strongest baseline by 1.69 and 2.51 percentage points, respectively. Initial objective-level analyses characterize partial-success signals and tie-breaking noise; shared-list training comparisons assess their empirical effects. Our main contributions are as follows:

- We introduce **T-SPLO**, a self-play framework that uses graded constraint feedback to construct response tiers and guide prompt selection as the policy evolves.
- We apply a **tie-aware Plackett–Luce objective** to verifier-defined tiers and compute its likelihood exactly with subset dynamic programming.
- We demonstrate improved final average instruction-following scores across two model families and three benchmarks, with end-to-end ablations and controlled objective analyses.

2 Related Work

Self-play preference optimization. One line of work refreshes training data as the policy changes. SPIN contrasts generated responses with fixed demonstrations (Chen et al., 2024), while SPPO, iterative DPO, online AI feedback, and self-rewarding training generate fresh candidates during training (Wu et al., 2025; Xiong et al., 2023; Guo et al., 2024; Yuan et al., 2024). Related work on preference-data construction includes rejection sampling optimization (RSO), an offline approach that uses reward-guided rejection sampling to approximate a target optimal policy (Liu et al., 2024). T-SPLO uses on-policy candidate generation and learns from verifier-defined tiers across multiple responses.

Verifiable instruction following. Executable feedback supports both instruction-data construction and policy training. AutoIF generates instructions, verification code, and test cases, then uses execution feedback to construct data for SFT and offline or online DPO (Dong et al., 2025a). VIF-

RAG combines atomic instructions and verifies synthetic data for instruction-following alignment in retrieval-augmented generation (Dong et al., 2025b). RECAST synthesizes multi-constraint instructions with rule-based and model-based validators; its RLVC procedure (Guo et al., 2026) uses the mean constraint satisfaction rate as a reward for GRPO (Shao et al., 2024). Table 1 includes GRPO with this graded reward as a baseline. Thus, graded constraint feedback is already useful beyond binary success labels. T-SPLO studies how to represent this feedback as ordered response tiers and optimize their joint likelihood, using fixed programmatic checkers and refreshing candidates through self-play.

Listwise preference optimization. Preference optimization began with pairwise objectives: DPO learns from a chosen–rejected pair through policy-to-reference likelihood ratios (Rafailov et al., 2023), and SimPO removes the reference policy using length-normalized log probability with a target margin (Meng et al., 2024). Multi-response objectives use several scored candidates jointly, as in RRHF and PRO (Yuan et al., 2023; Song et al., 2024), LiPO's connection to learning-to-rank (Liu et al., 2025), and λ-DPO's controllable multi-objective weighting (Sun et al., 2025). MoPLEx learns mixtures of Plackett–Luce models through expectation-maximization to represent heterogeneous annotator preferences (Li et al., 2026); T-SPLO instead uses a single verifier-defined tier order for each candidate list. Strict-order formulations require tie-breaking when multiple responses achieve identical discrete verifier scores. More generally, partitioned-preference learning represents ordered groups without specifying their internal order; Ma et al. (2021) compute the corresponding Plackett–Luce likelihood through numerical integration. T-SPLO uses this established likelihood formulation for verifier-defined tiers in iterative self-play and evaluates it exactly with subset dynamic programming on short candidate lists.

Curriculum and difficulty-weighted prompt selection. Under a limited generation budget, iterative alignment methods must also decide which prompts to prioritize. Active preference learning selects prompts and response pairs using the language model's predictive entropy and the implicit preference model's certainty (Muldrew et al., 2024), while curriculum alignment schemes order samples by empirical difficulty (Pattnaik et al., 2024). We do not claim difficulty-weighted prompt selection as a standalone algorithmic invention. Instead, T-SPLO's tier-gap prompt sampler reuses the graded verifier signal that structures the loss: prompts with larger constraint deficits receive higher sampling priority, while a uniform component limits sampling concentration. This combines prompt-level exploration and candidate-level optimization under a single graded feedback mechanism.

3 Method: Tier-Aware Self-Play Listwise Optimization

We consider a training pool $\mathcal{D}$ of prompts with verifiable constraints. Each prompt x has $k(x)$ constraints, and the policy $\pi_\theta(y \mid x)$ generates a response y with parameters θ. Our goal is to improve joint constraint satisfaction by learning from the different degrees of success among candidate responses. As illustrated in Figure 2, T-SPLO comprises three components: tier construction represents graded response quality (Section 3.1); a tie-aware listwise objective learns the resulting partial order (Section 3.2); and iterative self-play refreshes candidates with tier-gap prompt selection (Section 3.3). The verifier provides feedback at two levels: response scores determine the training order, while the current policy's unsatisfied constraints determine which prompts to sample.

3.1 Tier Construction from Graded Feedback

To preserve degrees of partial success, we construct response tiers from constraint-level verification. At round $t \in \{0, 1, 2\}$, the current policy π_{θ_t} generates rollouts for each selected prompt; θ_0 denotes the starting model. Let $Y_t(x) = (y_1, \ldots, y_G)$ denote the retained candidate list of size G. Our implementation generates eight stochastic rollouts and one greedy rollout per prompt, which are filtered before list construction; thus G can vary across prompts. A programmatic verifier evaluates each response against every constraint:

$$v_j(x, y) \in \{0, 1\}, \qquad s(x, y) = \sum_{j=1}^{k(x)} v_j(x, y). \tag{1}$$

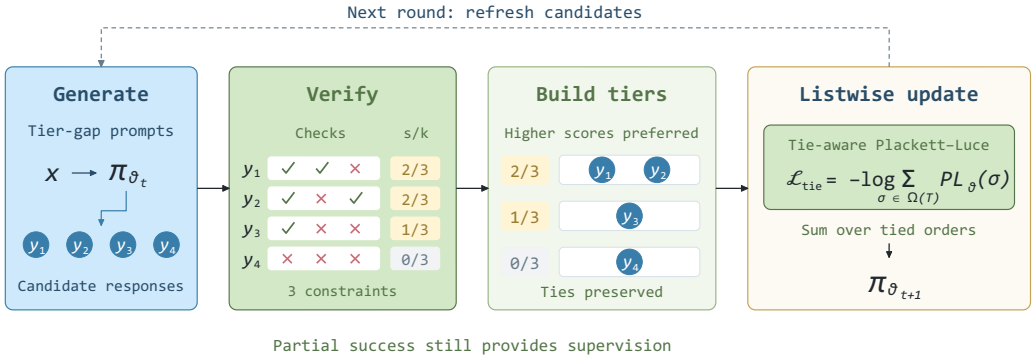


Figure 2: Overview of T-SPLO. A greedy audit guides tier-gap prompt sampling. Fractions show satisfied/total constraints. Verified responses form ordered tiers; the tie-aware Plackett–Luce objective marginalizes over within-tier orders. The updated policy generates the next round's candidates.

