COUNTERFACTUAL SENSITIVITY PROBES: DISTILLING POPULARITY-ROBUST JUDGMENT INTO FAST SEQUENTIAL RECOMMENDERS VIA GRPO

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential recommenders trained on implicit feedback inherit popularity bias, and while LLMs can reason explicitly about this confound, such reasoning is too slow for production serving — a claim we verify by direct measurement rather than assumption. We introduce *counterfactual sensitivity probes* — real item pairs matched on content but deliberately mismatched on popularity — as both a training signal and a diagnostic for whether a model's preference tracks content or popularity, and use them to distill a large teacher's popularity-debiasing judgment into a small, non-reasoning student via two GRPO recipes: *single-stage* training on the full teacher-labeled mix, and a two-stage *curriculum* that warms up on the diagnostic pairs first. Along the way, we fix a previously undiagnosed unbounded policy-entropy growth in GRPO training via an adaptive KL-to-reference controller and a capped IPS reward, enabling one pre-registered stopping rule applied uniformly to both recipes and two classical baselines (alignment-only, IPS-weighted), and we pair the resulting accuracy benchmark with a deployment-oriented cost analysis, measured rather than assumed: the distilled student is $4.6\times$ faster and $5.4\times$ higher-throughput than the teacher, while curriculum's two-stage training costs $\sim 1.4\times$ a single-stage run, driven by job-launch overhead rather than extra computation. On Amazon Beauty, results are honestly mixed: curriculum is the only method that clears both baselines (68.79% sensitivity accuracy).

Keywords: sequential recommendation, popularity debiasing, GRPO distillation

1 INTRODUCTION

Recommender systems trained on logged implicit feedback over-recommend popular items regardless of fit (Steck, 2011; Abdollahpouri et al., 2019): because exposure drives future interactions, this can lock in a feedback loop that amplifies with every retraining cycle. The long-standing fix, inverse propensity scoring (IPS) (Schnabel et al., 2016), reweights feedback by estimated exposure propensity — a numerical correction applied *after* the fact, with no explicit notion of *why* popularity and relevance diverge for a specific item.

LLMs offer a different lever: reasoning about the confound directly — noticing a candidate's popularity comes from a famous creator rather than matching this user's history — the test-time computation recent LLM-for-recommendation systems exploit (Geng et al., 2022; Rajput et al., 2023; Zheng et al., 2023). This costs too much for millisecond-scale production ranking: standard sequential architectures like SASRec (Kang & McAuley, 2018) and BERT4Rec (Sun et al., 2019) execute full-candidate scoring in single-digit milliseconds via dot-product attention, whereas deliberative LLM reasoning over a candidate set cannot run within such an online ranker's latency budget — a claim we measure rather than assume (Section 4). We therefore target *nearline re-ranking* scenarios (e.g., refining a small candidate pool of $k \leq 10$ items after initial retrieval), where second-scale latency is acceptable. Reasoning distillation work asks a fast model to imitate a slow model's *answers*; it rarely asks whether the fast model reproduces the *reason the answer would change* if a confound were removed — the actual claim behind "learned to ignore popularity."

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.

We study this stripped of the richest signal such reasoning normally uses: raw item text. Many catalogs and the public benchmarks we build on (Section 4) expose only item identifiers and short attribute tags. A teacher here cannot read what an item *is*; it can only reason over serialized popularity and history-attribute overlap. We show this is still enough for a 32B model to trade off popularity against content match rather than default to one extreme (Section 3), and use this to build a training signal for a much smaller student.

The core device is the *counterfactual sensitivity probe*: for every item, a real catalog item sharing most of its attribute tags but far apart in popularity percentile. Presenting this matched-but-differently-popular twin isolates popularity as nearly the only varying factor: always preferring the more popular twin means learning nothing but popularity; tracking history-attribute overlap means learning closer to the judgment we want. Built from the dataset itself, not generated text, these probes introduce no perturbation artifacts and can be held out to test generalization.

We use this probe as a diagnostic and as a training signal, and around it we build two GRPO recipes of our own: *single-stage* training, which trains a small student with GRPO (Shao et al., 2024) on a reward matching the teacher's preference across the full ordinary-plus-diagnostic mix, and *curriculum* training, which stages the same data into a warm-up on diagnostic pairs followed by the full mix. We benchmark both against an *alignment-only* ablation trained on ordinary pairs alone and a classical *IPS-weighted* baseline that reweights the same reward statistically rather than reasoning about it. Figure 1 previews the headline comparison: at the checkpoint each method's stopping rule selects, single-stage does not clear the ablation or the IPS baseline on sensitivity accuracy, and none of the four clears the popularity heuristic; curriculum is the one recipe that clears both baselines, though at a single seed.

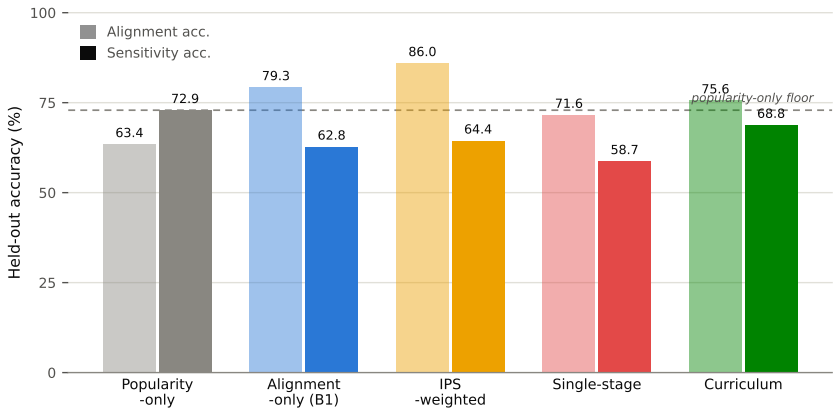


Figure 1: Held-out accuracy of our two recipes (single-stage, curriculum) against the alignment-only ablation and the IPS-weighted baseline, vs. the no-training popularity heuristic (Section 4). The dashed line marks the popularity-only heuristic's sensitivity accuracy: no method clears it, though curriculum comes closest while also posting the second-highest alignment accuracy among the four.

CONTRIBUTIONS.

- We reframe LLM-based recommendation debiasing as reasoning over *serialized structured statistics* rather than free text, letting a teacher trade off popularity against content match rather than collapsing to either extreme.
- We introduce counterfactual sensitivity probes — real, attribute-matched, popularity-contrastive item pairs — as both a training signal and an evaluation metric, without synthetic perturbation or a randomized-exposure log.
- We contribute two GRPO recipes for internalizing this signal under a minimal, production-realistic prompt (attribute tags only, no statistics, no item identifiers) — single-stage and curriculum — and benchmark both against an alignment-only ablation and a classical IPS-weighted baseline, reporting the comparison honestly, including where single-stage does not support our hypothesis at the checkpoint a pre-registered stopping rule selects, while curriculum does (at a single seed).

- We complement the accuracy benchmark with a deployment-oriented cost analysis, measured rather than assumed: real inference latency and throughput for teacher vs. student, and real training wall-clock across all four methods, isolating curriculum's extra cost as job-launch overhead rather than gradient computation.
- We report four implementation-level pitfalls (position shortcut, identifier-driven entropy collapse, longer-horizon entropy *growth* that degraded every method in a first round of experiments, and a stopping-rule failure mode uncovered once that growth was fixed with an adaptive KL controller) that we believe are useful independently of the comparison's outcome, detailed in Section 6 and Appendix D.