Here, $v_j(x, y) = 1$ means that response y satisfies the j-th constraint, and $s(x, y)$ counts the satisfied constraints. Full compliance corresponds to $s(x, y) = k(x)$; intermediate scores identify partial success.

The score structure determines which preferences the verifier supports. Let $a_1 < \cdots < a_M$ be the M distinct scores present in the candidate list. We group candidates with the same score into tiers:

$$T_m = \{y_i \in Y_t(x) : s(x, y_i) = a_m\}, \qquad n_m = |T_m|, \qquad \sum_{m=1}^{M} n_m = G. \tag{2}$$

The tiers are ordered from worst to best, $T_1 \prec \cdots \prec T_M$. Responses in different tiers have a verified preference relation, whereas responses within one tier are tied. We retain lists with $M \geq 2$, since a single tier supplies no cross-tier ordering. For example, scores $(2, 2, 1, 0)$ under three constraints give $\{y_4\} \prec \{y_3\} \prec \{y_1, y_2\}$. This list provides graded supervision even though no response is fully compliant.

3.2 Tie-Aware Listwise Objective

The objective should favor higher-quality tiers without assigning ranking labels to equally scored responses. We therefore optimize the likelihood of the tier order by marginalizing over all orders within each tie. Following direct preference optimization (Rafailov et al., 2023), we first define an implicit reward:

$$r_\theta(x, y) = \beta\big[\log \pi_\theta(y \mid x) - \log \pi_{\text{ref}}(y \mid x)\big]. \tag{3}$$

The reference policy $\pi_{\text{ref}} = \pi_{\theta_t}$ is frozen at the start of round t, while θ changes during that round's updates. The positive coefficient β scales the log-probability ratio. Each response log-probability sums the conditional log-probabilities of its generated tokens, excluding the prompt. Thus, r_θ measures the change in preference for a response relative to the round's starting policy.

Likelihood of a tiered order. For one fixed prompt, write $r_i = r_\theta(x, y_i)$. A permutation $\sigma = (\sigma_1, \ldots, \sigma_G)$ lists candidate indices from best to worst. Its Plackett–Luce probability (Plackett, 1975) is

$$\text{PL}_\theta(\sigma \mid x, Y_t) = \prod_{j=1}^{G} \frac{\exp(r_{\sigma_j})}{\sum_{\ell=j}^{G} \exp(r_{\sigma_\ell})}. \tag{4}$$

At position j, the numerator is the weight of the chosen response, and the denominator sums the weights of all responses not yet selected. Write $T = (T_1, \ldots, T_M)$ for the ordered tier collection, and let $\Omega(T)$ contain the permutations that place every higher-tier response before every lower-tier response, allowing any order within a tier. There are $\prod_{m=1}^{M} n_m!$ such permutations. The tie-aware

loss is

$$\mathcal{L}_{\text{tie}}(\theta; x, Y_t) = -\log \sum_{\sigma \in \Omega(T)} \text{PL}_\theta(\sigma \mid x, Y_t). \tag{5}$$

This is the partitioned-preference likelihood studied in learning-to-rank (Ma et al., 2021), here applied to verifier-defined response tiers. The sum assigns probability to the verifier-supported tier order as a whole. Consequently, equally scored candidates do not require arbitrary ranking labels, while differences in partial success contribute cross-tier comparisons.

Exact computation by dynamic programming. Direct enumeration requires $\prod_m n_m!$ strict orders. We instead reuse the probabilities of remaining candidate subsets to compute the same marginal likelihood exactly. For tier T_m, let $w_i = \exp(r_i)$ and $B_m = \sum_{\ell<m} \sum_{y_j \in T_\ell} w_j$, the total weight of all lower-tier responses. Define $F_m(S)$ as the probability that every response in the remaining subset $S \subseteq T_m$ is selected before any lower-tier response. With $F_m(\varnothing) = 1$, the recurrence is

$$F_m(S) = \sum_{y_i \in S} \frac{w_i}{B_m + \sum_{y_j \in S} w_j} F_m(S \setminus \{y_i\}). \tag{6}$$

Each term selects one response from S next and then completes the remaining subset. Higher tiers have already been selected, so they do not appear in the denominator. Applying this calculation from the highest tier downward gives $\mathcal{L}_{\text{tie}} = -\sum_{m=2}^{M} \log F_m(T_m)$; the lowest tier contributes probability one. The recurrence uses 2^{n_m} states and at most n_m transitions per state, giving $O(\sum_{m=2}^{M} n_m 2^{n_m})$ time with precomputed subset weight sums. We evaluate probabilities in log space for numerical stability. Appendix C.1 proves the recurrence and factorization, including the boundary cases.

List normalization. In training, we use $\overline{\mathcal{L}}_{\text{tie}} = \mathcal{L}_{\text{tie}}/(M-1)$, averaging over the nontrivial tier boundaries so lists with more tiers do not receive a larger weight simply from summing more terms.

3.3 Iterative Self-Play with Tier-Gap Prompt Selection

Selecting prompts from current constraint deficits. Graded feedback also identifies where the current policy leaves constraints unsatisfied. To direct candidate generation toward these prompts, at the start of each round we greedily decode one response $y_t^{\text{g}}(x)$ for every prompt in $\mathcal{D}$. Its tier gap is $d_t(x) = k(x) - s(x, y_t^{\text{g}}(x))$, the number of constraints the current greedy response leaves unsatisfied. A larger gap identifies a prompt with more remaining constraint violations. We assign sampling weights

$$q_t(x) = (1-\eta)\frac{\exp(d_t(x)/\tau)}{\sum_{x' \in \mathcal{D}} \exp(d_t(x')/\tau)} + \eta \frac{1}{|\mathcal{D}|}. \tag{7}$$

Here, $\tau > 0$ controls how strongly prompt sampling favors larger gaps, $\eta \in [0,1]$ is the uniform exploration weight, and $|\mathcal{D}|$ is the pool size. The first term prioritizes unsatisfied constraints; the second ensures $q_t(x) \geq \eta/|\mathcal{D}|$. We use $\tau = 1$ and $\eta = 0.3$ and draw N_p distinct prompts without replacement using these weights. The resulting selected set is denoted by $\mathcal{S}_t$.