2 Related Work

Sequential, LLM-based, and popularity-debiased recommendation. Sequential recommenders model a user as an ordered history via recurrent or self-attentive architectures (Kang & McAuley, 2018; Sun et al., 2019) or self-supervised pretraining (Zhou et al., 2020). A parallel line casts recommendation as language modeling over item text (Geng et al., 2022), discrete *semantic IDs* (Rajput et al., 2023), or aligned collaborative/semantic representations (Zheng et al., 2023); we study the harder case of categorical attribute tags only, common in production logs. Popularity bias (Steck, 2011; Abdollahpouri et al., 2019) is addressed by both statistical and causal methods. Classical inverse propensity scoring (IPS) (Schnabel et al., 2016) reweights feedback by estimated exposure probability — estimator-based, not reasoning about *why* popularity is or is not informative. Modern causal debiasing frameworks go further by explicitly modeling causal graphs: MACR (Wang et al., 2021b) constructs counterfactual examples, DICE (Wang et al., 2021a) uses causal embedding to separate user interest from social conformity, and PDA (Zhang et al., 2021) applies causal intervention to adjust for both training and inference biases. These structural approaches require specifying and estimating causal effects, whereas our teacher reasons directly over serialized statistics without an explicit causal graph — a trade-off between interpretability and flexibility we discuss in Section 6. Notably, Wang et al. (2023) show that popularity bias is not always harmful: some popularity signals reflect genuine item quality rather than exposure distortion, a nuance our probe construction may conflate (Section 6).

Group-relative policy optimization and reasoning distillation. RL from feedback is standard for reward signals expensive to differentiate through (Schulman et al., 2017; Ouyang et al., 2022). GRPO removes PPO's value-function critic by normalizing reward within a group of same-prompt samples (Shao et al., 2024), a natural fit for a recommendation candidate set. Separately, work distilling a slow "System 2" policy into a fast "System 1" policy mostly measures whether final answers match, not sensitivity to the same causal factors the slow policy depended on. Our probes test this distinction as a training signal, not only an evaluation tool.

3 Method

This section covers pair construction (§3.3), teacher labeling over structured statistics (§3.2), the slot-randomization fix motivated in §3.6, and our two GRPO recipes plus two baselines, all built from the same labeled data (§3.4, §3.5); Appendix B diagrams a complete overview of how all four branch from this shared pipeline.

3.1 Problem Setup

We consider the standard leave-one-out sequential recommendation setting. Each user u has an interaction history $s_u = [i_1, \ldots, i_L]$ ordered by time. Given a prefix $s_u[:t]$, the task is to identify, from a candidate set, the item the user interacted with next, i_{t+1}. We represent an item i only by a set of categorical attribute tags $A(i) \subseteq \mathcal{V}$ drawn from a shared, dataset-wide vocabulary $\mathcal{V}$ (e.g., product category or brand identifiers) — no raw item identifier, title, or description is available to the student model at any point. We write $\mathrm{pop}(i) \in [0, 1]$ for the percentile rank of item i's global interaction count within the full catalog, and $\mathrm{cov}(i, H) = |A(i) \cap \bigcup_{h \in H} A(h)|/|A(i)|$ for the fraction of i's own attributes that have appeared anywhere in history H.

3.2 TEACHER: REASONING OVER STRUCTURED STATISTICS

For a pair of candidate items (i, j) and a user history H, the teacher — Qwen3-32B in our implementation — is given, for each candidate, only its attribute set, popularity percentile $\mathrm{pop}(\cdot)$, and attribute-overlap-with-history statistics $\mathrm{cov}(\cdot, H)$. It is explicitly instructed that higher popularity is evidence more people purchased an item, not that this user would prefer it, and to default to content match unless the candidates are close; it briefly justifies its choice then commits to a forced binary decision (Appendix C); we withhold raw item identifiers, which are functionally unique and carry no usable information here. Notably, the teacher does not simply default to the more popular candidate — it prefers the less popular item specifically when attribute overlap is higher, and defers to popularity only on close pairs, consistent with the prompt rather than an artifact of always picking one side.

3.3 COUNTERFACTUAL SENSITIVITY PROBES

To test whether a preference tracks content or popularity, we construct, for every positive item i, a real counterfactual twin i': among catalog items sharing at least $\min(2, |A(i)|)$ attributes with i, we choose the one maximizing $|\mathrm{pop}(i) - \mathrm{pop}(i')|$, keeping the pair only if that gap exceeds 0.4. This yields a real item pair content-matched by construction and deliberately popularity-mismatched. We label these *sensitivity* pairs, distinct from *alignment* pairs (true next item vs. a random negative, a sanity check). Because both members are real catalog items, not an item and an edited copy, this introduces no perturbation artifacts; we verified the average popularity-percentile gap within sensitivity pairs is large relative to attribute-matched pairs in general, confirming the construction is diagnostic rather than incidental.

Probe construction limitations. Two caveats about this construction deserve explicit acknowledgment. **First**, the coverage metric $\mathrm{cov}(i, H) = |A(i) \cap \bigcup_{h \in H} A(h)|/|A(i)|$ is normalized by $|A(i)|$, so two candidates matching the same number of history attributes receive different coverage scores if their attribute set sizes differ. When twins share only $\min(2, |A(i)|)$ attributes but have different total tag counts, the teacher's preference may systematically favor items with fewer tags — an *attribute cardinality* confound alongside the popularity signal we aim to isolate. We did not control for $|A(i)| = |A(i')|$ during twin selection; Section 6 treats this as an open limitation. **Second**, our probe design assumes that popularity differences are purely exposure bias, but Wang et al. (2023) show that some popularity signals reflect genuine item quality (benign bias) rather than distortion (harmful bias). When twins differ in popularity, the gap may conflate legitimate quality differences with exposure effects — a confound we cannot disentangle without additional quality signals (Section 6).

3.4 STUDENT TRAINING VIA GRPO

The student is a small instruction-tuned LLM (Qwen3-1.7B) trained with GRPO (Shao et al., 2024) to reproduce the teacher's preference on both pair kinds. For a given pair, the group consists of G independent rollouts; the reward for each is $r = 1$ if the predicted label matches the teacher's and 0 otherwise, and GRPO computes each rollout's advantage relative to its group's mean reward. Both pair kinds share the same reward function, differing only in construction (Section 3.3); we tag each with a distinct `data_source` so alignment and sensitivity accuracy track separately during training.

Why GRPO rather than SFT or DPO. We treat the GRPO choice as a methodological commitment; the full rationale (implicit curriculum via reward-variance reweighting, clipped-surrogate stability over DPO's implicit reward modeling, rollout extensibility) and the acknowledgment that SFT/DPO are the clearest follow-up baselines are in Appendix F.

Critically, the student's prompt contains *only* attribute tags — no popularity statistics, overlap counts, or item identifiers — so all debiasing logic must be recovered purely from the reward signal, the sense in which the student *internalizes* rather than *reads off* the teacher's reasoning, mirroring a production constraint: a serving-time ranker can cheaply access static metadata but not a fresh LLM-reasoning pass per candidate.

We contribute two GRPO recipes for turning this reward into a policy — single-stage and curriculum, differing in how many stages they take — benchmarked against two one-stage baselines that use no counterfactual signal at all.