Updating and regenerating candidates. Self-play keeps the candidate distribution aligned with the policy being improved. The round-start policy generates rollouts for $\mathcal{S}_t$, and the verifier constructs tiers from the retained candidates as in Section 3.1. Let $\mathcal{V}_t \subseteq \mathcal{S}_t$ contain the prompts whose retained lists have at least two tiers. For nonempty $\mathcal{V}_t$, we start from θ_t and take a fixed number of optimization steps on

$$\mathcal{J}_t(\theta) = \frac{1}{|\mathcal{V}_t|} \sum_{x \in \mathcal{V}_t} \overline{\mathcal{L}}_{\text{tie}}(\theta; x, Y_t(x)). \tag{8}$$

Candidate texts, verifier scores, and the reference policy remain fixed during these updates. The resulting parameters define θ_{t+1}; we then recompute prompt gaps and regenerate candidate lists from the updated policy. This alternation uses the same graded feedback to improve response ranking and adapt prompt selection to the policy's current behavior. Our experiments use three rounds, $N_p = 1{,}024$ prompts per round, and 256 optimization steps per round. At inference, the trained policy generates responses directly without candidate ranking or verifier-based selection.

Uniform prompt sampling is evaluated in Section 4.3.

Table 1: Main results (%) on IFEval, IFBench, and Multi-IF after training rounds 0–2 on two backbones. Overall averages the three benchmark scores.

Round 0

	Qwen2.5-3B-Instruct						Llama-3.2-3B-Instruct					
	IFEval		IFBench		Multi-IF	Overall	IFEval		IFBench		Multi-IF	Overall
Method	P-strict	P-loose	P-strict	P-loose	T-strict	Avg.	P-strict	P-loose	P-strict	P-loose	T-strict	Avg.
Starting model	*61.74*	*65.62*	*24.67*	*27.67*	*39.06*	*41.82*	*70.61*	*75.60*	*25.67*	*29.67*	*53.57*	*49.95*
RFT (SFT on positives)	58.78	62.29	**27.00**	**29.67**	38.32	**41.37**	71.72	78.00	**26.33**	30.33	54.72	50.92
DPO	61.18	64.88	23.67	26.67	38.98	41.28	71.35	76.71	23.67	28.00	54.38	49.80
SimPO	59.89	63.59	23.67	27.00	38.43	40.66	71.53	76.34	24.67	28.00	54.61	50.27
λ-DPO (m=1)	**62.11**	**65.43**	23.33	26.33	38.17	41.20	70.24	75.42	22.67	26.67	54.90	49.27
SPPO	60.81	63.77	24.00	27.00	39.13	41.31	68.58	74.12	26.00	**30.67**	54.24	49.61
GRPO	59.70	63.40	23.33	25.33	39.02	40.68	71.16	76.52	25.67	29.67	52.76	49.86
T-SPLO (ours)	60.63	64.14	24.00	27.67	**39.20**	41.28	**73.38**	**78.56**	25.33	29.67	**56.37**	**51.69**

Round 1

	Qwen2.5-3B-Instruct						Llama-3.2-3B-Instruct					
	IFEval		IFBench		Multi-IF	Overall	IFEval		IFBench		Multi-IF	Overall
Method	P-strict	P-loose	P-strict	P-loose	T-strict	Avg.	P-strict	P-loose	P-strict	P-loose	T-strict	Avg.
Starting model	*61.74*	*65.62*	*24.67*	*27.67*	*39.06*	*41.82*	*70.61*	*75.60*	*25.67*	*29.67*	*53.57*	*49.95*
RFT (SFT on positives)	58.60	61.92	26.00	29.00	37.77	40.79	70.43	76.16	24.33	27.67	55.97	50.24
DPO	62.48	**66.36**	**27.67**	**30.67**	40.53	43.56	75.05	79.85	25.00	29.67	60.35	53.47
SimPO	**63.03**	65.43	27.33	29.33	40.83	**43.73**	74.31	78.93	24.33	29.67	59.51	52.72
λ-DPO (m=1)	62.29	**66.36**	26.67	30.33	39.35	42.77	72.09	77.82	25.33	29.33	57.30	51.57
SPPO	62.85	65.99	25.67	28.67	39.50	42.67	73.20	78.56	25.00	30.33	57.18	51.79
GRPO	60.63	64.14	26.00	28.33	38.54	41.72	72.09	77.08	25.33	**31.00**	57.30	51.57
T-SPLO (ours)	60.81	64.33	26.33	29.33	**41.08**	42.74	**76.52**	**81.33**	**26.33**	**31.00**	**62.60**	**55.15**

Round 2

	Qwen2.5-3B-Instruct						Llama-3.2-3B-Instruct					
	IFEval		IFBench		Multi-IF	Overall	IFEval		IFBench		Multi-IF	Overall
Method	P-strict	P-loose	P-strict	P-loose	T-strict	Avg.	P-strict	P-loose	P-strict	P-loose	T-strict	Avg.
Starting model	*61.74*	*65.62*	*24.67*	*27.67*	*39.06*	*41.82*	*70.61*	*75.60*	*25.67*	*29.67*	*53.57*	*49.95*
RFT (SFT on positives)	59.33	61.74	27.33	30.33	37.58	41.41	70.24	75.05	24.67	29.67	54.90	49.94
DPO	61.74	65.25	**31.00**	**34.00**	41.19	44.64	76.89	**81.33**	25.00	29.33	62.79	54.89
SimPO	62.66	66.36	27.67	31.33	40.94	43.76	72.27	76.16	26.67	30.33	61.68	53.54
λ-DPO (m=1)	63.40	67.47	29.00	30.67	42.34	44.91	74.68	78.93	27.33	29.33	60.24	54.08
SPPO	64.14	66.91	25.67	29.67	39.54	43.12	73.01	77.82	23.67	29.33	57.89	51.52
GRPO	63.22	65.80	26.00	28.00	40.53	43.25	74.12	79.11	24.33	28.00	58.00	52.15
T-SPLO (ours)	**65.62**	**68.76**	**31.00**	33.33	**43.18**	**46.60**	**78.00**	80.78	**28.00**	**32.67**	**66.21**	**57.40**

P-strict/P-loose: prompt-level strict/loose accuracy; T-strict: turn-level strict accuracy. Avg. weights the three strict benchmark scores equally. T-SPLO uses tier-gap selection; baselines sample uniformly. GRPO uses reward s/k and refreshes rollouts within each round; the other methods refresh between rounds. All rows use seed 1. Bold and underlined values mark the best and second-best distinct trained scores; ties share markings.

4 Experiments

4.1 Experimental Setup

Models and training. We evaluate T-SPLO on Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct using 4,096 Nectar prompts (Zhu et al., 2024) augmented with IFEval constraints. Each round selects 1,024 prompts and takes 256 optimization steps; round indices 0–2 report checkpoints after one–three completed rounds. Candidates are generated with vLLM (Kwon et al., 2023); LoRA adapters (Hu et al., 2021) are trained with AdamW (Loshchilov & Hutter, 2017).

Benchmarks and evaluation. Evaluation uses greedy decoding on IFEval (Zhou et al., 2023), IFBench (Pyatkin et al., 2026), and Multi-IF (He et al., 2024). Prompt-strict accuracy requires all checked constraints to pass; Overall averages IFEval prompt-strict, IFBench prompt-strict, and Multi-IF turn-level strict accuracy equally. Appendix A provides the full protocol.

Table 2: General-domain results (%) for the final checkpoints on eight benchmarks. Results are grouped by starting model and capability; Overall weights the four capabilities equally.

	Math			Knowledge			Commonsense				Reasoning	Overall
Method	GSM8K	MATH	Avg.	MMLU	ARC-C	Avg.	HellaS.	PIQA	WinoG.	Avg.	BBH	Avg.
Qwen2.5-3B-Instruct												
Starting model	64.37	16.80	40.58	65.41	48.04	56.72	74.93	77.64	68.90	73.82	40.90	53.01
RFT (SFT on positives)	64.75	12.28	38.51	65.39	47.18	56.29	74.87	75.84	69.85	73.52	39.67	52.00
DPO	**65.88**	**17.68**	**41.78**	65.38	47.87	56.62	74.89	77.37	69.53	73.93	41.04	**53.34**
SimPO	64.44	15.48	39.96	**65.45**	48.12	56.78	**74.99**	**77.48**	69.77	74.08	40.93	52.94
λ-DPO (m=1)	64.22	16.16	40.19	65.43	47.78	56.60	74.97	77.37	69.61	73.98	40.95	52.93
SPPO	63.61	16.60	40.10	65.38	47.44	56.41	74.88	**77.48**	**69.93**	**74.09**	40.68	52.82
GRPO	63.99	16.34	40.16	65.41	**48.29**	**56.85**	74.83	77.42	69.46	73.90	40.38	52.82
T-SPLO (ours)	64.52	14.86	39.69	65.35	47.61	56.48	74.98	**77.48**	68.90	73.78	**41.21**	52.79
Llama-3.2-3B-Instruct												
Starting model	67.40	31.42	49.41	62.31	46.25	54.28	71.56	75.90	68.59	72.01	39.66	53.84
RFT (SFT on positives)	67.85	31.74	**49.80**	**62.21**	45.99	54.10	70.88	76.01	68.51	71.80	**40.01**	**53.93**
DPO	66.49	31.74	49.11	61.98	46.59	54.28	71.57	75.90	68.43	71.97	39.47	53.71
SimPO	66.64	31.28	48.96	62.13	**46.76**	**54.44**	**71.90**	76.28	68.27	72.15	39.39	53.74
λ-DPO (m=1)	**68.08**	31.50	49.79	62.13	46.42	54.27	71.78	**76.39**	68.59	**72.25**	39.30	53.90
SPPO	67.32	31.56	49.44	62.15	45.99	54.07	71.67	76.17	68.67	72.17	39.70	53.85
GRPO	67.40	31.56	49.48	62.20	46.08	54.14	71.51	76.06	**68.82**	72.13	39.73	53.87
T-SPLO (ours)	67.70	**31.82**	49.76	61.94	46.42	54.18	71.65	75.84	68.75	72.08	39.35	53.84

Avg. is the within-category mean; Overall weights the four categories equally. Bold and underlined values indicate the best and second-best trained methods. The starting model is the untrained reference.

4.2 Main Results

Instruction-Following Performance. Table 1 compares T-SPLO with positive-only RFT, four preference objectives (DPO, SimPO, λ-DPO, and SPPO), and GRPO (Shao et al., 2024) using the graded reward s/k as in RLVC (Guo et al., 2026). **(1) Higher final instruction-following performance.** After three rounds, T-SPLO achieves the best Overall scores of 46.60% on Qwen and 57.40% on Llama, exceeding the strongest baseline by 1.69 and 2.51 percentage points. Its gains over RFT are 5.19 and 7.46 points, and over graded-reward GRPO are 3.35 and 5.25 points. With uniform prompt sampling, T-SPLO retains a 2.88-point lead over GRPO on Qwen (Table 3). **(2) Gains across single-turn and multi-turn tasks.** On Qwen, T-SPLO leads IFEval and Multi-IF by 1.48 and 0.84 points over the best baseline for each metric, and matches the best IFBench score of 31.00%. On Llama, it leads all three primary metrics, with its largest margin of 3.42 points on Multi-IF. **(3) Continued improvement through self-play.** Overall rises from 41.28% to 42.74% to 46.60% on Qwen and from 51.69% to 55.15% to 57.40% on Llama. From round 0 to round 2, Overall gains 5.32 points on Qwen and 5.71 on Llama.

General-Domain Capability. T-SPLO combines stronger instruction following with aggregate general-domain performance close to that of the starting models (Table 2). Its instruction-following gains of 4.78 points on Qwen and 7.45 points on Llama accompany general-domain Overall scores of 52.79% and 53.84%, compared with 53.01% and 53.84% before training. This evaluation covers GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), MMLU (Hendrycks et al., 2020), ARC-Challenge, HellaSwag, PIQA, WinoGrande, and BBH, weighting math, knowledge, commonsense, and reasoning equally in Overall.

4.3 In-depth Analysis

Ablation Study. Table 3 evaluates end-to-end variants on Qwen2.5-3B with a shared prompt pool, three-round schedule, step budget, learning rate, and evaluation protocol (Appendix A.1). Variants change the objective, candidate regeneration, or prompt selection. Three patterns emerge: **(1) Graded tiers and ties.** T-SPLO gains 1.06 Overall points over binary feedback, 2.31 over random tie-breaking, and 1.70 over keeping one response per tier. Both tie-handling variants score lower on every benchmark. **(2) Cross-tier comparisons.** All-pairs DPO exceeds best–worst DPO by 1.96 points. T-SPLO reaches 46.60%, exceeding them by 0.43 and 2.39 points, respectively. **(3) Candidate and prompt refresh.** Updating prompts, candidates, and the reference policy gains 1.74 points over fixed-data training. Tier-gap sampling with exploration adds 0.47 points over uniform sampling and 0.35 over sampling without exploration.

Table 3: Ablation study (%) on Qwen2.5-3B after three training rounds; w/o means without. (a) Supervision objective. (b) Prompt selection and self-play.