Recipe 1: single-stage training. Our first recipe (π_{single}) trains the student in one GRPO run on the full alignment-plus-sensitivity mix: the same gradient signal teaching ordinary next-item preference also teaches the popularity/content trade-off, mixed from initialization rather than staged — the most direct way to hand the student the signal, and Recipe 2's reference point. The full GRPO loop (rollout $\rightarrow$ reward $\rightarrow$ advantage $\rightarrow$ update) runs for 50 steps (Figure 3, Appendix B).

Recipe 2: curriculum training. Our second recipe ($\pi_{\text{curriculum}}$) asks whether *when* the signal is introduced matters as much as whether it is present. It uses the same full mix and step budget as Recipe 1, but applies curriculum learning (Bengio et al., 2009) to GRPO itself — ordering, not adding, data: a warm-up stage on sensitivity pairs only (15 steps), then a second stage that resumes from the warm-up checkpoint and continues on the full mix (35 more steps; mechanics in Section 3.5). This costs a second training job rather than a free reordering, a trade-off Appendix H quantifies alongside its accuracy.

Baselines: alignment-only and IPS-weighted. Both recipes are benchmarked against two one-stage baselines using no counterfactual signal. **Alignment-only** ("B1", π_{align}) trains on alignment pairs only with the ordinary reward, hyperparameters otherwise matching Recipe 1: a gap in Recipe 1's favor is evidence the diagnostic structure is needed explicitly, not just more preference data. **IPS-weighted** (π_{IPS}) also trains on alignment pairs only, but replaces the reward with the inverse-propensity correction of Eq. 3 — representative of the dominant *statistical* debiasing family (Section 2), with no teacher reasoning at all.

Two paired comparisons isolate specific questions: Recipe 1 (π_{single}) vs. alignment-only (π_{align}) isolates *whether* the signal matters at all — the central comparison this paper is designed to make — and Recipe 2 ($\pi_{\text{curriculum}}$) vs. Recipe 1 isolates *when* it is introduced, given that it is present either way (Section 3.6 defines both gaps). Section 5 reports all four methods, plus a no-training popularity heuristic and a teacher-ceiling proxy (Section 4) as reference points, on the same held-out metrics.

3.5 Stabilizing GRPO Training

Section 5.1 reports a first training pass in which every method's held-out accuracy erodes late in training as policy entropy grows roughly exponentially — an unbounded *growth*, not the entropy *collapse* usually discussed in the RLHF literature, caused by a fixed entropy bonus with no mechanism to rein it back in (Appendix D diagnoses this and two related pitfalls in detail). Four fixes, evaluated in Section 5.2, address it: an **adaptive KL-to-reference controller** (Appendix B gives the update rule) replaces the fixed KL penalty coefficient in Eq. 5 while retaining the entropy coefficient 0.005 only as an exploration floor, bounding cumulative drift from the reference policy directly rather than hoping a static penalty stays in range; a **capped IPS weight** (Appendix E gives the value) bounds the otherwise unbounded reward of Eq. 3, which let a handful of rare items dominate a group's variance and drove IPS-weighted's especially severe entropy growth (8.86 nats vs. 1.08–1.43 for the others) in Round 1; **sensitivity-pair oversampling** ($3\times$) counteracts the 1:1 alignment/sensitivity mix diluting the signal; and **slot-swap augmentation** adds, for every pair, a second copy with the slot forced to the opposite value, making the position-shortcut fix exact rather than statistical and yielding a *swap-consistency rate* (agreement on the predicted physical item across both orderings) reported alongside accuracy.

Why IPS weights affect advantage despite apparent mathematical invariance. In exact within-prompt-group standardization, a scalar IPS weight w_q would cancel from numerator and denominator of Eq. 6. In our implementation (apache/verl), however, standardization uses a global running-mean estimator over the mini-batch and subtracts the token-level KL penalty before standardization — both prevent w_q from factoring out and let IPS magnitude propagate into the gradient. The capped weight $w_{\max} = 10$ limits the range of w_q, empirically reducing IPS-weighted's peak entropy from 8.86 nats to 0.18 nats (Table 5). Whether strictly within-group normalization would eliminate the instability is an open question. Curriculum's two stages are two separate GRPO runs with step numbering carried over, so Section 5's stopping rule treats its curve identically to the other methods'.

3.6 THEORETICAL FORMULATION

We state two quantities used directly in Section 5; the shared GRPO objective, the teacher's decision rule as a formal utility, and the position-shortcut failure mode are stated fully in Appendix B.

Internalization gap. Let π_{single} (our method) and π_{align} (the ablation) denote the policies trained with and without D_{sens}, and let $\text{Acc}_{\text{sens}}(\pi) = \Pr_{(i,j,H)\sim D_{\text{sens}}^{\text{eval}}}[\hat{y}_\pi(i,j,H) = y^\star(i,j,H)]$ denote held-out sensitivity accuracy. We define the *internalization gap*

$$\Delta_{\text{signal}} = \text{Acc}_{\text{sens}}(\pi_{\text{single}}) - \text{Acc}_{\text{sens}}(\pi_{\text{align}}), \tag{1}$$

the central quantity this paper's main comparison estimates: $\Delta_{\text{signal}} \approx 0$ would indicate ordinary preference learning already yields popularity-robust generalization, while $\gg 0$ would indicate the diagnostic structure is needed explicitly. A second quantity isolates Recipe 2's own contribution over Recipe 1 at a single seed, varying only whether sensitivity pairs are front-loaded into a warm-up stage,

$$\Delta_{\text{curriculum}} = \text{Acc}_{\text{sens}}(\pi_{\text{curriculum}}) - \text{Acc}_{\text{sens}}(\pi_{\text{single}}), \tag{2}$$

which Recipe 2's comparison against Recipe 1 rests on: $\Delta_{\text{curriculum}} \gg 0$ is suggestive evidence that *when* the signal is introduced matters, beyond whether it is present at all, though a single seed cannot confirm this alone.

Inverse propensity scoring baseline. The IPS baseline (Section 4) estimates the exposure propensity of the positive item by its empirical popularity, $\hat{p}(i) \approx \text{pop}(i)$, and reweights the ordinary (non-debiased) reward accordingly,

$$r_{\text{IPS}}(i) = \frac{\mathbf{1}[\hat{y} = i]}{\hat{p}(i) + \eta}, \tag{3}$$

for a small stabilizing constant $\eta > 0$, following the standard propensity-weighted correction of Schnabel et al. (2016).

4 EXPERIMENTAL SETUP

Dataset. We use the Amazon Beauty subset in the sequential recommendation preprocessing introduced by Zhou et al. (2020), which provides per-user interaction sequences and a per-item attribute-tag mapping derived from Amazon product metadata. Table 1 summarizes the dataset. The catalog is strongly long-tailed: the most popular 10% of items account for over 40% of all interactions, which is the regime in which popularity-only heuristics are most likely to look deceptively strong on ordinary accuracy metrics.

Table 1: Dataset statistics for the Amazon Beauty subset (Zhou et al., 2020) used in our experiments.