(a) Objective

Variant	IFEval P-strict	IFBench P-strict	Multi-IF T-strict	Overall Avg.
T-SPLO	**65.62**	**31.00**	**43.18**	**46.60**
w/o Graded	64.14	29.67	42.82	45.54
w/o Tie (shuffle)	62.85	29.00	41.01	44.29
w/o Tie (one per tier)	62.66	29.67	42.37	44.90
w/o Listwise (all pairs)	65.25	**31.00**	42.26	46.17
w/o Listwise (best–worst)	62.66	29.00	40.97	44.21

(b) Prompt selection and self-play

Variant	IFEval P-strict	IFBench P-strict	Multi-IF T-strict	Overall Avg.
T-SPLO	**65.62**	31.00	43.18	**46.60**
w/o Tier-gap	63.77	31.33	**43.29**	46.13
w/o Exploration	64.70	**32.33**	41.71	46.25
w/o Self-play	63.03	30.00	41.56	44.86

All variants share the prompt pool, optimization-step budget, learning rate, seed, and main-table evaluation protocol. Metrics follow Table 1; Overall averages the three benchmarks. Graded: binary pass/fail feedback. Tie: the standard Plackett–Luce loss on a strict order, obtained by shuffling tied responses or keeping one response per tier. Listwise: DPO on all cross-tier pairs or on the best–worst pair. Tier-gap: uniform prompt sampling. Exploration: $\eta = 0$. Self-play: fixed round-0 candidates, selected prompts, and reference policy. The T-SPLO row is repeated in both panels. Bold and underlined values indicate the best and second-best distinct scores within each panel; ties receive the same marking.

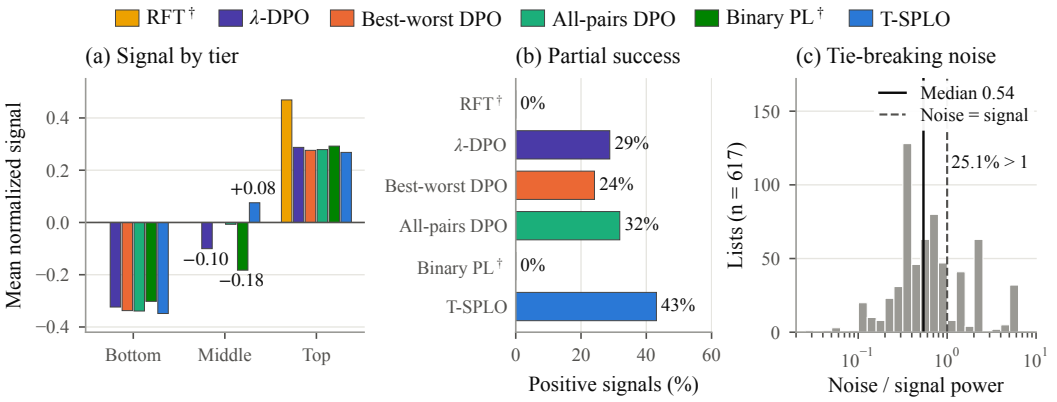


Figure 3: Initial Qwen objective signals. (a) Per-list normalized signals by tier. (b) Positive signals among partial successes (†: 483 lists/1,500 responses; others: 620/2,450). (c) Noise-to-signal power on 617 tied lists; dashed line: 1. See Appendix C.2.

Analysis of Graded Supervision. We isolate objective behavior under uniform prompt sampling within each backbone, denoting the tie-aware variant by T-SPLO (uniform). Figure 3 analyzes initial signals on 620 usable Qwen lists; Figure 4 tracks training-pool outcomes and regenerated candidates.

T-SPLO's objective gives positive reward-space signals to 43% of 2,450 partially compliant responses at initialization, versus 32% for all-pairs DPO and 24% for best–worst DPO (Figure 3b). Binary PL and RFT give none. The mean normalized middle-tier signal is $+0.08$ for T-SPLO and -0.18 for binary PL (Figure 3a). Thus, graded supervision assigns upward initial signals to more partially compliant candidates.

On the 617 lists containing ties, uniform random within-tier ordering preserves the expected unnormalized initial tie-aware signal but adds noise with mean noise-to-signal power ratio 1.07 and median 0.54; noise exceeds signal on 25.1% of these lists (Figure 3c). Exact marginalization removes this variation while retaining the expected initial signal (Appendix C.2).

After one training round, T-SPLO (uniform) improves greedy pass rates on 4,096 training prompts by 4.7 points on Qwen and 8.1 on Llama (Figure 4a,b). Pass rates exceed those of DPO by 2.19 percentage points on Qwen and 3.49 on Llama; the corresponding margins over SimPO are 2.97 and 0.98 points.

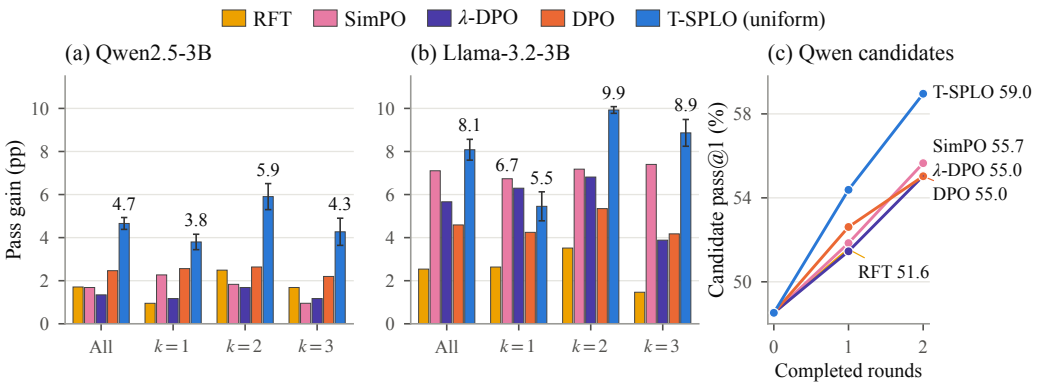


Figure 4: Uniform-sampling diagnostics. (a, b) One-round gains on 4,096 training prompts by constraint count k. Mean $\pm$ s.d.: ten Qwen/two Llama T-SPLO runs; baselines: one run. (c) Candidate pass rates (1,024 prompts/stage); 0: starting model. RFT's two-round sample is unavailable.

Two-constraint gains reach 5.9 points on Qwen and 9.9 on Llama. For single-constraint Llama prompts, however, T-SPLO gains 5.5 points versus SimPO's 6.7 and λ-DPO's 6.3. Marginalization gains 2.15 training-pool points over random tie-breaking on Qwen, but controlled test results show no consistent advantage over all-pairs DPO or binary PL (Table 4).

Qwen candidate pass rates under uniform sampling rise from 48.5% initially to 54.4% after one round and 59.0% after two (Figure 4c). After two rounds, T-SPLO exceeds SimPO by 3.3 points and DPO and λ-DPO by 3.9 each. These regenerated training samples share prompts across methods within each stage, with resampling between stages (Appendix A).

5 Conclusion

We present T-SPLO, which learns from graded verifier feedback through tie-aware listwise optimization and tier-gap prompt selection. Exact marginalization preserves partial-success comparisons without arbitrary within-tier rankings, while self-play refreshes candidates as the policy improves. Experiments show that T-SPLO achieves higher final average instruction-following performance than the compared baselines across the evaluated model families, while largely preserving aggregate general-domain performance. These findings support ordered tiers as an effective supervision structure for learning from partial successes and ties in self-play.

Limitations. Extending T-SPLO beyond the evaluated setting requires addressing two main limitations. **(1) Domain-specific feedback.** Evaluation covers verifiable multi-constraint instruction following with two 3B model families. Constraint counts weight checks equally and omit other aspects of response quality. Equally scored responses may satisfy different constraint subsets, so verifier ties need not imply equal quality in human judgments. **(2) Scaling candidate pools.** Larger model and rollout scales remain unexplored. Exact marginalization is exponential in tied-tier size; subset dynamic programming is practical for our lists of at most nine responses, but larger pools require additional computational strategies.

Future Directions. These limitations motivate two directions for extending T-SPLO to broader tasks and larger scales. **(1) Broader graded evaluation.** Combining task-specific checks with learned evaluators could extend tier construction to code, reasoning, and open-ended generation, accounting for constraint importance and evaluator disagreement. Candidate quality differences could also inform prompt selection, complementing the current greedy-response constraint deficits. **(2) Scalable tie-aware optimization.** Approximations preserving within-tier permutation invariance could support larger candidate pools while retaining cross-tier comparisons. Evaluation with larger models, longer training horizons, and more diverse constraints could clarify when tiered supervision improves learning and how to balance candidate diversity, feedback quality, and training cost.

AI Use Statement. In this work, we used generative AI tools to assist with method design, data cleaning, experiment scripting, and summarization of experimental results. We also used language models to generate synthetic training responses within the proposed self-play procedure: at each round, the current policy generates candidate responses to selected prompts, and programmatic constraint verifiers score them to construct the ordered tiers used for optimization (Section 3). The generation protocol and model backbones are described in Appendices A.1 and A.2. Additionally, we used generative AI tools to retrieve and summarize related literature, draft and revise portions of the manuscript, polish language, reconstruct figures, and edit and format LaTeX. Generative AI tools also assisted with drafting and checking the mathematical derivations. The authors have checked the references cited in the manuscript and take responsibility for the final content of this work, including text, claims, code, figures, and other artifacts produced with AI assistance.

Ethics Statement. This work studies verifiable instruction following using publicly available language models and benchmarks, with training feedback supplied by programmatic verifiers. The resulting models may inherit biases, harmful content, and privacy risks from the underlying models and data. Improved instruction following could also be misused to execute harmful requests more reliably. Our verifiers measure satisfaction of specified constraints; they do not establish the factual accuracy, fairness, or safety of a response. Accordingly, the reported benchmark gains should not be interpreted as evidence of improved safety alignment. Downstream applications should include safety evaluation and safeguards appropriate to their intended use, and use of the models and datasets should respect their respective licenses and usage conditions.

Reproducibility Statement. To support reproducibility, Section 3 specifies the verifier-based tier construction, tie-aware objective, dynamic-programming recurrence, and iterative self-play procedure with tier-gap prompt selection. Appendices A.1 and A.2 describe the implementation setup, candidate-generation protocol, training budget, evaluation metrics, and model backbones. Section 4 reports the main comparisons and ablations, with supplementary controlled component comparisons in Appendix B. Appendix C proves the exact dynamic program, clarifies the invariance and boundary cases, and derives the initial signals and tie-breaking noise used in Section 4.3. Code is available in our anonymous repository: `https://anonymous.4open.science/r/T-SPLO-956D/`.

REFERENCES

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

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

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. In *International Conference on Learning Representations*, volume 2025, pp. 39286–39313, 2025a.

Guanting Dong, Xiaoshuai Song, Yutao Zhu, Runqi Qiao, Zhicheng Dou, and Ji-Rong Wen. Toward verifiable instruction-following alignment for retrieval augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23796–23804, 2025b.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, et al. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*, 2024.

Zhengkang Guo, Wenhao Liu, Mingchen Xie, Jingwen Xu, Zisu Huang, Muzhao Tian, Jianhan Xu, Yuanzhe Shen, Qi Qian, Muling Wu, et al. Recast: Expanding the boundaries of llms' complex instruction following with multi-constraint data. In *International Conference on Learning Representations*, volume 2026, pp. 64656–64693, 2026.

Yun He, Di Jin, Chaoqi Wang, Chloe Bi, Karishma Mandyam, Hejia Zhang, Chen Zhu, Ning Li, Tengyu Xu, Hongjiang Lv, et al. Multi-if: Benchmarking llms on multi-turn and multilingual instructions following. *arXiv preprint arXiv:2410.15553*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Dongyue Li, Ziniu Zhang, Lu Wang, and Hongyang R Zhang. Learning mixtures of plackett-luce models for multi-objective alignment. *arXiv preprint arXiv:2608.25200*, 2026.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. In *International conference on learning representations*, volume 2024, pp. 54605–54624, 2024.

Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, et al. Lipo: Listwise preference optimization through learning-to-rank. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2404–2420, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Jiaqi Ma, Xinyang Yi, Weijing Tang, Zhe Zhao, Lichan Hong, Ed Chi, and Qiaozhu Mei. Learning-to-rank with partitioned preference: Fast estimation for the plackett-luce model. In *International Conference on Artificial Intelligence and Statistics*, pp. 928–936. PMLR, 2021.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

William Muldrew, Peter Hayes, Mingtian Zhang, and David Barber. Active preference learning for large language models. *arXiv preprint arXiv:2402.08114*, 2024.

Pulkit Pattnaik, Rishabh Maheshwary, Kelechi Ogueji, Vikas Yadav, and Sathwik Tejaswi Madhusudhan. Enhancing alignment using curriculum learning & ranked preferences. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 12891–12907, 2024.

Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hanna Hajishirzi. Generalizing verifiable instruction following. *Advances in Neural Information Processing Systems*, 38, 2026.

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.

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.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18990–18998, 2024.

Yuhui Sun, Xiyao Wang, Zixi Li, YiTian Ding, Tianyang Ling, Jialuo Chen, Tianyi Yu, Zhenlong Yuan, and Jinman Zhao. Listwise direct preference optimization with multi-dimensional preference mixing. *arXiv preprint arXiv:2506.19780*, 2025.