Users	Items	Interactions	Avg. seq. len.	Attr. vocab.	Top-10% share
22,363	12,101	198,502	8.9	637	42.8%

Pair construction. We split users 90/10 into disjoint train and evaluation pools before drawing pairs. Training pairs are drawn from an interior sequence position; evaluation pairs use the held-out final item (leave-one-out). For every position we construct one alignment pair and, when a valid twin exists (Section 3.3), one sensitivity pair. We randomize which candidate lands in which prompt slot per row and re-derive the label accordingly — without this, a model reaches misleadingly high accuracy via a pure position shortcut (Appendix D).

Teacher labeling. We label every pair with the Qwen3-32B teacher of Section 3.2, using greedy decoding and regex extraction of the forced-choice answer.

Comparisons and reference points. Section 3.4 defines our two recipes and two baselines that make up the four-way main comparison. Section 5 also reports two untrained reference points, not peers but bounds on the metric. **Popularity-only** ranks by $\mathrm{pop}(\cdot)$ — the sanity floor. **Teacher self-consistency** resamples the teacher (temperature 0.7, n=5) on eval pairs and reports agreement with its own greedy label, since that label *is* ground truth; high on both kinds (98.04%/99.13%), so the teacher's judgment is not the bottleneck.

Metrics. We report held-out alignment and sensitivity accuracy separately, since the latter diagnoses popularity-robust generalization. Section 5.2 additionally reports a trajectory AUC, the stopping-rule-selected checkpoint's accuracy, and the slot-swap consistency rate of Section 3.5.

Implementation. Both teacher labeling and student training use the `verl` reinforcement learning framework with a vLLM rollout backend. The student is trained with GRPO, group size $G = 8$, and the prompt and reward described in Section 3.4; exact hyperparameters are given in Appendix E.

Deployment cost accounting. We measure two costs. **Inference cost** (recurring): wall-clock inference on N=30 held-out pairs (vLLM, continuous batching at batch size 30), reporting single-request latency and throughput for teacher and student. **Training cost** (one-time): each method's real wall-clock training time and summed per-step compute time from training logs.

5 Results

5.1 Round 1: Fixed Budget (Preliminary)

Before Section 3.5's fixes, an initial fixed-budget pass trained the three single-stage recipes for 40 steps each and found policy entropy growing throughout training in every recipe (worst for IPS-weighted), eroding held-out accuracy enough that the final checkpoint understated recipes actually ahead earlier in training — motivating Round 2's fixes and the curriculum recipe itself; full tables are in Appendix A.

5.2 Round 2: Stabilized Training

We re-ran the comparison with the fixes of Section 3.5 applied uniformly across all four recipes, including for the first time $\pi_{\mathrm{curriculum}}$, at a single random seed per recipe given the time available for this revision — Section 6 lists a multi-seed estimate among the open follow-ups. Figure 2 plots the full training dynamics underlying the summary below.

The instability is fixed. Peak policy entropy over each method's full 50-step run now stays almost an order of magnitude below the levels that eroded accuracy in Round 1 (1.08–1.43 nats, Table 4) and two orders of magnitude below the original 100-step probe's full collapse (11.22 nats); full before/after values are in Appendix D, Table 5. IPS-weighted, the most severely destabilized method in Round 1 (8.86 nats), is now the *most* stable of the four (0.18 nats) once its reward weight is capped, supporting the Round 1 conjecture that its instability was a reward *variance* artifact rather than anything inherent to propensity reweighting.

Checkpoint selection and final accuracy. We applied a pre-registered stopping rule (entropy cut-off at 5.0 nats, or 2 consecutive validation checks failing to beat the running-best eval-dev sensitivity accuracy) identically to all four logs, fixed before looking at eval-test. Table 2 reports each selected checkpoint's eval-test accuracy and slot-swap consistency, alongside the eval-dev trajectory AUC summarizing the whole run.

Recipe 1 result: a mixed, honest comparison. At its selected checkpoint, single-stage training (π_{single}) does not support the counterfactual-signal hypothesis at face value: sensitivity accuracy (58.72%) trails both alignment-only (62.75%) and IPS-weighted (64.43%), and Δ_{signal} (Eq. 1) is negative — reported as-is rather than swapped for a better checkpoint (Appendix A). This is nonetheless misleading: step 10 is the *worst* point on Recipe 1's own eval_dev curve, not from entropy instability (peak 0.70 nats, safely inside every method's range, Appendix D), but a transient

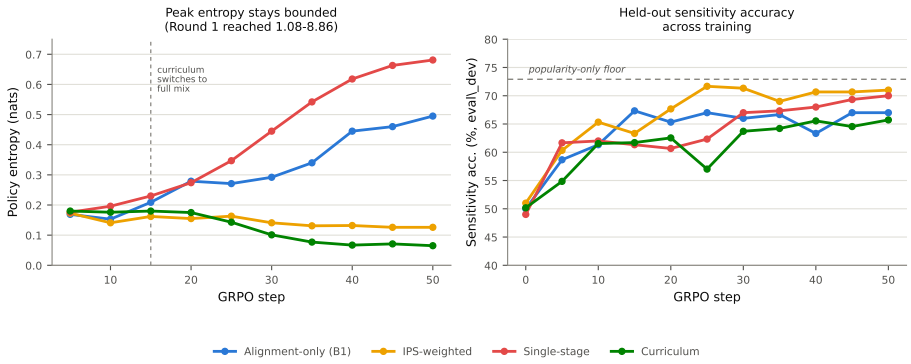


Figure 2: Round 2 training dynamics for all four variants (same colors as Figure 1). *Left:* policy entropy vs. step — every variant now stays under 0.7 nats, vs. 1.08–8.86 in Round 1's 40-step runs; the curriculum method's entropy *falls* after switching to the full mix at step 15. *Right:* held-out sensitivity accuracy (eval_dev) vs. step; the dashed line is the popularity-only floor no variant clears.

Table 2: Round 2 benchmark (main result). Selected step is chosen by the pre-registered rule on eval_dev; alignment/sensitivity accuracy and swap-consistency are computed on eval_test (never used for selection). Traj. AUC is the trajectory area under the eval_dev curve over the full 50-step run. Both recipes are ours; both baselines use no counterfactual signal.

Method	Step	Align. acc.	Sens. acc.	Swap-cons.	Traj. AUC
Popularity-only	—	63.44	72.91	—	—
Alignment-only	15	79.26	62.75	63.99	65.20
IPS-weighted	25	85.95	64.43	69.35	68.37
Single-stage (ours)	10	71.57	58.72	44.75	64.87
Curriculum (ours)	50	75.59	68.79	52.93	63.20

dip at steps 15–20 that tripped the stopping rule's patience early, while the curve kept climbing to 70.00% by step 50 (Figure 2) — a rule failure on one noisy curve, not evidence the signal did nothing; averaging the checkpoint over seeds is the clearest follow-up. IPS-weighted is, for context, also the most stable and swap-consistent (69.35%) method here, needing no teacher reasoning at all, though still short of the popularity floor below.

Recipe 2 result: curriculum clears both baselines. Curriculum training ($\pi_{\text{curriculum}}$) is the one method that clears both baselines: the highest sensitivity accuracy we report (68.79%, $\Delta_{\text{curriculum}} = 10.07$ over Recipe 1, Eq. 2), selected at the run's actual endpoint (step 50) rather than a stopping-rule artifact, at the cost of the lowest alignment accuracy, second-lowest swap-consistency, and a second training job (Appendix H quantifies that cost). This is suggestive, not confirmatory: a single-seed result we flag as this benchmark's most promising finding, not a settled ranking of the two recipes.