Yue Wu, Zhiqing Sun, Rina Hughes, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. In *International Conference on Learning Representations*, volume 2025, pp. 91558–91582, 2025.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. *arXiv preprint arXiv:2312.11456*, 2023.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

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.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, Karthik Ganesan, Wei-Lin Chiang, Jian Zhang, and Jiantao Jiao. Starling-7b: Improving helpfulness and harmlessness with rlaif. In *First Conference on Language Modeling*, 2024.

A Implementation and Evaluation Details

A.1 Data and Training Protocol

Data. We use English single-turn questions from `berkeley-nest/Nectar`, filtered to 20–400 tokens. We remove normalized duplicates and questions sharing a 13-word span with another source question or an evaluation prompt, then append "Please follow these additional instructions:" and one to three compatible IFEval instructions. Of the 25 instruction types, 20 are used for training and five for unseen-type development. Disjoint source-question splits contain 4,096 training, 256 development (128 seen-type and 128 unseen-type), 128 audit, and 512 model-screening prompts. The training pool contains 1,366/1,365/1,365 prompts with one/two/three constraints. Official instruction templates may overlap with IFEval test text, so IFEval measures transfer to new content with familiar templates.

Candidates and verification. For T-SPLO and the non-GRPO baselines, vLLM generates eight stochastic responses and one greedy response per selected prompt using the backbone's chat template. Sampling uses temperature 0.8, top-p 0.95, no top-k filtering, repetition penalty 1.0, and at most 1,024 new tokens. We remove incomplete, degenerate repetitive, and duplicate responses; training postscript checks additionally require nonempty content. After grouping by score and shuffling within tiers, we reserve up to three responses from each extreme tier, fill intermediate tiers from highest to lowest, and fill remaining slots from the extremes, retaining at most nine responses. Single-tier lists are discarded. An AI-assisted audit of 100 responses found agreement with released checker semantics; this was not independent human validation. Official test checkers are unchanged.

Training configuration. All main runs use seed 1 and three training rounds, each selecting 1,024 prompts and taking 256 optimization steps. Round indices 0–2 report checkpoints after one–three completed rounds. The non-GRPO methods use eight retained lists per step; GRPO uses eight prompt groups. We train attention-projection LoRA adapters in BF16 with rank 16, scaling 32, and zero dropout. AdamW uses learning rate 10^{-5}, moments $(0.9, 0.95)$, zero weight decay, 10% warm-up, cosine decay, and gradient clipping at 1.0. Training sequences include response EOS tokens; sequences exceeding 2,048 tokens raise an error. Tie-aware PL, DPO, λ-DPO, and SPPO use $\beta = 0.1$. T-SPLO uses tier-gap sampling without archived-response replay; baselines sample uniformly. T-SPLO and the non-GRPO baselines share the initial candidate pool, then regenerate candidates from their own round-start policies. GRPO refreshes candidates within each round as described below. The main comparison evaluates complete training procedures; the shared-list diagnostic isolates objectives.

Baselines. RFT averages length-normalized log-likelihoods of fully compliant responses. DPO and SimPO sample one response from each extreme tier; SimPO uses length-normalized scores, scale 2.5, and margin-to-scale ratio 0.5. The single-dimension version of λ-DPO ($m = 1$, one verifier dimension) matches length-normalized implicit rewards to a softmax of verifier counts at temperature 1. SPPO regresses log-probability ratios toward centered leave-one-out win rates scaled by $1/\beta$, counting ties as half a win. All-pairs DPO averages over strict cross-tier pairs.

GRPO (Shao et al., 2024) standardizes s/k rewards within each prompt's eight stochastic rollouts, assigns zero reward to incomplete or degenerate responses, and skips equal-reward groups. It uses token-level ratio clipping at 0.2 and a k3 KL penalty toward the initial backbone with coefficient 0.04, both untuned. Each round comprises four phases of 256 prompts and 64 steps each. The first phase of round 0 reuses the shared starting-policy samples; every later phase generates fresh rollouts from its phase-start policy. Optimizer state and the cosine schedule persist across phases within a round. GRPO matches the other methods' step budget, stochastic rollouts per prompt, learning rate, and LoRA configuration.

Component study. Table 3 uses the Qwen backbone and the training configuration above, without per-variant learning-rate tuning. Variants share the initial candidate pool, then regenerate candidates and select prompts with their own policies and sampling rules. The fixed-candidate variant instead reuses round-0 lists and selected prompts and keeps the initial reference policy. Random tie-breaking applies strict PL to randomly ordered tied responses; one response per tier applies strict PL after

retaining one random member of each tier. Binary feedback collapses scores to pass/fail labels and can reduce the number of usable lists. All variants use the main-table vLLM evaluation protocol.

Shared-list diagnostic. Figure 4a,b fixes round-0 candidates and uniform prompt sampling within each backbone, then measures greedy accuracy after one training round on all 4,096 training prompts. T-SPLO (uniform) uses ten Qwen and two Llama runs with identical initial lists; other objectives use one run. Requiring two score levels for all objectives restricts Qwen RFT to 483 of 620 usable lists and excludes 265 sampled prompts with only fully compliant candidates.

Figure 4c reports seed-1 stochastic candidate pass rates, averaging eight responses per prompt over 1,024 prompts at each generation stage. All plotted methods sample prompts uniformly, including T-SPLO (uniform), which omits tier-gap selection. Prompt sets are shared across methods within a stage and resampled between stages. The horizontal axis counts completed training rounds: 0 is the starting model. This count is one greater than the round index in Table 1; T-SPLO (uniform) remains a separate variant. All available Qwen candidate checkpoints are shown, including two completed rounds for λ-DPO; the RFT two-round candidate sample is absent from this analysis snapshot.

A.2 Benchmarks and Models

We evaluate Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct with greedy decoding and released checkers. IFEval has 541 prompts (1,280-token generation limit); IFBench has 300 prompts with held-out instruction types (2,048 tokens); Multi-IF has 909 English conversations and 2,714 evaluated turns (2,048 tokens per turn), with the model's earlier answers included in the history. Multi-IF turn-strict accuracy (T-strict) averages strict pass indicators over turns, rather than requiring every turn of a conversation to pass. Prompt-strict (P-strict) requires all constraints to pass; instruction-strict averages individual checks; prompt-loose (P-loose) uses released relaxed checks. Instruction-following Overall weights the three strict benchmark scores equally, while general-domain Overall weights the four categories in Table 2 equally. Main-table scores are single-seed point estimates.