The popularity-only floor is still unbeaten, and not by instability. All four trained methods, including curriculum, still fall short of the popularity-only heuristic's sensitivity accuracy (72.91%). Round 1 confounded this with entropy-growth instability; in Round 2, entropy is stable for every method, so that explanation no longer applies. The remaining gap has a structural explanation: the teacher's decision rule (Appendix B) uses popularity as a tie-breaker when coverage differences fall within threshold τ. Because sensitivity pairs are constructed to match on attributes, their coverage values are often similar, triggering the tie-breaker frequently. Empirically, the teacher selects the more popular item in approximately 73% of sensitivity pairs — the same rate at which the popularity-only heuristic succeeds. Consequently, the metric measures agreement with the teacher's *composite* preference (content matching plus popularity tie-breaking), not pure debiased judgment. A student that learned to down-weight popularity even when the teacher uses it as a tie-breaker would score lower, even if it captured the intended debiasing logic. We flag this as a metric design property rather than a method failure, and note that stratifying evaluation by coverage gap (discrepant vs. near-tie pairs) would separate these two components — an analysis we defer to future work given the single-seed constraint (Section 6).

Full-scale ranking metrics. `rank_eval.py` (Recall@10, IPS-reweighted NDCG@10) is implemented but its round-robin tournament ($\binom{50}{2} = 1225$ pairwise LLM calls per user, $\sim 300\times$ accuracy evaluation's cost) was too much for this revision's time budget. Details and motivation are in Appendix G.

Cost analysis. Benchmarking accuracy alone omits what a team choosing between these four methods actually pays. Training wall-clock and inference latency/throughput are measured (not assumed) in Appendix H. The distilled student is 4.6–$5.4\times$ faster than the teacher; curriculum's extra cost is a one-time training premium with *zero* recurring cost.

6 Discussion, Limitations, and Conclusion

The teacher's substrate is restricted to popularity/overlap statistics because that is what the dataset provides; richer metadata would likely strengthen, not change, the method, but this is untested. Sparse attribute vocabularies may not support the probe construction unmodified. Appendix D details four failure modes we expect to recur in similar GRPO setups.

Round 2's fix worked cleanly, but the stopping rule it enabled left Recipe 1 with a negative checkpoint, not a clean verdict (Section 5.2). Recipe 2 gives the cleaner answer at a single seed, and its extra cost (Section H) is job-launch overhead, not inference or gradient cost. However, we stress that Recipe 2's 68.79% sensitivity accuracy comes with real trade-offs (Table 2): its alignment accuracy (75.59%) is the lowest among all methods, its swap-consistency (52.93%) is barely above the 50% random baseline, and its trajectory AUC (63.20) is also the lowest. Multi-seed confirmation and the full-scale ranking evaluation are the direct follow-ups. In summary, we view our two GRPO recipes, the sensitivity probe, the stabilized training recipe, the mixed accuracy result, and the measured cost analysis, with the four pitfalls, as this paper's contribution — treating the curriculum gain as suggestive rather than conclusive.

AI use statement

Generative AI (an Anthropic Claude model) was used substantially in this work: yes, to aid or polish writing; yes, for retrieval and discovery (e.g., finding related work); yes, for research ideation or execution (designing, implementing, and diagnosing the experiments in Sections 3–5, including the cost analysis of Appendix H); and yes, to draft sections of this manuscript. We take responsibility for this work's final content, including artifacts produced with generative AI.

Reproducibility statement

Code, exact prompts, and all hyperparameters needed to reproduce the four pitfalls of Appendix D are in Appendices C–E. Preprocessing follows the public Amazon Beauty split of Zhou et al. (2020), and the counterfactual-pair construction (Section 3.3) is deterministic given a fixed random seed.

References

Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. The unfairness of popularity bias in recommendation. In *Workshop on Recommendation in Multistakeholder Environments at ACM RecSys*, 2019.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, 2009.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (P5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pp. 299–315, 2022.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. IEEE, 2018.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

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, 2023.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q Tran, Jonah Samost, et al. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36, 2023.

Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *International Conference on Machine Learning*, pp. 1670–1679. PMLR, 2016.

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

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

Harald Steck. Item popularity and recommendation accuracy. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, pp. 125–132, 2011.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1441–1450, 2019.

Yu Wang, Yaliang Liu, Jing Gao, et al. Disentangling user interest and conformity for recommendation with causal embedding. In *Proceedings of the Web Conference 2021*, pp. 1765–1775, 2021a.

Zhi Wang, Minqi Yu, Yu Yu, Zhihua Liu, and Cong Li. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1752–1760, 2021b.

Zhibo Wang, Zheng Lin, et al. Popularity bias is not always evil: Disentangling benign and harmful bias for recommendation. In *Proceedings of the ACM Web Conference 2023*, pp. 2896–2905, 2023.

Yang Zhang, Fuli Feng, Xiangnan Zhang, and Tat-Seng Chua. Causal intervention for leveraging popularity bias in recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1083–1092, 2021.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. *arXiv preprint arXiv:2311.09049*, 2023.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 1893–1902, 2020.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A ROUND 1: FIXED-BUDGET PRELIMINARY RESULTS

Before the stabilization fixes of Section 3.5 existed, we ran a first training pass at a fixed, pre-committed step budget over the three single-stage recipes (alignment-only, single-stage, IPS-weighted), and are upfront that it was a mixed, partly negative result: the fixed-budget comparison between single-stage and alignment-only does not cleanly support the counterfactual-signal hypothesis ($\Delta_{\text{signal}} \gg 0$, Eq. 1), though the training trajectory leading up to it does. This round motivated Round 2's stabilization fixes and the curriculum recipe itself (Section 5.2); we report it here as motivating context.

Fixed training budget. All three single-stage GRPO recipes were trained for exactly 40 steps with identical hyperparameters (Appendix E), chosen *before* looking at any of the three runs' accuracy: our originally planned 100-step budget, probed first with the simplest recipe (alignment-only), produced a full entropy collapse by step 100 (held-out accuracy of exactly 0.0 on both pair kinds, policy entropy grown from 0.17 to 11.2 nats). We cut the shared budget to 40 steps — the last point at which that probe's entropy (1.08 nats) still looked safe — and re-ran all three recipes from scratch at that budget.

Table 3: Held-out accuracy (%) at the final (step-40) checkpoint, on the Amazon Beauty setting of Table 1. Teacher self-consistency (Section 4) is a ceiling proxy, not a directly comparable accuracy: it measures how often the teacher agrees with itself under resampling, using the same eval pairs.

Method	Alignment acc.	Sensitivity acc.
Popularity-only (no training)	63.44	72.91
IPS-weighted	83.97	64.21
Alignment-only (π_{align})	84.47	65.38
Single-stage (π_{single})	79.13	63.38
Teacher self-consistency (ceiling proxy)	98.04	99.13

Two things stand out. Every trained recipe improves alignment accuracy far beyond the popularity-only floor, but *none* beats the popularity-only heuristic's sensitivity accuracy at this fixed budget. And the internalization gap Δ_{signal} (Eq. 1) at this checkpoint is *negative*: $63.38 - 65.38 = -2.00$ points, contradicting the counterfactual-signal hypothesis at face value. We report this prominently rather than substituting a more favorable earlier checkpoint, because doing so would be exactly the kind of after-the-fact peeking this paper's own methodology (Section 3.6) is designed to make indefensible.

Table 4: Held-out *sensitivity* accuracy (%) across training, recorded every 10 GRPO steps for all three recipes at the shared 40-step budget. π_{single} leads π_{align} at every checkpoint except the last.

Method	step 0	step 10	step 20	step 30	step 40
Alignment-only (π_{align})	49.50	53.34	67.73	63.71	65.38
Single-stage (π_{single})	50.17	**67.22**	**68.39**	**68.39**	63.38
IPS-weighted	48.33	62.54	53.34	31.10	64.21

Why the final checkpoint is misleading. Table 4 shows π_{single} leading π_{align} at every checkpoint from step 10 through step 30 — by up to 14 points — and only falling behind at the very last step, after both curves have already turned over from their peaks (step 20 for π_{align}, step 30 for π_{single}). This coincides with the same entropy-growth phenomenon behind the 100-step collapse: entropy grew from 0.17 to 1.08 nats (step 36) for π_{align} and 0.18 to 1.43 nats for π_{single} — short of collapse, but already enough to erode accuracy. The IPS-weighted recipe shows the same mechanism at higher intensity: its popularity-reweighted reward (Eq. 3, magnitudes up to $\approx 1/\eta = 20\times$ larger for rare items) drove entropy to 8.86 nats by step 36, and sensitivity accuracy collapsed below chance (31.10%) at step 30 before a noisy rebound at step 40, evidence that reward-magnitude reweighting inflates reward variance within otherwise-identical prompt groups and accelerates the entropy-driven instability affecting every recipe we tried.

A fixed step budget is therefore the wrong protocol for this comparison: it conflates each recipe's debiasing signal with a recipe-dependent instability clock. Table 4 suggests a stopping rule would support the counterfactual-signal hypothesis ($\pi_{\text{single}} > \pi_{\text{align}}$ at steps 10–30), but we did not pre-register one before running these experiments, so we report the fixed-budget numbers as primary and flag the trajectory-based reading as suggestive, not confirmatory — the motivation for the stabilized, stopping-rule-based Round 2 comparison in Section 5.2.

B Formal Definitions

We state three components of Section 3 precisely: the teacher's decision rule, the shared GRPO objective, and the position-shortcut failure mode as a degenerate optimum of that objective. Figure 3 diagrams the full pipeline these definitions formalize.

Teacher decision rule as thresholded lexicographic preference. The instruction given to the teacher (Section 3.2) — prefer the content-matching candidate unless the two are close, in which case prefer the more popular one — is naturally written as a thresholded lexicographic utility. Define, for candidate $k \in \{i, j\}$ and history H,

$$u(k, H) \;=\; \mathrm{cov}(k, H) \;+\; \varepsilon \cdot \mathbf{1}[\,|\mathrm{cov}(i, H) - \mathrm{cov}(j, H)| \le \delta\,] \cdot \mathrm{pop}(k), \tag{4}$$

for a small $\varepsilon > 0$ and closeness threshold $\delta \ge 0$, so that the teacher's target decision is $y^\star(i, j, H) = \arg\max_{k \in \{i,j\}} u(k, H)$. Popularity enters the decision *only* through the indicator term, i.e. only when content match is within δ of a tie — exactly the tie-breaking behavior we verify empirically in Section 3.2. This is a population-level idealization of the teacher's actual behavior (a free-text-justified LLM judgment), not a rule it is given explicitly; we use it only to make precise what a debiased decision rule looks like in this feature space.

GRPO objective. For a prompt q (a candidate pair and history) drawn from a data source $D \in \{D_{\text{align}}, D_{\text{sens}}\}$, the current policy $\pi_{\theta_{\text{old}}}$ samples a group of G responses $\{o_1, \ldots, o_G\}$. Writing $\rho_{k,t}(\theta) = \pi_\theta(o_{k,t} \mid q, o_{k,<t}) / \pi_{\theta_{\text{old}}}(o_{k,t} \mid q, o_{k,<t})$ for the per-token importance ratio, the student is trained to maximize

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q \sim D,\, \{o_k\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{k=1}^{G} \frac{1}{|o_k|} \sum_{t=1}^{|o_k|} \min\Big(\rho_{k,t}(\theta) \hat{A}_k,\; \mathrm{clip}\big(\rho_{k,t}(\theta), 1-\epsilon, 1+\epsilon\big) \hat{A}_k \Big) \right] - \beta\, \mathbb{E}_{q \sim D}\big[D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}}) \big] \tag{5}$$

with the group-relative, standardized advantage

$$\hat{A}_k \;=\; \frac{r_k - \mathrm{mean}(r_1, \ldots, r_G)}{\mathrm{std}(r_1, \ldots, r_G)}, \qquad r_k \;=\; \mathbf{1}[\,\hat{y}(o_k) = y^\star(i, j, H)\,], \tag{6}$$

where $\hat{y}(o_k) \in \{A, B\}$ is the slot label the student extracts from rollout o_k and $y^\star(i, j, H)$ is the slot holding the teacher-preferred candidate under the current (randomized) slot assignment (Section 4). We use the low-variance unbiased KL estimator $D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) = \pi_{\text{ref}}(o \mid q)/\pi_\theta(o \mid q) - \log\big(\pi_{\text{ref}}(o \mid q)/\pi_\theta(o \mid q)\big) - 1$, following Shao et al. (2024). All four recipes of Section 3.4 share Equations 5–6 exactly; they differ only in whether D includes D_{sens}, whether r_k is replaced by the propensity-weighted reward of Eq. 3, and — for $\pi_{\text{curriculum}}$ alone — in D itself changing partway through training rather than staying fixed for the whole run (Section 3.5).

Adaptive KL-to-reference controller. Section 3.5's fix for entropy growth replaces the fixed KL coefficient β in Eq. 5 with the standard proportional controller (Ziegler et al., 2019),

$$\beta_{t+1} = \beta_t \Big(1 + \kappa \cdot \mathrm{clip}\big((D_{\text{KL}}(\pi_{\theta_t} \| \pi_{\text{ref}}) - D^\star)/D^\star,\; -0.2,\; 0.2 \big) \Big), \tag{7}$$

for a target per-step KL budget $D^\star$ and gain κ (values in Appendix E): β grows when observed drift exceeds the target and shrinks when it falls short, bounding cumulative drift directly rather than hoping a static β stays in range for an entire run.

Position shortcut as a degenerate optimum. Let $c(i, j, H) \in \{A, B\}$ denote the prompt slot into which the teacher-preferred candidate is placed. If c is not randomized — e.g. the positive item is

always placed in slot A, so $c \equiv A$ — then the constant policy π_0 that ignores its input and always outputs slot A satisfies, for any evaluation distribution D constructed the same way,

$$\mathrm{Acc}(\pi_0) \;=\; \mathbb{E}_{(i,j,H)\sim D}\big[\mathbf{1}[\,c(i,j,H) = A\,]\big] \;=\; \Pr_{D}[y^\star = A]\,, \tag{8}$$

i.e. the constant policy's accuracy equals the label base rate exactly, independent of any signal in (i, j, H). Because $\pi_0 \in \arg\max_\pi \mathbb{E}[r]$ whenever c is deterministic, gradient-based optimization of Equation 5 has no incentive to escape π_0, and empirically converges to a policy statistically indistinguishable from it: this is precisely the diagnostic used to detect the failure mode described in Appendix D, where measured held-out accuracy matched the right-hand side of Equation 8 to more than ten decimal digits. Randomizing $c(i, j, H)$ independently per example removes this degenerate optimum without changing $y^\star$ itself, since the teacher's preference is a property of the underlying candidates, not of the slot they are shown in.