B Supplementary Controlled Results

Table 4 compares objective variants under uniform sampling. Pool scores use shared round-0 lists; test differences follow two training rounds with candidate regeneration, unlike the three-round tier-gap study in Table 3. All-pairs DPO and binary PL match or exceed tie-aware PL on the training pool. On IFEval, tie-aware PL scores higher than all three alternatives. On IFBench, it scores lower, with all-pairs DPO leading by 5.67 points. These benchmark-dependent outcomes differ from the three-round end-to-end ablation.

Table 4: Controlled objective variants on Qwen2.5-3B. Pool accuracy (%) follows one training round on shared lists. Test differences (tie-aware minus row, percentage points) follow two training rounds with candidate regeneration.

	Pool accuracy	Test difference (pp)	
Objective	(4,096 prompts)	IFEval	IFBench
Starting model	48.22	–	–
All-pairs DPO	52.93	+0.18	−5.67
Binary PL	53.34	+0.74	−1.33
Random tie-breaking	50.73	+2.59	−0.67
Tie-aware PL	52.88	–	–

Uniform prompt sampling throughout; tie-aware PL denotes T-SPLO (uniform). The tie-aware pool score averages ten runs with identical initial lists; other pool scores and test comparisons use one run. Positive test differences favor tie-aware PL; they are not differences from the pool column.

C Mathematical Details

We consider a fixed, finite list with weights $w_i = \exp(r_i) > 0$, using the notation of Section 3.2. Conditioning on the fixed prompt and candidate list is suppressed.

C.1 Exactness of the Tier Dynamic Program

After higher tiers are exhausted, the next PL draw selects $y_i \in S \subseteq T_m$ with probability $w_i/(B_m + \sum_{y_j \in S} w_j)$. Selecting a lower-tier response violates the tier order; selecting y_i leaves completion probability $F_m(S \setminus \{y_i\})$. Summing these disjoint events gives Equation 6; induction from $F_m(\varnothing) = 1$ proves exactness. Once a tier is exhausted, the remaining PL distribution is independent of its internal selection order, so

$$\sum_{\sigma \in \Omega(T)} \mathrm{PL}_\theta(\sigma) = \prod_{m=2}^{M} F_m(T_m). \tag{9}$$

The lowest tier contributes one. Taking negative logarithms gives the method's loss decomposition. A tier of size n_m has 2^{n_m} subsets and $n_m 2^{n_m - 1}$ transitions, yielding $O(\sum_{m=2}^{M} n_m 2^{n_m})$ time with precomputed subset sums and $O(2^{n_m})$ DP storage per tier. Our lists contain at most nine responses.

Invariance and boundary cases. Relabeling candidates within a tier bijectively maps admissible orders and preserves their total probability, proving within-tier permutation invariance. This is symmetry under relabeling, not independence from the rewards of tied responses. Singleton tiers admit one order and recover strict PL. A single tier admits all orders, giving zero loss and gradient; normalization by $M - 1$ applies only to retained lists with $M \geq 2$.

Additional graded supervision. Lists with mixed pass/fail labels also have distinct verifier counts. Graded feedback additionally retains lists with distinct counts but no fully compliant response: 137 lists in the shared Qwen pool. Binary PL collapses each such list to one tier with zero loss, whereas graded PL retains cross-tier supervision.

C.2 Initial Gradients and Tie-Breaking Noise

For an unnormalized PL loss $\mathcal{L}$, define the reward-space ascent signal $g_i = -\partial \mathcal{L}/\partial r_i$. For a strict order of G responses with y_i at position p (best first), differentiation at $r \equiv 0$ gives

$$g_i = f_G(p) = 1 - \sum_{j=1}^{p} \frac{1}{G - j + 1} = 1 - (H_G - H_{G-p}), \tag{10}$$

where $H_0 = 0$ and H_j is the j-th harmonic number. Differentiating the marginal likelihood gives

$$g_{\mathrm{tie}} = \sum_{\sigma \in \Omega(T)} \omega_\theta(\sigma) g_{\mathrm{str}}(\sigma), \qquad \omega_\theta(\sigma) = \frac{\mathrm{PL}_\theta(\sigma)}{\sum_{\sigma' \in \Omega(T)} \mathrm{PL}_\theta(\sigma')}. \tag{11}$$

At initialization, admissible orders are equiprobable, so the signal for a member of T_m is $\bar{f}_m$, the average over its tier's position block $\mathcal{P}_m$. Uniform tie-breaking gives $g_i = \bar{f}_m + \epsilon_i$, with $\mathbb{E}\epsilon_i = 0$ and

$$\mathbb{E}\|\epsilon\|_2^2 = \sum_m n_m \operatorname{Var}_{p \sim \mathrm{Unif}(\mathcal{P}_m)}[f_G(p)]. \tag{12}$$

Because f_G strictly decreases, the variance is positive whenever a tier has multiple responses. A common rescaling of signal and noise leaves their power ratio unchanged. Training divides tie-aware loss by $M - 1$ and strict-order loss by $G - 1$; the equality above concerns unnormalized signals. It holds when policy and reference coincide at a round's start; later, the posterior weights need not be uniform.

Signal diagnostics. For a fixed prompt, write $s_i = s(x, y_i)$ and $k = k(x)$. Figure 3a normalizes each list's signal to unit Euclidean norm and averages coordinates by verifier-tier role. Panel (b) counts positive signals among partial successes ($0 < s_i < k$): 2,450 responses across 620 lists for tie-aware PL, both DPO controls, and λ-DPO, and 1,500 across 483 lists for binary PL and RFT. Neither binary PL nor RFT provides positive partial-success signals, including on excluded lists. Panel (c) computes $\mathbb{E}\|\epsilon\|_2^2/\|g_{\mathrm{tie}}\|_2^2$ analytically over all within-tier orders. It displays the 617 tied lists; the three untied lists have zero noise and are omitted from the logarithmic axis. The mean ratio is 1.07 among tied lists (1.06 across all 620); 155/617, or 25.1%, exceed one.

The other initial signals are: expected best–worst DPO, $1/(2n_M)$ on top-tier and $-1/(2n_1)$ on bottom-tier responses; all-pairs DPO, $(L_i - U_i)/(2P)$, where L_i, U_i count lower- and higher-tier responses and P counts strict pairs; binary PL, the block-average signal after pass/fail collapse; λ-DPO, $\mathrm{softmax}(s_1, \ldots, s_G)_i - 1/G$ in length-normalized reward coordinates; and RFT, $1/n_+$ on each fully compliant response and zero elsewhere in length-normalized log-probability coordinates, where $n_+ = |\{i : s_i = k\}|$. Positive overall scales cancel under per-list normalization. These diagnostics compare objective-level signals, not neural-parameter gradient magnitudes.