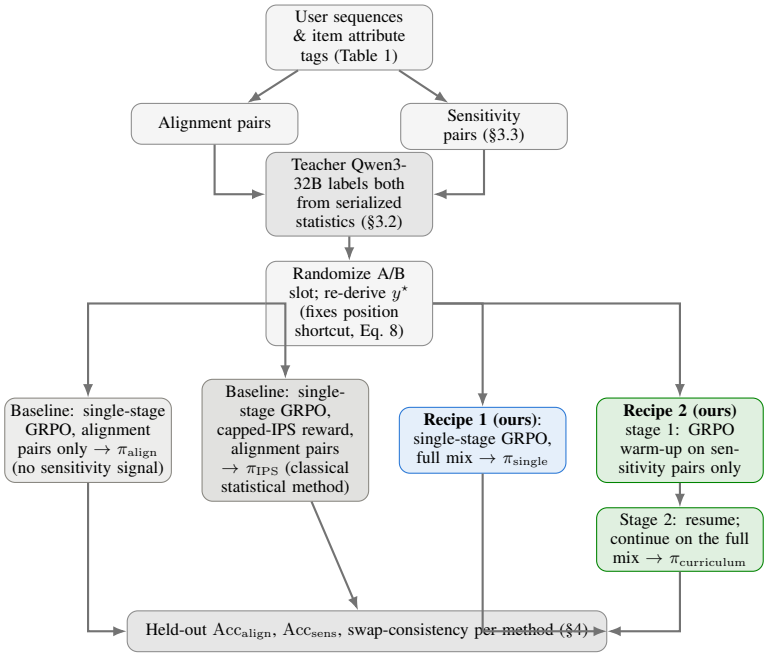


Figure 3: Complete overview of the pipeline. Real interaction sequences become alignment and sensitivity pairs, both labeled by a 32B teacher reasoning only over serialized popularity and attribute-overlap statistics. After the slot-randomization fix (§3.6), a 1.7B student is trained under Recipe 1 (single-stage, blue), Recipe 2 (curriculum, green, two stages), and two baselines using no counterfactual signal (alignment-only and capped-IPS, gray). All four are evaluated on the same held-out metrics.

C Prompt Templates

Teacher prompt. The teacher receives, for a user history $H = [h_1, \ldots, h_k]$ and two candidates A, B:

```
User history (attribute sets, in order): [{attrs(h_1)},
..., {attrs(h_k)}]
Candidate A: attributes = {attrs(A)}, global popularity
percentile = {pop(A)}, N history items sharing ≥1 attribute
= {overlap_count(A)}, fraction of A's attributes seen in
history = {cov(A, H)}
Candidate B: [analogous fields for B]
Instruction: higher popularity means more people bought it,
not that this user prefers it. Prioritize content match;
use popularity only as a tie-breaker when content match
is close. Briefly justify your choice, then answer with
exactly ``#### A'' or ``#### B'' on the final line.
```

Student prompt. The student receives only attribute sets, with no statistics and no item identifiers:

```
User history (attribute sets, in order): [{attrs(h_1)},
..., {attrs(h_k)}]
Candidate A's attributes: {attrs(A)}
Candidate B's attributes: {attrs(B)}
Instruction: decide which candidate the user is more likely
to interact with next. You must commit to A or B even if
uncertain. Answer with exactly ``#### A'' or ``#### B'' on
the final line.
```

D Implementation Pitfalls

We record four failure modes encountered while implementing this method, in enough detail to be diagnosable in a re-implementation.

Position shortcut. In an initial version of the pair construction, the teacher-preferred item (the true positive, for alignment pairs) was always placed in the same prompt slot. A model trained on this data reached accuracy far above chance within a handful of steps; the symptom that revealed this was a shortcut rather than genuine learning was that the trained model's accuracy on held-out data matched $P(\text{ground truth} = \text{that slot})$ in the evaluation set to more than ten decimal digits, which is exactly what a model that always answers with the fixed slot — regardless of content — would produce. The fix, which does not require re-running teacher labeling, is to randomize the slot assignment independently per row after labeling and re-derive the label from whichever physical item the teacher preferred, since the teacher's preference is a property of the item rather than of the slot it happened to occupy when labeled.

Identifier-driven entropy collapse. A version of the student prompt that included bare item identifiers alongside attribute tags trained to near-chance accuracy: policy entropy fell to near zero within approximately 15 GRPO steps, after which the GRPO advantage — computed within a group of otherwise-identical rollouts — was exactly zero for the remainder of training, since a deterministic policy produces no reward variance within a group. We attribute this to item identifiers carrying no information a language model can transfer across examples (each identifier is functionally unique), unlike attribute tags, which repeat across hundreds of items in this dataset and are the only substrate in this feature space with cross-example transferable signal. Removing raw item identifiers from the student prompt entirely, increasing the entropy coefficient, and increasing the learning rate were the changes made in response, and are the configuration used for every run reported in Section 5.

Entropy-growth instability (the opposite failure, at longer horizons). The fix above removes entropy *collapse* but introduces the opposite risk: with a fixed positive entropy coefficient and no upper bound on entropy itself, policy entropy is free to grow without limit once the policy-gradient term's magnitude shrinks (which it does as the policy approaches any local optimum of the sparse binary reward). In our original 100-step probe run (alignment-only ablation), entropy grew roughly monotonically and, past step $\approx$40, approximately exponentially: 0.52 nats at step 40, 3.88 at step 60, 11.22 by step 96, at which point held-out accuracy on both alignment and sensitivity pairs was exactly 0.0 and response perplexity exceeded 7×10^4 — complete policy collapse into incoherent output. The instability is present, at lower intensity, in every 40-step run reported in Section 5.1 (entropy reaching 1.08–1.43 nats by step 36 depending on variant), and is far more severe for the IPS-weighted baseline (8.86 nats by step 36), which we attribute to its popularity-reweighted reward having substantially higher variance across otherwise-identical prompt groups than the plain binary reward used by the other two variants (Section 5.1). We are not aware of a fix we can recommend with confidence from a single run of diagnosis; the GRPO literature we are aware of tunes the entropy coefficient once and does not discuss it drifting between under- and over-regularizing entropy within a single run at a fixed coefficient, which is what we observed. A capped or decaying entropy bonus, or a validation-tracked early-stopping rule keyed to entropy itself, both seem like more promising fixes than a fixed coefficient. Section 3.5 describes the fix we subsequently tested (an adaptive KL-to-reference controller in place of the fixed entropy coefficient, bounding cumulative policy drift directly rather than hoping a fixed bonus stays in range for the whole run), with results in Section 5.2.

Stopping-rule failure mode (uncovered once entropy growth was fixed). Round 2's adaptive KL controller resolved the entropy-growth instability above (peak entropy down one to two orders of magnitude for every variant, Table 5), which let us apply the pre-registered stopping rule of Section 5.2's follow-up sketch without an instability confound. Applied to the single-stage full model's run, the rule selected step 10 — its worst-performing checkpoint among the four Round 2 variants on held-out sensitivity accuracy — because sensitivity accuracy dipped at steps 15 and 20 relative to step 10, tripping the rule's 2-consecutive-failures patience criterion, even though the eval_dev curve continued climbing afterward to a run-best 70.00% by step 50 (Section 5.2). The rule is not misimplemented: a patience-based stopping criterion evaluated against a single noisy validation curve is, by construction, exposed to exactly this kind of local dip, and nothing in our setup gives it a way to distinguish a genuine peak from a temporary one without seeing future steps it is explicitly not allowed to peek at. We do not have a fix we can recommend with confidence from one occurrence; averaging the selected step (or the validation metric that drives selection) over multiple seeds is the most direct mitigation, at the cost of multiplying the compute budget by the number of seeds averaged.

Table 5: Peak policy entropy (nats) reached during each Round 2 method's full 50-step run, vs. the corresponding Round 1 figures (all pre-stopping-rule-selection). Referenced from Section 5.2.

Method	Round 1 (40 steps)	Round 2 (50 steps)
Alignment-only	1.08	0.50
IPS-weighted, capped	8.86	0.18
Single-stage (ours)	1.43	0.70
Curriculum (ours)	—	0.18

E HYPERPARAMETERS

Counterfactual twin search requires at least $\min(2, |A(i)|)$ shared attributes and a popularity-percentile gap of at least 0.4. Student training uses GRPO with group size $G = 8$, train batch size 32, PPO mini-batch size 32, and a train/evaluation user split of 90/10 with a fixed random seed. Both teacher and student use greedy or near-greedy decoding with a forced-choice final answer; the student's chat template disables extended chain-of-thought generation, since responses of this kind otherwise exhaust the response token budget before reaching a parseable answer.

Round 1 (Section 5.1). Actor learning rate 5×10^{-6}, fixed entropy coefficient 0.005, fixed KL loss coefficient 0.001 (low-variance KL estimator, applied as a loss term). PPO clipping range $\epsilon = 0.2$. IPS propensity smoothing $\eta = 0.01$. All three GRPO variants were trained for exactly 40 steps (not the 100-step budget originally planned, for the reason given in Section 5.1 and Appendix D), with validation every 10 steps and a checkpoint saved at step 40. Rollout sampling uses temperature 1.0 and top-$p = 0.9$.

Round 2 (Section 5.2). Actor learning rate 3×10^{-6} with a cosine schedule (10% linear warmup); entropy coefficient 0.005 retained only as an exploration floor (Section 3.5); the fixed KL loss term is replaced by an adaptive KL-to-reference controller in the reward (Section 3.5) with initial coefficient $\beta_0 = 0.001$, target per-step KL $D^\star = 0.05$, controller gain $\kappa = 0.05$, and horizon equal to the run's step budget; capped IPS weight $w_{\max} = 10$; sensitivity-pair oversampling factor $3\times$ for the single-stage and curriculum variants. PPO clipping range $\epsilon = 0.2$. IPS propensity smoothing $\eta = 0.01$. All variants trained for 50 steps at a single seed, with validation every 5 steps and a checkpoint saved at every validation step, so the stopping rule of Section 5.1 has a checkpoint available at every candidate step. Rollout sampling uses temperature 1.0 and top-$p = 0.9$.

F WHY GRPO RATHER THAN SFT OR DPO

Given that the training signal consists of static binary labels generated offline by the teacher, a natural question is why we use online policy-gradient optimization (GRPO) rather than simpler offline preference distillation methods such as Supervised Fine-Tuning (SFT) on the teacher's labels or Direct Preference Optimization (DPO) (Rafailov et al., 2023). Our choice is motivated by three

considerations. **First**, GRPO's group-relative advantage normalization implicitly reweights the loss by intra-group reward variance, which provides a form of implicit curriculum: easy pairs (where the model is already confident) contribute near-zero gradient, while hard pairs (where rollouts disagree) drive updates — a dynamic SFT's uniform cross-entropy cannot replicate. **Second**, GRPO's clipped surrogate objective bounds the per-step policy change, which we found empirically more stable than DPO's implicit reward modeling when the teacher's labels are themselves noisy (teacher self-consistency is 99.13% but not perfect). **Third**, GRPO's rollout-based sampling naturally extends to future work where the reward signal is not a pre-computed label but a learned reward model or a more complex verifiable objective. We nonetheless acknowledge that SFT and DPO are simpler, more stable baselines for this task, and their comparison is the clearest follow-up experiment (Section 6). In this paper, we treat the GRPO choice as a methodological commitment whose value must be judged against the instability diagnostics of Section 3.5 and Appendix D — not as a demonstrated advantage over offline distillation.

G Full-Scale Ranking Metrics Details

`rank_eval.py` implements Recall@10 and IPS-reweighted NDCG@10. Its evaluation protocol is a round-robin tournament: for each user's held-out item, every pair among the top-50 candidates is presented to the student and the student's forced-choice is recorded. This issues $\binom{50}{2} = 1225$ pairwise LLM calls per user, $\sim 300\times$ the accuracy evaluation's cost per model. The protocol is not blocked by instability (Round 2 checkpoints are stable) but was not completed in this revision's time budget across four models. The full-scale ranking evaluation remains the clearest next step.

H Cost Analysis Details

Benchmarking accuracy alone omits what a team choosing between these four methods actually pays for each. We measure two costs. **Training cost** (one-time): each method's real wall-clock training time and summed per-step compute time from verl's logged `perf/time_per_step`. **Inference cost** (recurring): wall-clock inference on $N{=}30$ held-out pairs (vLLM, continuous batching at batch size 30), reporting single-request latency and throughput for teacher and student.

Table 6: Real training cost, from job start/end timestamps and verl's logged `perf/time_per_step`. Curriculum's row sums its two sequential stages.

Method	Runs	Wall-clock	Step time
Alignment-only	1	37.3 min	22.3 min
IPS-weighted	1	34.5 min	19.5 min
Single-stage (ours)	1	36.1 min	20.7 min
Curriculum (ours)	2	51.8 min	21.3 min

Table 7: Real inference cost on the same hardware ($N{=}30$ held-out pairs, vLLM, eager mode). Latency times one prompt at a time; throughput is under continuous batching at batch size 30. All four trained student variants share the student row.

Model	Params	Out. tokens	Latency (ms)	Throughput (/s)
Teacher (reasoning)	32B	183	5,915	1.89
Student (no reasoning)	1.7B	121	1,293	10.12

Training is one-time. Curriculum's wall-clock (51.8 min) is $\sim 1.4\times$ a single-stage run's (34.5–37.3 min), but its summed per-step compute time (21.3 min) matches the other methods' (19.5–22.3 min) — the extra wall-clock is almost entirely the fixed ∼15-minute cold-start of launching a second training job, not extra gradient computation. Operationally: infrastructure that keeps the engine warm and resumes in-process pays little extra; a job scheduler that treats each stage as a fresh submission roughly doubles the overhead.

Inference is recurring. All four trained variants share architecture and decoding config, so serving cost is identical and 4.6–$5.4\times$ cheaper than the teacher's (Table 7) regardless of recipe — confirming

Section 1's motivation by measurement, not assumption. Neither latency is millisecond-scale on this unoptimized setup; a tuned stack would lower both, but the relative gap should persist. Curriculum's gain thus costs a one-time training premium with *zero* recurring cost — favorable if it replicates across seeds.